Unveiling Memorization–Generalization Coexistence: A Case Study on Arithmetic Tasks with Label Noise

Linyu Liu [1] **Pinyan Lu** [1 2]

Abstract

Highly over-parameterized models can simultaneously memorize noisy labels and generalize well, yet how these behaviors coexist remains poorly understood. In this work, we investigate the underlying mechanisms of this coexistence using modular arithmetic tasks under heavy label noise. Through extensive experiments on two-layer neural networks, we find that larger models tend to generalize better under appropriate optimization and model configurations, while noisy labels are memorized faster than clean data. Over-parameterized models internally form a generalization structure, but its expression in the output is suppressed by the need to fit noisy labels. Remarkably, even with 80% label noise, near-perfect test accuracy can be achieved by extracting this internal structure using frequency-based methods. We further propose a task-agnostic method to partition networks into generalization and memorization components. Although this subnetwork improves generalization, it is limited compared with frequency-based extraction, indicating that the generalization structure is distributed across neurons and motivating the development of new tools to retrieve generalizable knowledge from over-parameterized networks.

1. Introduction

Large Language Models succeed largely because increasing model capacity consistently improves generalization (Brown et al., 2020; Kaplan et al., 2020; Henighan et al., 2020; Hoffmann et al., 2022). In structured tasks, larger models exhibit emergent abilities to capture underlying rules from data (Wei et al., 2022). However, real-world training data is often contaminated with noise, such as mislabeled samples or factual inconsistencies. While the scaling paradigm suggests that larger models perform better in clean environments (Liu et al., 2025), it remains unclear how they handle corrupted data. Specifically, we need to understand the internal competition between two distinct components: generalization, which is the discovery of structural rules that apply to unseen data, and memorization, which is the fitting of specific noisy labels that do not reflect the underlying rule.

Analyzing this interaction in realistic tasks is challenging because generalization is often difficult to diagnose mechanistically. Unseen examples may share local patterns, semantic features, or distributional correlations with the training data, so high test accuracy does not by itself reveal whether the model has learned a true rule or exploited shortcuts. To address this, we use modular arithmetic tasks (such as addition and multiplication modulo P) as a controlled experimental environment. These tasks are ideal for several reasons. First, they provide a clear distinction between rule-based generalization and point-wise memorization (Pearce et al., 2023). Second, they exhibit grokking (Power et al., 2022; Nanda et al., 2023; Zhong et al., 2023), a phenomenon where generalization happens long after the model fully memorizes the training set. Third, these tasks allow for rigorous analysis because they have known analytical solutions under certain setups (Morwani et al., 2024; Tian, 2025).

In this work, by introducing label noise into this setup, we can precisely track how a model resolves the conflict between capturing a mathematical rule and memorizing random noise. We conduct extensive experiments across diverse settings, covering various activation functions, optimizers, and regularization strengths. This broad approach allows us to determine if the benefits of scaling persist under extreme corruption and how the model internally organizes these two competing behaviors. We focus on three key questions:

Q1. Under what conditions does increasing model size improve generalization when labels are noisy?

Q2. Can models still recover internal structural rules under extreme label noise?

Q3. Is it possible to separate the neurons responsible for

[1]Taylor Lab, Huawei Technologies Co., Ltd. [2]Key Laboratory of Interdisciplinary Research of Computation and Economics, Shanghai University of Finance and Economics. Correspondence to: Linyu Liu <liulinyu001@gmail.com>, Pinyan Lu <lu.pinyan@mail.shufe.edu.cn>.

Preprint. July 21, 2026.

generalization from those used for memorization?

Through these experiments, we provide the following findings:

1. We show that scaling model size continues to improve generalization even with noisy data, provided that the optimization and regularization are properly configured.
2. We observe a counter-intuitive learning dynamics where noisy labels are often memorized faster than the clean ones at a very early stage.
3. We find that models can *internally* recover the underlying algebraic rule under extreme label noise ($\sim 80\%$ noise), though these rules are suppressed by the need to interpolate noisy labels.
4. We demonstrate that partitioning the network has the potential to improve generalization, generalization and memorization remain deeply entangled at the neuronal level.

These results highlight both the robustness and the limitation of over-parameterized learning under corruption: models may learn generalizable rules even when forced to memorize noise, but current neuron-level tools cannot fully extract this rule structure from memorization. The rest of the paper elaborates on these findings. Section 2 introduces the problem setup. Section 3 provides an overview of accuracy performance from both the end-of-training and dynamics perspectives (Q1). Section 4 mechanically analyzes the internal representation of the generalization (Q2). Section 5 examines whether memorization and generalization can be separated at the neuron level (Q3). We defer related work to Appendix A.

2. Problem Setup

We study a modular arithmetic task $a \circ b = c$ following Power et al. (2022), where $a, b, c \in \{0, \ldots, P-1\}$ and $\circ$ denotes a modular operation such as addition, subtraction, or multiplication. We focus primarily on modular addition and use other operations to test generality. The task is formulated as a P-class classification problem.

We train a two-layer neural network parameterized by θ:

$$\boldsymbol{f}_\theta(a, b) = \boldsymbol{W}\,\phi(\boldsymbol{U}\boldsymbol{e}_a + \boldsymbol{V}\boldsymbol{e}_b) + \boldsymbol{\mu}, \tag{1}$$

where $\theta = \{\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{\mu}\}$. Here, $\boldsymbol{e}_a, \boldsymbol{e}_b \in \mathbb{R}^P$ are one-hot encodings of a and b; $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{M \times P}$ are first-layer weights; $\boldsymbol{W} \in \mathbb{R}^{P \times M}$ is the second-layer weight matrix; $\boldsymbol{\mu} \in \mathbb{R}^P$ is the bias; and M denotes the number of hidden neurons. The output $\boldsymbol{f}_\theta(a, b) \in \mathbb{R}^P$ is trained using cross-entropy loss to predict the one-hot label $\boldsymbol{e}_{a \circ b}$.

Unless stated otherwise, we fix $P = 113$ following Nanda et al. (2023) and vary the model width as $M = 2^k + 1$ for $k \in \{4, \ldots, 12\}$, since ReLU models empirically generalize better at odd widths in these tasks (Pearce et al., 2023).

Dataset. The total dataset is

$$\mathcal{D}_{\text{total}} = \{(a, b, c) : a, b \in \{0, \ldots, P-1\},\ c = a \circ b\}.$$

We randomly split $\mathcal{D}_{\text{total}}$ into training, validation, and test sets with ratios 50%, 20%, and 30%, respectively. In Appendix C.1, we also vary the training ratio while fixing the test ratio.

To introduce label noise, we randomly select a subset $\mathcal{D}_{\text{noise}} \subset \mathcal{D}_{\text{train}}$ with noise ratio α, i.e., $\frac{|\mathcal{D}_{\text{noise}}|}{|\mathcal{D}_{\text{train}}|} = \alpha$. For each $(a, b, c) \in \mathcal{D}_{\text{noise}}$, we define the observed label $\tilde{c}$ as

$$\tilde{c} = \text{Uniform}\big(\{0, \cdots, P-1\} \setminus \{c\}\big), (a, b, c) \in \mathcal{D}_{\text{noise}}.$$

The resulting noisy training subset is given by

$$\tilde{\mathcal{D}}_{\text{noise}} := \{(a, b, \tilde{c}) : (a, b, c) \in \mathcal{D}_{\text{noise}}\}.$$

The noisy training set is $\tilde{\mathcal{D}}_{\text{train}} = \mathcal{D}_{\text{clean}} \cup \tilde{\mathcal{D}}_{\text{noise}}$. All evaluations are performed on clean validation and test sets.

Evaluation. To measure the performance of a model $\boldsymbol{f}$, we define its accuracy on a dataset $\mathcal{D}$ as

$$\text{Acc}\left(\boldsymbol{f}, \mathcal{D}\right) = \frac{1}{|\mathcal{D}|} \sum_{(a,b,c) \in \mathcal{D}} \mathbb{I}\left(\hat{c} = c\right),$$

where $\hat{c} = \arg\max_{c'} \left[\boldsymbol{f}\left(a, b\right)\right]_{c'}$, and $\mathbb{I}\left(\cdot\right)$ is the indicator function.

Optimizers. We train models with cross-entropy loss using Adam, AdamW (Loshchilov & Hutter, 2017), and Muon (Jordan et al.), with weight decay controlling regularization strength. We also experiment with SGD, but it is inefficient and requires careful hyperparameter tuning to achieve generalization (see Figure 19).

Unless specified otherwise, all results are reported after 200,000 epochs of full-batch training. We apply a linear learning-rate warmup over the first 10 steps. The default setting uses ReLU activations with AdamW (learning rate 10^{-3}, weight decay 0.1, random seed 42) on the modular addition task.

3. When Bigger Is Better

In this section, we investigate how the presence of label noise interacts with model capacity in modular arithmetic tasks. We approach this question from two complementary perspectives. First, we examine end-of-training performance, asking under which conditions—such as optimization and regularization choices—increasing model size

consistently improves generalization despite noise. Second, we analyze the training dynamics to study whether the model learns clean labels and noisy labels at different rates.

3.1. End-of-Training Performance

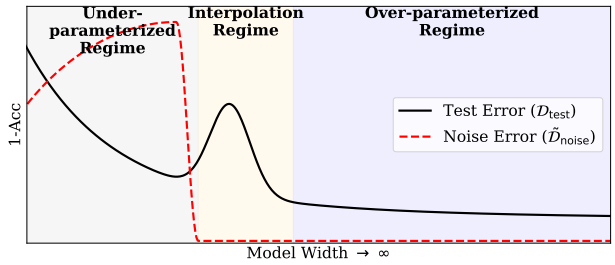


Figure 1. Illustration of double descent. Beyond the interpolation threshold, the over-parameterized regime corresponds to the second descent, where increasing model size leads to improved generalization, even after fitting all training samples, including label noise.

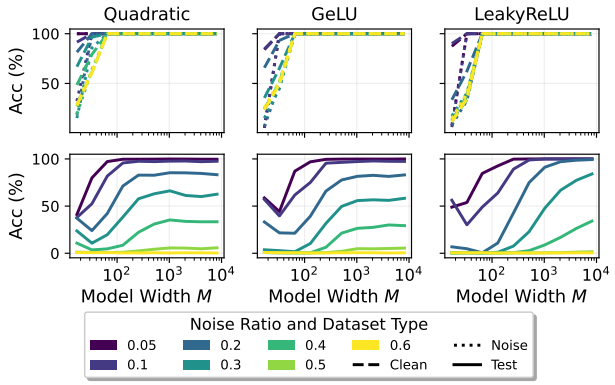


Figure 2. Performance across various activation functions. Generally, test accuracy is non-decreasing in the over-parameterized regime, but ceilings of test accuracy exist for the quadratic and GeLU activations.

We robustly observe the *double descent* phenomenon of the test *error* ($1 -$ test accuracy) across different activation functions, noise-to-training ratios, optimizers (including AdamW, Adam, and Muon), and arithmetic tasks (see the full results in Section C.1). As illustrated in Figure 1, *double descent* refers to the non-monotonic test-error curve in which generalization first worsens near the interpolation threshold and then improves again as model size further increases. In the over-parameterized regime, continually enlarging the model leads to non-decreasing test accuracy, while the threshold for entering this regime depends on the regularization strength. We summarize the observations in Result 3.1.

Result 3.1. *Double descent is consistently observed across varying setups:*

1. Within the over-parameterized regime, bigger models generalize better.

2. Strengthening regularization will shift models from the over-parameterized regime toward the under-parameterized regime. (Figure 3)

3. Model sizes of test-accuracy saturation are activation-dependent (Figure 2)

Regularization does play a role in shaping the double-descent curve by controlling the effective model complexity (EMC) (Nakkiran et al., 2021). Intuitively, stronger regularization restricts the set of functions that can be realized during training, effectively reduce the model's capacity. When the EMC is limited, learning signals for generalization and memorization compete more strongly. As a result, increasing regularization can be beneficial when memorization of label noise is undesirable.

Models with quadratic or GeLU activation functions exhibit clear ceilings in test accuracy as model size increases within the explored model size range (width $< 10{,}000$, Figure 2). We include quadratic activations here because their analytical solutions are available (see Section 4). One possible explanation for these ceilings is that the symmetry of these activations constrains how noise can be absorbed while preserving generalization. In contrast, ReLU activations store noise in a non-symmetric manner, while the representation of generalization remains symmetric (see Section 4), allowing model capacity to be used more flexibly and leading to improved generalization.

We also observe that models are more confident (i.e., exhibit lower test cross-entropy loss) in the over-parameterized regime than in the under-parameterized regime. Larger models are generally more confident (Figure 3, bottom row). Cross-entropy loss reflects the confidence calibration of model predictions. In the under-parameterized regime, models often predict the true label even for corrupted samples. To align the predicted probability with the portion of correctness, the model assigns lower probabilities, resulting in higher uncertainty and higher test cross-entropy. In contrast, in the over-parameterized regime, larger models can separate data with larger margins while still satisfying the same ℓ_2 regularization constraint. Larger margins lead to more confident predictions and, consequently, lower test cross-entropy loss.

Remarks on model misspecification. We note that some model architectures are unable to interpolate all noisy labels, in which case the over-parameterized regime does not strictly exist. For example, models with tied first-layer weights ($\boldsymbol{U} = \boldsymbol{V}$) cannot memorize asymmetric random noise where $a \circ b \neq b \circ a$. In such cases, test accuracy does not vary monotonically with model size, and selecting an appropriate regularization strength may be more effective than

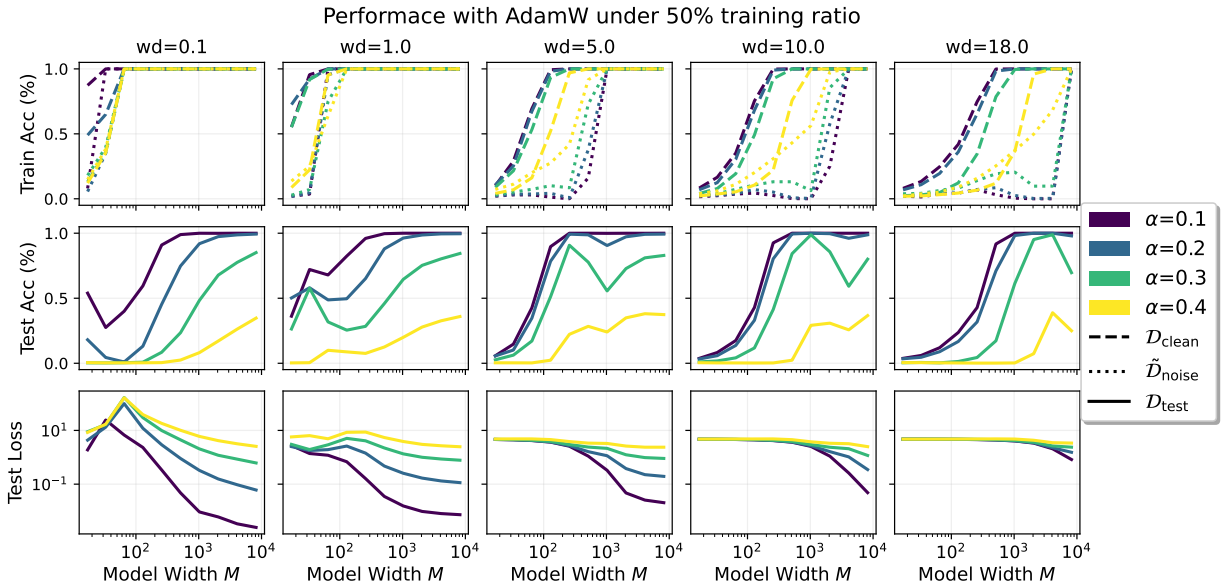


Figure 3. Performance of ReLU models trained with AdamW under varying weight decay. The first row shows training accuracy on clean ($\mathcal{D}_{\text{clean}}$) and noisy ($\tilde{\mathcal{D}}_{\text{noise}}$) data to characterize memorization. The second row shows test accuracy, and the third row shows test cross-entropy loss. Increasing weight decay shifts the interpolation threshold to larger sizes, delaying the second-descent regime.

increasing model capacity. We provide a detailed discussion in Section C.2.

3.2. Training Dynamics

We observe a counter-intuitive phenomenon in the training dynamics: noisy labels are consistently memorized faster than clean labels at the initial training epochs, as summarized in Result 3.2. While the learning rates for these two components are not drastically different, this inversion of the typical learning order is robust across various optimizers, including AdamW, Adam, and Muon (see Section C.4 for details). Even for SGD, which generally requires significantly longer timescales to achieve generalization, this trend remains consistent (Figure 19).

Result 3.2. *In both under- and over-parameterized regimes, the model prioritizes the memorization of noisy labels over the fitting of clean labels.*

Selected learning curves are shown in Figures 4 and 5. The faster memorization of noisy labels appears in both regimes, but the over-parameterized case in Figure 5 further reveals two patterns: (i) increased model capacity accelerates the memorization process, aligning with empirical findings in large language models (Tirumala et al., 2022); and (ii) the gap between the memorization speeds of noise and clean data becomes more pronounced at lower noise ratios α.

This observation stands in contrast to existing studies suggesting that models initially fit typical samples and ignore atypical ones, thereby making early stopping effective (Li

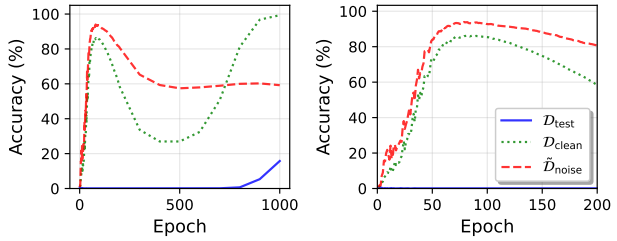


Figure 4. Under-parameterized learning dynamics with a zoom-in on the first 200 epochs ($\alpha = 0.3$, $M = 2049$, Adam with weight decay 0.001). Noisy-label accuracy rises earlier than clean-label accuracy.

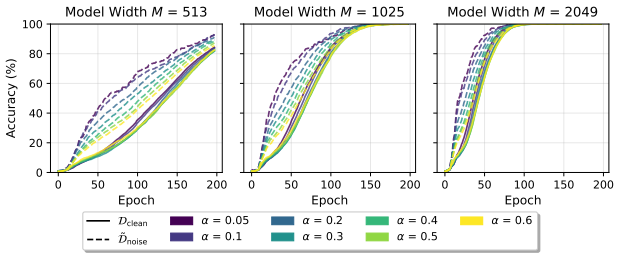


Figure 5. Over-parameterized learning dynamics under AdamW. Noisy-label accuracy rises earlier than clean-label accuracy.

et al., 2020). Our findings suggest a different early-stage dynamical behavior: under full-batch training, noisy labels can receive a larger per-sample optimization signal than clean labels, causing their accuracy to rise earlier. Importantly, this phenomenon is not restricted to two-layer networks on modular arithmetic tasks. In Section C.3, we provide a gradient-based discussion and show that a similar early noisy-over-clean learning order also appears on MNIST with a three-layer neural network under full-batch training. Thus, while modular arithmetic provides a controlled setting where the behavior can be cleanly measured and analyzed, the phenomenon may reflect a broader property of early full-batch optimization under explicit label corruption. Additional results are provided in Section C.4.

4. Representation of Generalization

We address Q2 by asking whether the apparent failure of ReLU networks under extreme label noise reflects the absence of a learned internal rule, or whether such a rule is learned but obscured by noisy outputs. We find evidence for the latter by separating the internal representation of the generalizable rule from that of memorization. Specifically, our main finding is that, although ReLU networks do not have the same expression as NN with quadratic activation, their generalization component closely matches the analytical solution known for quadratic activations under *clean* data (Morwani et al., 2024) (Equation (2)). Thus, it suggests a way to separate the learned rule from memorized noise: retain the dominant frequency component aligned with this analytical structure and remove the residual components.

Analytical solution for NN with quadratic activation. The two-layer NN (1) can be equivalently written as the summation of the neurons:

$$\boldsymbol{f}_\theta(a,b) = \sum_{m\in\mathcal{M}} \boldsymbol{w}_m \phi\left(\boldsymbol{u}_m^\top \boldsymbol{e}_a + \boldsymbol{v}_m^\top \boldsymbol{e}_b\right) + \boldsymbol{\mu},$$

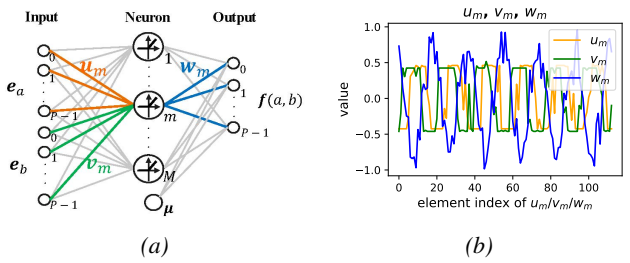
where $\mathcal{M} = \{1, \cdots, M\}$. The triple, $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$, represents the mth neuron. Here, $\boldsymbol{w}_m$ is the mth column vector of $\boldsymbol{W}$, and $\boldsymbol{u}_m^\top$ and $\boldsymbol{v}_m^\top$ are the mth row vectors of $\boldsymbol{U}$ and $\boldsymbol{V}$, respectively. Figure 6 visualizes this decomposition. The margin-maximizing solution for modular addition with quadratic activation $\phi(z) = z^2$ takes the form (Gromov, 2023; Morwani et al., 2024)

$$u_{mi} = \lambda \cos(\tfrac{2\pi}{P}\omega_m i + \varphi_m^{(a)}), \tag{2a}$$

$$v_{mj} = \lambda \cos(\tfrac{2\pi}{P}\omega_m j + \varphi_m^{(b)}), \tag{2b}$$

$$w_{mk} = \lambda \cos(\tfrac{2\pi}{P}\omega_m k + \varphi_m^{(c)}), \tag{2c}$$

with $\varphi_m^{(a)} + \varphi_m^{(b)} = \varphi_m^{(c)}$ and $\omega_m \in \{1, \ldots, \frac{P-1}{2}\}$ (Prop. B.1). This solution will achieve 100% accuracy on all the clean samples. Empirically, ReLU models trained on clean data also develop a periodic pattern in $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$ (Figure 6), but there is no explicit solution for them.



(a) *(b)*

Figure 6. **(a)** A neuron is represented by $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$. **(b)** A representative neuron from a trained ReLU model exhibits periodicity.

Frequency filtration separates rule learning from noise memorization. Drawing inspiration from the analytical solution of the NN with quadratic activation, we hypothesize that the generalization signal in a ReLU neuron is carried by its dominant frequency. We perform a Fourier-based decomposition on each hidden neuron $m \in \mathcal{M}$, characterized by the parameters $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$. Specifically, we apply a filter to isolate the frequency ω with the maximum magnitude in the weight vector $\boldsymbol{w}_m$, and retrieve the same frequency component from $\boldsymbol{u}_m$ and $\boldsymbol{v}_m$. This dominant component is referred to as the generalization part, $(\boldsymbol{u}_m^G, \boldsymbol{v}_m^G, \boldsymbol{w}_m^G)$, while the remaining frequency components constitute the residual part, $(\boldsymbol{u}_m^R, \boldsymbol{v}_m^R, \boldsymbol{w}_m^R)$. The formal mathematical definitions of this filtration process are detailed in Appendix D.1. We subsequently construct two sub-networks: one composed of the dominant frequency components, denoted by $\{\boldsymbol{U}^G, \boldsymbol{V}^G, \boldsymbol{W}^G, \boldsymbol{\mu}\}$, and another consisting of the residuals, $\{\boldsymbol{U}^R, \boldsymbol{V}^R, \boldsymbol{W}^R, \boldsymbol{\mu}\}$. By evaluating these partitioned models separately, we quantify their respective contributions to memorization and generalization.

Result 4.1. *The generalization representation of ReLU models closely matches the analytical solution* (2) *for modular addition and subtraction tasks.*

Figure 7 shows that the dominant-frequency sub-network achieves high test accuracy even under severe label noise, whereas the residual component is much more aligned with noisy-label memorization. This provides direct evidence that generalization and memorization coexist in the trained network but are encoded differently in frequency space.

The rule-separation effect of FF is surprisingly strong. Even when the model is trained with 80% label noise, the dominant-frequency sub-network can recover near-perfect test accuracy, indicating that the model has internally learned the correct rule although its raw output is heavily corrupted by noise memorization. Conversely, the residual sub-network largely preserves the ability to fit noisy labels while contributing little to test generalization. This sharp contrast suggests that label noise does not simply prevent the model from learning the rule; instead, the rule is learned but

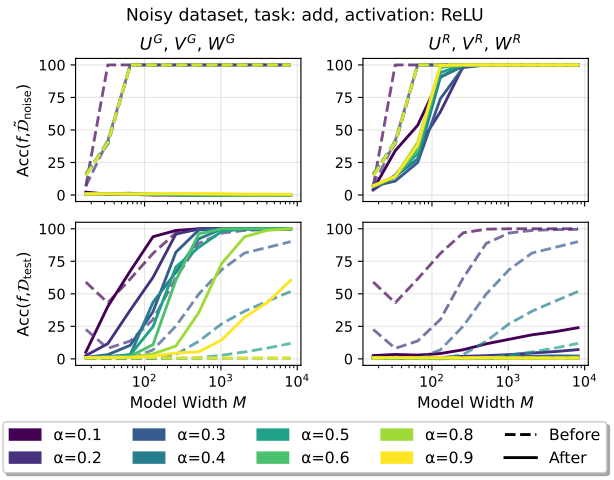


Figure 7. FF for ReLU networks. The dominant-frequency component $\{U^G, V^G, W^G\}$ recovers high test accuracy (after FF), while the residual component $\{U^R, V^R, W^R\}$ mainly retains noise memorization.

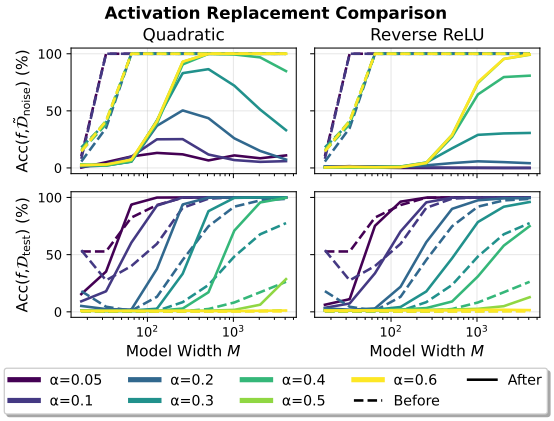


Figure 8. Replacing the trained ReLU activation with quadratic or reverse ReLU preserves or even improves test accuracy and reduces noisy-label accuracy, suggesting that the learned weights already encode a rule representation compatible with other activation functions.

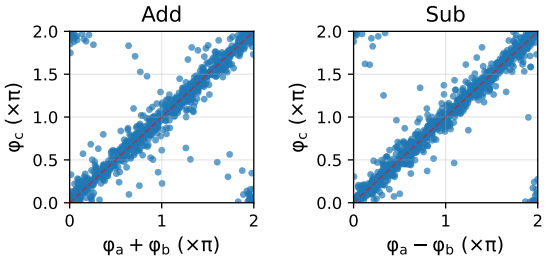


Figure 9. Relationship of phases under $M = 1025$ and $\alpha = 0.3$. Each point represents a neuron, where φ_a, φ_b, and φ_c are the phases of $\boldsymbol{u}^G$, $\boldsymbol{v}^G$, and $\boldsymbol{w}^G$, respectively.

masked by additional residual frequency components used for memorization. We further observe that FF also improves test accuracy for models trained on noise-free samples, for networks with quadratic activations, and for models on modular subtraction tasks, suggesting that dominant-frequency filtration is not merely removing noisy artifacts but extracting a general rule representation. The same frequency-based filtering idea can also be extended to a Transformer (see Appendix D.2).

Additional evidence for the analytical structure. We provide two complementary checks that the ReLU representation is coarsely aligned with the analytical solution. First, activation replacement shows that the learned rule does not rely on the exact shape of ReLU: after training, replacing ReLU with quadratic or reverse ReLU (defined as $\phi(x) = \max\{-x, 0\}$) preserves and often enhances generalization while reducing noisy-label accuracy (Figure 8). This is consistent with the hypothesis that the weights have already formed a symmetric rule representation.

Second, the phases of the filtered dominant components match the phase relationship in the analytical solution. For addition, the phases satisfy $\varphi_m^{(a)} + \varphi_m^{(b)} \approx \varphi_m^{(c)}$; for subtraction, they satisfy $\varphi_m^{(a)} - \varphi_m^{(b)} \approx \varphi_m^{(c)}$ (Figure 9). These results, together with FF and activation replacement, support the same conclusion: ReLU networks generalize by forming a frequency-structured representation close to the quadratic analytical solution, enabling the disentanglement of the generalization and memorization in the internal representation. Additional experiment results are deferred to Appendix D.3-D.4.

The above evidences suggest that FF can disentangle memorization from generalization. However, FF is inherently limited in two aspects: (i) it does not yield a strict functional decomposition, since $\boldsymbol{f}_\theta \neq \boldsymbol{f}_{\{\boldsymbol{W}^G, \boldsymbol{U}^G, \boldsymbol{V}^G, \boldsymbol{\mu}\}} + \boldsymbol{f}_{\{\boldsymbol{W}^R, \boldsymbol{U}^R, \boldsymbol{V}^R, \boldsymbol{\mu}\}}$; and (ii) it relies on prior knowledge of task-specific structure. This motivates the question of whether memorization and generalization can instead be separated in a task-agnostic manner (Q3).

5. Network Partitioning

We now address Q3 by asking whether generalization and memorization can be separated at the neuron level. Specifically, we investigate whether a trained network can be partitioned into two neuron subsets: one subset $\mathcal{M}^G \subseteq \mathcal{M}$ responsible for generalization, and another subset $\mathcal{M}^R \subseteq \mathcal{M}$ responsible for memorizing label noise. Selecting $\mathcal{M}^G$ for generalization is equivalent to pruning the remaining neurons from the full network. For a selected neuron subset, the corresponding sub-networks are defined as

$$\boldsymbol{f}_{\mathcal{M}^G}(a,b) = \sum_{m \in \mathcal{M}^G} \boldsymbol{w}_m \phi\big(\boldsymbol{u}_m^\top \boldsymbol{e}_a + \boldsymbol{v}_m^\top \boldsymbol{e}_b\big) + \boldsymbol{\mu}, \quad (3a)$$

$$\boldsymbol{f}_{\mathcal{M}^R}(a,b) = \sum_{m \in \mathcal{M}^R} \boldsymbol{w}_m \phi\big(\boldsymbol{u}_m^\top \boldsymbol{e}_a + \boldsymbol{v}_m^\top \boldsymbol{e}_b\big) + \boldsymbol{\mu}. \quad (3b)$$

We rank neurons using an importance score and select high-scoring neurons for generalization and low-scoring neurons for noise memorization. The full selection procedure is given in Algorithm 1.

Algorithm 1 Neuron Selection for Network Decomposition

Input: Network $\boldsymbol{f}_\theta$ with M neurons $\mathcal{M}$; Metric $S(m)$; $\mathcal{D}_{\text{val}}$; $\tilde{\mathcal{D}}_{\text{noise}}$; Threshold τ.
Initialize: $\mathcal{M}^G \leftarrow \emptyset, \mathcal{M}^R \leftarrow \emptyset$.
{// Step 1: Ranking neurons}
Calculate score $s_m = S(m)$ for each $m \in \mathcal{M}$.
Sort neurons descending: $\{m_{\pi(1)}, \ldots, m_{\pi(M)}\}$ where $s_{m_{\pi(i)}} \geq s_{m_{\pi(j)}}$ for $i < j$.
{// Step 2: Selection for generalization}
for $k \in \{1, \ldots, M\}$ **do**
  $\mathcal{M}_k = \{m_{\pi(1)}, \ldots, m_{\pi(k)}\}$; $A_k = \text{Acc}(\boldsymbol{f}_{\mathcal{M}_k}, \mathcal{D}_{\text{val}})$.
end for
$k^G = \arg\max_k A_k$; $\gamma^G = k^G/M$ and $\mathcal{M}^G = \{m_{\pi(1)}, \ldots, m_{\pi(k^G)}\}$.
{// Step 3: Minimal selection for noise memorization}
for $k \in \{1, \ldots, M\}$ **do**
  $\mathcal{M}'_k = \{m_{\pi(M-k+1)}, \ldots, m_{\pi(M)}\}$. {Bottom-up}
  if $\text{Acc}(\boldsymbol{f}_{\mathcal{M}'_k}, \tilde{\mathcal{D}}_{\text{noise}}) \geq \tau$ **then**
    $k^R = k$, **break**.
  end if
end for
$\gamma^R = k^R/M$ and $\mathcal{M}^R = \{m_{\pi(M-k^R+1)}, \ldots, m_{\pi(M)}\}$.
Output: Ratios γ^G, γ^R; Selected neurons $\mathcal{M}^G, \mathcal{M}^R$; Sub-networks $\boldsymbol{f}^G = \boldsymbol{f}_{\mathcal{M}^G}, \boldsymbol{f}^R = \boldsymbol{f}_{\mathcal{M}^R}$.

A natural importance score is the inverse participation ratio (IPR) (Pastor-Satorras & Castellano, 2016; Girvin & Yang, 2019; Gromov, 2023; Doshi et al., 2024), proposed and applied in Doshi et al. (2024). It measures the periodicity of each neuron and is defined as

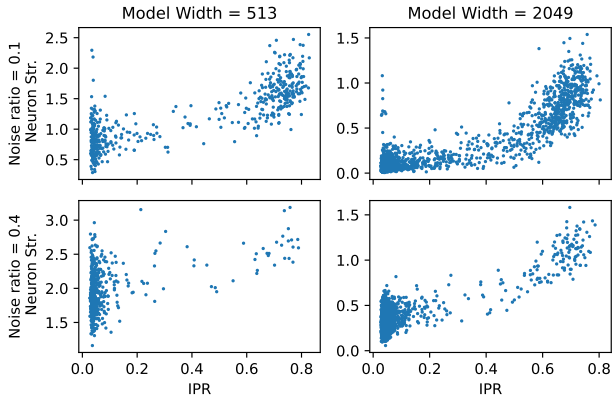


Figure 10. Neuron's IPR vs. Str. on the modular addition task. Neurons with stronger periodicity also tend to have larger Str., supporting Str. as a task-agnostic proxy for neuron importance.

$$\text{IPR}_m := \left(\frac{\|\tilde{\boldsymbol{w}}_m\|_4}{\|\tilde{\boldsymbol{w}}_m\|_2}\right)^4, \quad (4)$$

where $\|\cdot\|_p$ is the ℓ_p-norm and $\tilde{\boldsymbol{w}}_m$ is the Fourier transform of $\boldsymbol{w}_m$. Equation (4) implies $\text{IPR}_m \in [1/M, 1]$, where a larger value indicates stronger periodicity. This metric is well-suited for modular addition (Doshi et al., 2024), whose generalizable solutions rely on periodic structures.

To avoid relying on a task-specific Fourier prior, we also propose a task-agnostic score, neuron strength (Str.), defined as

$$\text{Str}_m := \|\boldsymbol{w}_m\|_\infty \phi\left(\|\boldsymbol{u}_m\|_\infty + \|\boldsymbol{v}_m\|_\infty\right), \quad (5)$$

where $\|\cdot\|_\infty$ denotes the ℓ_∞-norm. Intuitively, Str. measures the largest possible contribution of a neuron to the output margin. Unlike IPR, it does not assume periodicity and can therefore be applied to tasks such as modular multiplication, where periodicity in the original input domain is no longer a direct proxy for generalization.

Result 5.1. *Neurons that contribute most to generalization typically exhibit higher neuron strength. Specifically, neurons with high IPR also attain high Str. in modular addition and subtraction. (Figure 10)*

Result 5.2. *The sub-networks are not overlapped when model width is sufficiently large, i.e., $\gamma^G + \gamma^R \leq 1$, where $\gamma^G = \frac{|\mathcal{M}^G|}{|\mathcal{M}|}$ and $\gamma^R = \frac{|\mathcal{M}^R|}{|\mathcal{M}|}$. (Figure 11)*

Figure 10 shows that IPR and Str. are highly correlated. We also observe that a larger noise ratio reduces the number of high-IPR or high-Str. neurons. The optimal neuron ratio for generalization (γ^G) is also lower at a higher noise ratio (Figure 11). These may explain the deterioration of generalization under heavy label noise. Additional results in Appendix E show that Str. yields performance comparable

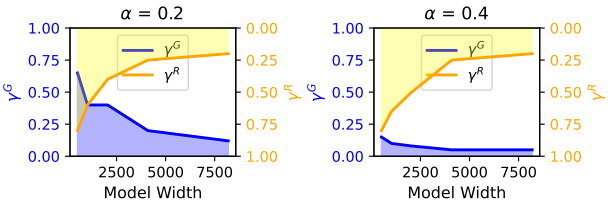


Figure 11. Neuron selection ratios using Str. on the modular addition task. Higher noise ratio leads to a smaller sub-network for generalization (γ^G). Two sub-networks have no overlapping neurons under sufficiently large model width.

to IPR on addition and subtraction, while being more effective on modular multiplication, where IPR is less suitable.

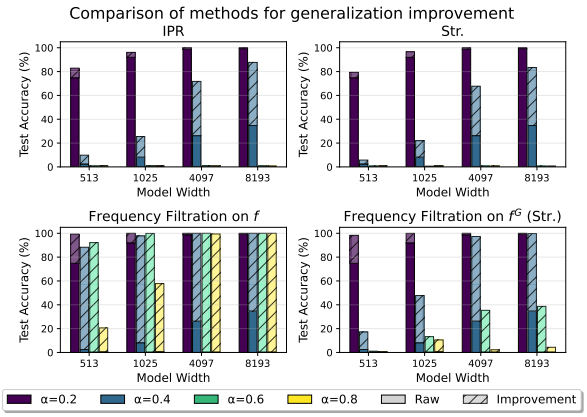


Figure 12. Generalization improvement on modular addition. IPR and Str. achieve similar performance. Applying FF to the sub-network f^G selected by Str. further improves generalization, but the gain remains substantially smaller than applying FF directly to the full model f.

Result 5.3. *Generalization improvement from neuron selection/pruning is limited compared with FF.*

Figure 12 shows that neuron selection can improve generalization, especially under light label noise, and that the task-agnostic Str. score performs comparably to the task-specific IPR score. However, this improvement is much weaker than the gain achieved by applying FF directly to the full model. Even applying FF after selecting the generalization sub-network f^G does not recover the performance of FF on the original network. This contrast indicates that generalizable structure is not localized to a small subset of neurons; instead, it is diffusely encoded across the network and intertwined with components used for noise memorization.

This comparison clarifies the limitation of task-agnostic post-hoc neuron selection. Unlike FF, which uses the domain-specific Fourier structure of the task to extract the latent rule, neuron selection can only identify neurons that are more important on average. As a result, pruning may partially improve generalization but cannot fully unleash the rule already learned by the model. This suggests two broader implications: future learning paradigms may need to explicitly isolate memorization from logical inference during training, and post-hoc knowledge extraction may require domain-specific structural priors rather than generic neuron-importance scores.

Conclusion. Our results demonstrate that over-parameterized models can internally learn underlying rules even under explicit label noise, though this generalization remains heavily entangled with memorization across distributed neurons. While task-agnostic neuron selection provides only limited improvement, frequency filtration can successfully recover latent rules when a suitable structural prior is available. Crucially, we observe that these core entanglement mechanisms extend beyond algebraic tasks to architectures like Transformers and datasets such as MNIST. We acknowledge that our setup inherently abstracts away the complex variability of natural vision or language tasks, but this controlled setting is necessary to provide a rigorous, verifiable boundary between structural generalization and unstructured memorization. Ultimately, this foundational study highlights a fundamental limitation of post-hoc pruning-based extraction, suggesting that fully disentangling these processes in practical applications will require novel training paradigms or architectural designs with explicit function separation.

References

Adlam, B. and Pennington, J. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*, pp. 74–84. PMLR, 2020.

Advani, M. S., Saxe, A. M., and Sompolinsky, H. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019. doi: 10.1073/pnas.1903070116. URL https://www.pnas.org/doi/abs/10.1073/pnas.1903070116.

Belkin, M., Hsu, D., and Xu, J. Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180, 2020.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Davies, X., Langosco, L., and Krueger, D. Unifying grokking and double descent. *arXiv preprint arXiv:2303.06173*, 2023.

Doshi, D., Das, A., He, T., and Gromov, A. To grok or not to grok: Disentangling generalization and memorization on corrupted algorithmic datasets. In *The Twelfth International Conference on Learning Representations*, 2024.

Gamba, M., Englesson, E., Björkman, M., and Azizpour, H. Deep double descent via smooth interpolation. *arXiv preprint arXiv:2209.10080*, 2022.

Girvin, S. M. and Yang, K. *Modern condensed matter physics*. Cambridge University Press, 2019.

Gromov, A. Grokking modular arithmetic. *arXiv preprint arXiv:2301.02679*, 2023.

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. Surprises in high-dimensional ridgeless least squares interpolation. *Annals of statistics*, 50(2):949, 2022.

Henighan, T., Kaplan, J., Katz, M., Chen, M., Hesse, C., Jackson, J., Jun, H., Brown, T. B., Dhariwal, P., Gray, S., et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Huang, Y., Hu, S., Han, X., Liu, Z., and Sun, M. Unified view of grokking, double descent and emergent abilities: A perspective from circuits competition. *arXiv preprint arXiv:2402.15175*, 2024.

Jordan, K., Jin, Y., Boza, V., Jiacheng, Y., Cecista, F., Newhouse, L., and Bernstein, J. Muon: An optimizer for hidden layers in neural networks, 2024. *URL https://kellerjordan. github. io/posts/muon*, 6.

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

Li, M., Soltanolkotabi, M., and Oymak, S. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International conference on artificial intelligence and statistics*, pp. 4313–4324. PMLR, 2020.

Liu, S., Zhu, Z., Qu, Q., and You, C. Robust training under label noise by over-parameterization. In *International Conference on Machine Learning*, pp. 14153–14172. PMLR, 2022a.

Liu, Y., Liu, Z., and Gore, J. Superposition yields robust neural scaling. *arXiv preprint arXiv:2505.10465*, 2025.

Liu, Z., Kitouni, O., Nolte, N. S., Michaud, E., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022b.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Mallinar, N. R., Beaglehole, D., Zhu, L., Radhakrishnan, A., Pandit, P., and Belkin, M. Emergence in non-neural models: grokking modular arithmetic via average gradient outer product. In *Forty-second International Conference on Machine Learning*, 2025.

McCracken, G., Moisescu-Pareja, G., Letourneau, V., Precup, D., and Love, J. Uncovering a universal abstract algorithm for modular addition in neural networks. *arXiv preprint arXiv:2505.18266*, 2025.

Mei, S. and Montanari, A. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766, 2022.

Morwani, D., Edelman, B. L., Oncescu, C.-A., Zhao, R., and Kakade, S. M. Feature emergence via margin maximization: case studies in algebraic tasks. In *The Twelfth International Conference on Learning Representations*, 2024.

Muthukumar, V., Vodrahalli, K., Subramanian, V., and Sahai, A. Harmless interpolation of noisy data in regression. *IEEE Journal on Selected Areas in Information Theory*, 1 (1):67–83, 2020.

Nakkiran, P., Venkat, P., Kakade, S., and Ma, T. Optimal regularization can mitigate double descent. *arXiv preprint arXiv:2003.01897*, 2020.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023.

Pastor-Satorras, R. and Castellano, C. Distinct types of eigenvector localization in networks. *Scientific reports*, 6 (1):18847, 2016.

Pearce, A., Ghandeharioun, A., Hussein, N., Thain, N., Wattenberg, M., and Dixon, L. Do machine learning models memorize or generalize?, August 2023. URL https://pair.withgoogle.com/explorables/grokking/. Accessed: 2024-01-20.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Somepalli, G., Fowl, L., Bansal, A., Yeh-Chiang, P., Dar, Y., Baraniuk, R., Goldblum, M., and Goldstein, T. Can neural nets learn the same model twice? investigating reproducibility and double descent from the decision boundary perspective. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pp. 13699–13708, 2022.

Song, H., Kim, M., Park, D., Shin, Y., and Lee, J.-G. Learning from noisy labels with deep neural networks: A survey. *IEEE transactions on neural networks and learning systems*, 34(11):8135–8153, 2022.

Tian, Y. Provable scaling laws of feature emergence from learning dynamics of grokking. *arXiv preprint arXiv:2509.21519*, 2025.

Tirumala, K., Markosyan, A., Zettlemoyer, L., and Aghajanyan, A. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35: 38274–38290, 2022.

Umar, A. H., Tezoh, F. K. N., Barbier, J., Acevedo, S., and Laio, A. The effect of label noise on the information content of neural representations. *arXiv preprint arXiv:2510.06401*, 2025.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Xu, Z., Wang, Y., Frei, S., Vardi, G., and Hu, W. Benign overfitting and grokking in relu networks for xor cluster data. *arXiv preprint arXiv:2310.02541*, 2023.

Yang, Z., Yu, Y., You, C., Steinhardt, J., and Ma, Y. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pp. 10767–10777. PMLR, 2020.

Zhong, Z., Liu, Z., Tegmark, M., and Andreas, J. The clock and the pizza: Two stories in mechanistic explanation of neural networks. *Advances in neural information processing systems*, 36:27223–27250, 2023.

A. Related Work

Model-wise Double Descent and Over-parameterization. The discovery of the double descent phenomenon (illustrated in Figure 1) marked a significant shift in modern machine learning, suggesting that the classical bias–variance trade-off is incomplete (Belkin et al., 2019; Yang et al., 2020). In the over-parameterized regime, test error can decrease again after the interpolation threshold, leading to superior generalization. Theoretically, this second descent has been rigorously characterized in linear regression under various assumptions (Belkin et al., 2020; Advani et al., 2020; Adlam & Pennington, 2020; Bartlett et al., 2020; Mei & Montanari, 2022; Hastie et al., 2022; Muthukumar et al., 2020), although some studies indicate that appropriate regularization can mitigate or eliminate this effect (Nakkiran et al., 2020). In non-linear settings, double descent has been robustly observed in computer vision tasks, particularly in the presence of label noise (Nakkiran et al., 2021; Umar et al., 2025).

Recent efforts have sought mechanistic explanations for the generalization power of large-scale models. One perspective proposes that over-parameterized models generalize through smoother interpolation of noisy data points (Gamba et al., 2022; Somepalli et al., 2022). Furthermore, the emergence of neural scaling laws has been increasingly attributed to the superposition of discrete features or representations (Liu et al., 2025). Davies et al. (2023); Huang et al. (2024) further unify double descent with grokking by framing both phenomena as a competition between memorization and generalization circuits. Our work extends this line of research by investigating how model size specifically governs the resolution of this competition under label noise.

Grokking and Emergent Generalization. In over-parameterized models, generalization often emerges long after the training loss has vanished, a phenomenon known as grokking (Power et al., 2022). This decoupling between fitting and rule discovery is prominently observed in modular arithmetic tasks across various architectures (Liu et al., 2022b; Nanda et al., 2023; Mallinar et al., 2025). While specific algorithms, such as Pizza and Clock, have been identified for Transformers solving these tasks (Nanda et al., 2023; Zhong et al., 2023), and a universal abstract algorithm for deep neural networks has been uncovered (McCracken et al., 2025), the exact analytical solution for two-layer ReLU networks remains unsolved. Existing theoretical works have explored provable solutions using quadratic activations or specific regularization schemes (Gromov, 2023; Tian, 2025), yet the role and impact of label noise in these settings remain largely unexplored.

Learning under Label Noise. Most existing work on learning with label noise focuses on robust training strategies to prevent overfitting (Li et al., 2020; Liu et al., 2022a). We refer readers to Song et al. (2022) for a comprehensive survey. Xu et al. (2023) theoretically analyze the performance of two-layer ReLU networks on noisy XOR data under the assumption of fixed second-layer weights, which deviates from standard end-to-end training dynamics. Doshi et al. (2024) examine the effects of noise ratios and different regularization methods on arithmetic tasks. In contrast, we primarily focus on the relationship between model size and generalization performance, and further investigate the mechanistic entanglement between memorization and generalization.

B. Analytical Solution Under Quadratic Activation

The fact that the solution in Equations (2) is exact for quadratic activations has been well discussed in Pearce et al. (2023); Morwani et al. (2024). We present their theory here, adapted to our notation, in Proposition B.1.

Proposition B.1. *(Theorem 7, Morwani et al. (2024)) For the modular addition task, any two-layer network $f_\theta = \boldsymbol{W}\left(\boldsymbol{U}\boldsymbol{e}_a + \boldsymbol{V}\boldsymbol{e}_b\right)^2$ with $M \geq 4(P-1)$ that maximizes the margin on $\mathcal{D}_{\text{total}}$ satisfies the following conditions:*

1. for each neuron $\{\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m\}$, there exists a scaling constant $\lambda \in \mathbb{R}$ and a frequency $\omega \in \left\{1, \cdots, \frac{P-1}{2}\right\}$, such that

$$u_{mi} = \lambda \cos\left(\frac{2\pi}{P}\omega_m i + \varphi_m^{(a)}\right), \tag{6a}$$

$$v_{mj} = \lambda \cos\left(\frac{2\pi}{P}\omega_m j + \varphi_m^{(b)}\right), \tag{6b}$$

$$w_{mk} = \lambda \cos\left(\frac{2\pi}{P}\omega_m k + \varphi_m^{(c)}\right), \tag{6c}$$

for some phase offsets $\varphi_m^{(a)}, \varphi_m^{(b)}, \varphi_m^{(c)} \in \mathbb{R}$ satisfying $\varphi_m^{(a)} + \varphi_m^{(b)} = \varphi_m^{(c)}$.

2. For every frequency $\omega \in \left\{1, \cdots, \frac{P-1}{2}\right\}$, at least one neuron in the network uses this frequency.

C. Supplementary Experiments for Section 3

C.1. Double-descent curves on varying setups

Model-wise double descent is clearly observed across different activation functions, training ratios, optimizers, and arithmetic tasks. We summarize the Figures about it in Table 1.

Table 1. Figure indices showing double descent curves under varying setups.

Activation Functions	Training Ratios	Optimizers	Arithmetic Tasks
2, 13	14 (40%), 3(50%), 15(60%)	3 (AdamW), 16 (Adam), 17 (Muon)	18

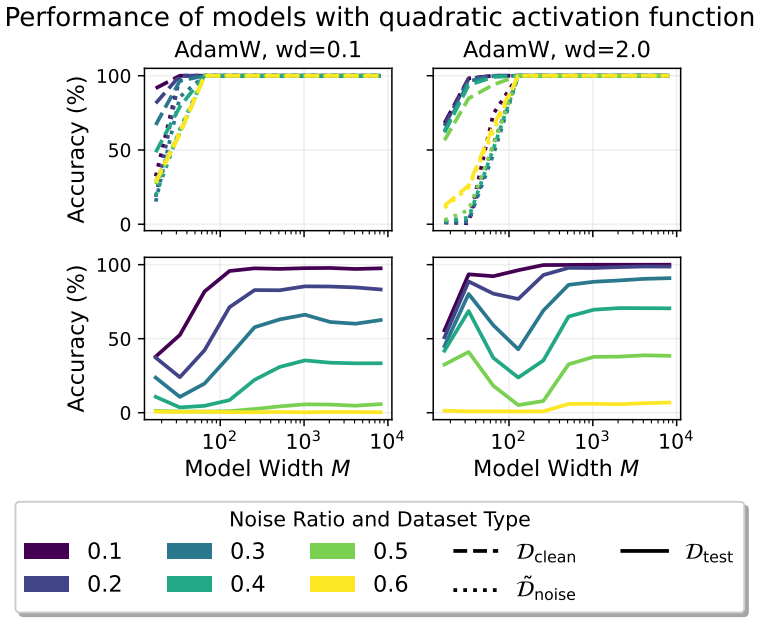


Figure 13. Ceilings of test accuracy for quadratic activation functions exist across varying weight decays.

SGD is not a good choice. Figure 19 shows the training dynamics of SGD under different choices of learning rate and weight decay. Among all configurations, only the setting with $\text{lr} = 0.1$ and $\text{wd} = 0.001$ exhibits noticeable generalization behavior. This suggests that SGD requires highly restrictive hyperparameter tuning to achieve reasonable generalization. Moreover, even at epoch 200,000, the test accuracy remains relatively low and is still increasing, indicating that SGD converges extremely slowly. Overall, SGD is inefficient for this task, as it consumes an excessive number of training epochs to reach a modest level of generalization.

C.2. Discussion on Model Misspecification

We consider a scenario where the model architecture is not expressive enough to memorize all noisy labels, which leads to atypical generalization behavior.

Modular addition and multiplication are commutative operations, i.e., $a \circ b = b \circ a$. Accordingly, an alternative architecture enforces a tied first layer by setting $U = V$ (Pearce et al., 2023). Under random label noise, however, the noisy samples generally violate this symmetry. As a result, the tied model is unable to interpolate all noisy labels, and the coexistence phase does not emerge in this setting, leading to irregular test accuracy curves (see the left panel in Figure 20).

To verify that the asymmetric structure of random noise is responsible for this behavior, we construct a symmetric noise pattern and re-evaluate the tied model. Specifically, for a noise ratio α in modular addition, we randomly sample $\frac{\alpha}{2}|\mathcal{D}_{\text{train}}|$ examples to form $\mathcal{D}_{\text{asym}}$, where each $(a, b, c) \in \mathcal{D}_{\text{asym}}$ satisfies $a < b$. The noisy dataset is then constructed as

$$\tilde{\mathcal{D}}_{\text{noise}} = \{(a, b, \tilde{c}) \mid (a, b, c) \in \mathcal{D}_{\text{asym}} \text{ or } (b, a, c) \in \mathcal{D}_{\text{asym}},\ \tilde{c} \neq c,\ \tilde{c} \in \{0, \ldots, P-1\}\}.$$

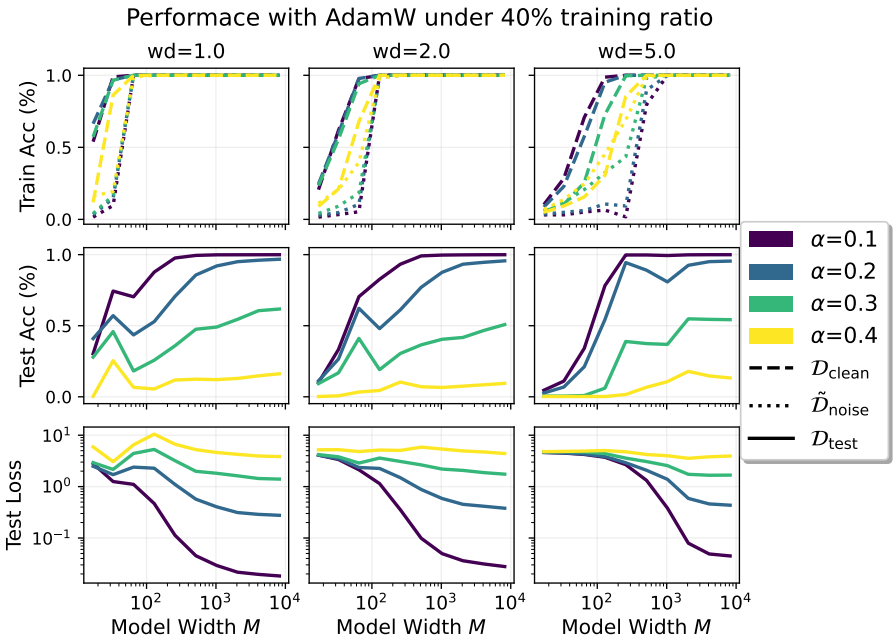


Figure 14. Performance of AdamW under 40% training ratio.

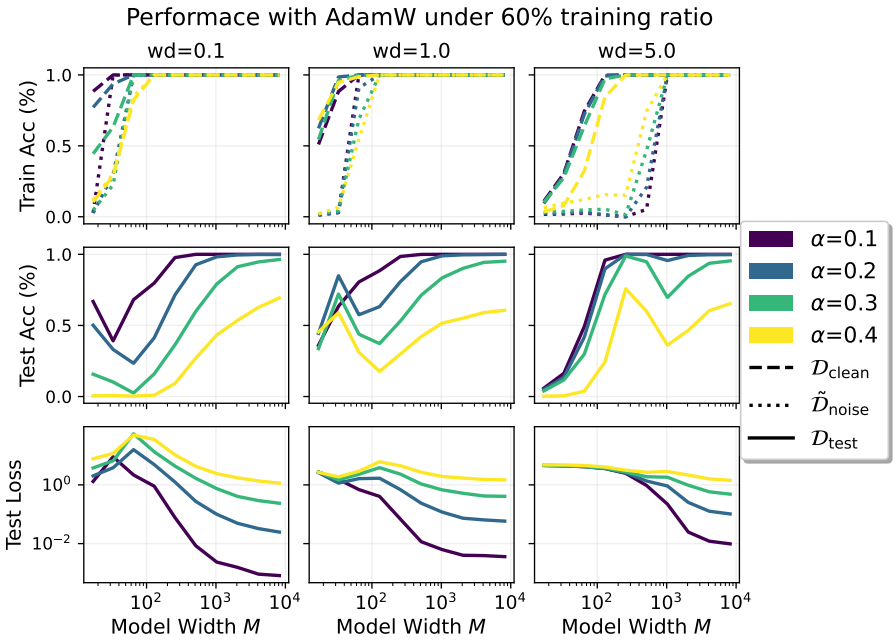


Figure 15. Performance of AdamW under 60% training ratio.

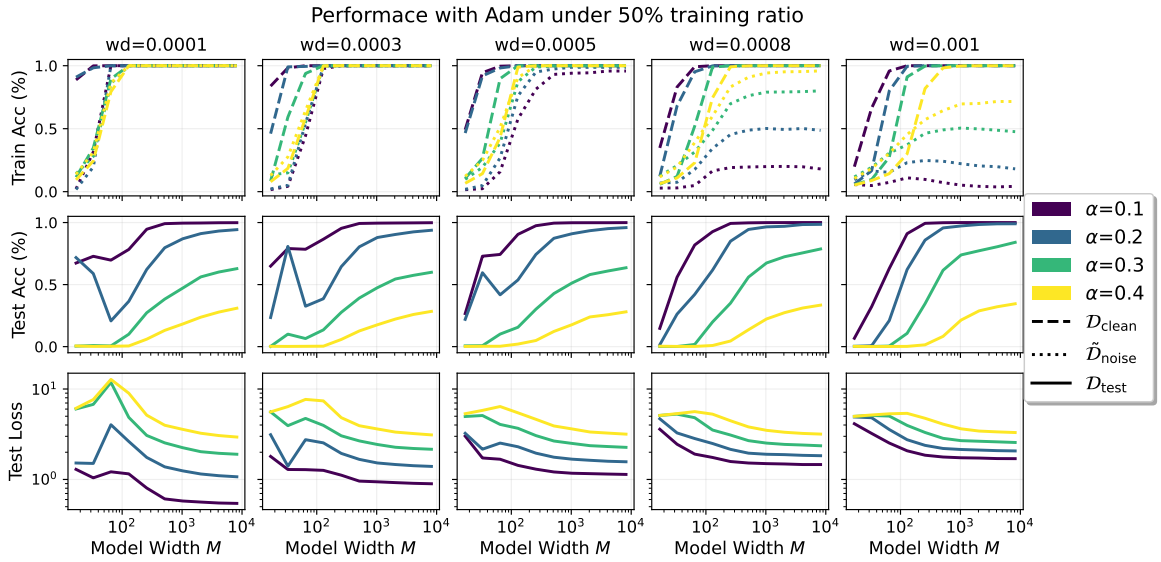


Figure 16. Performance of Adam under varying weight decays.

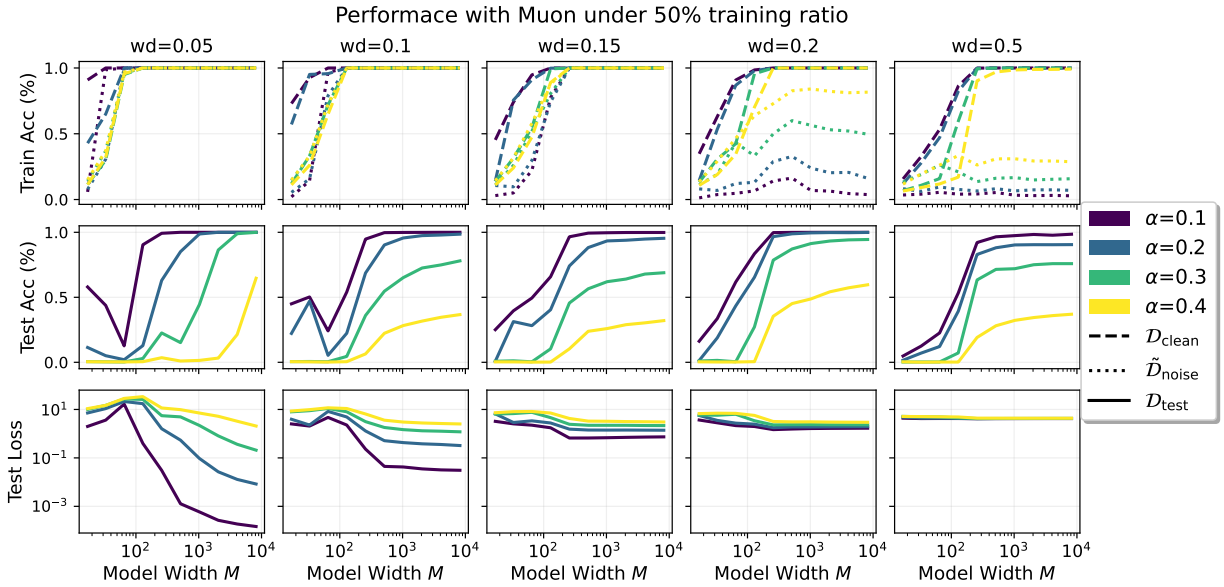


Figure 17. Performance of Muon under varying weight decays.

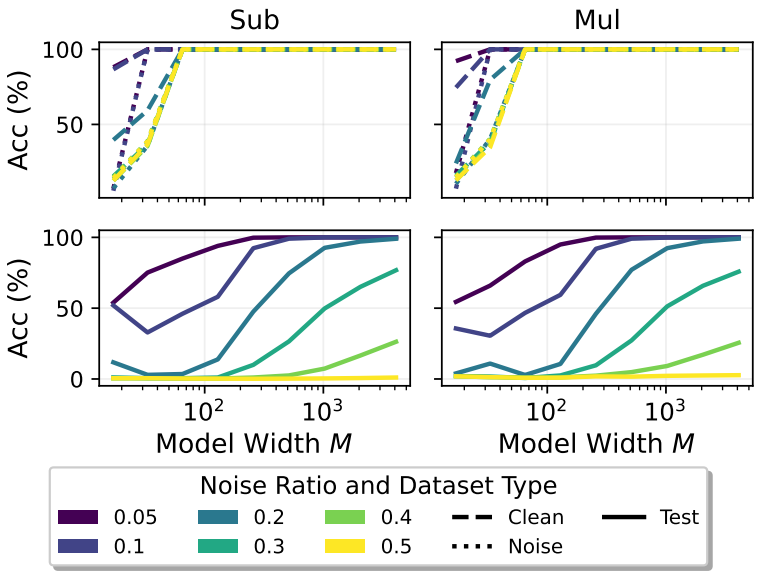


Figure 18. Performance on modular subtraction (*left*) and multiplication (*right*) tasks. Models are trained using AdamW under 50% training ratio.

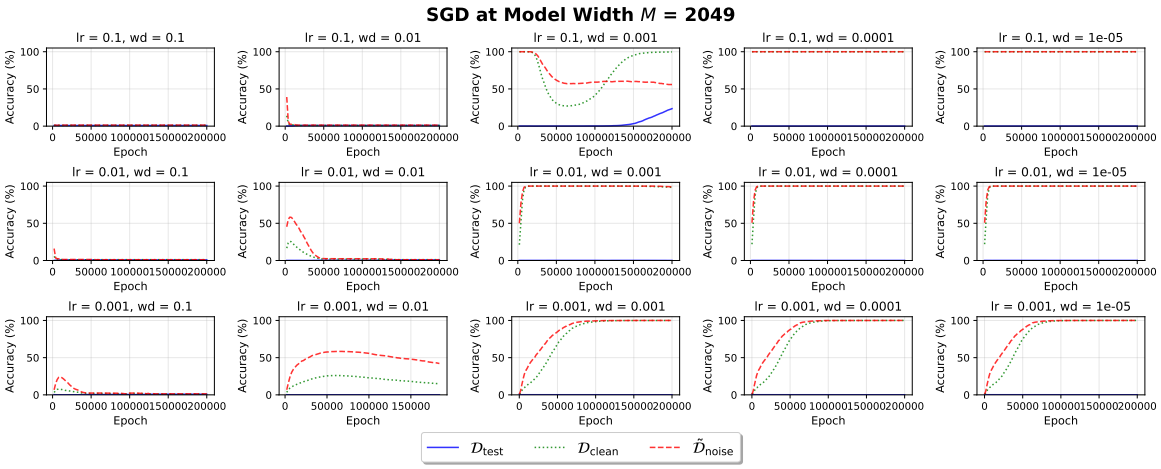


Figure 19. Training dynamics of SGD under different learning rates (lr) and weight decay (wd) values at $M = 2049$ and $\alpha = 0.3$. Results are reported every 2,000 epochs, where the total epochs are 200,000.

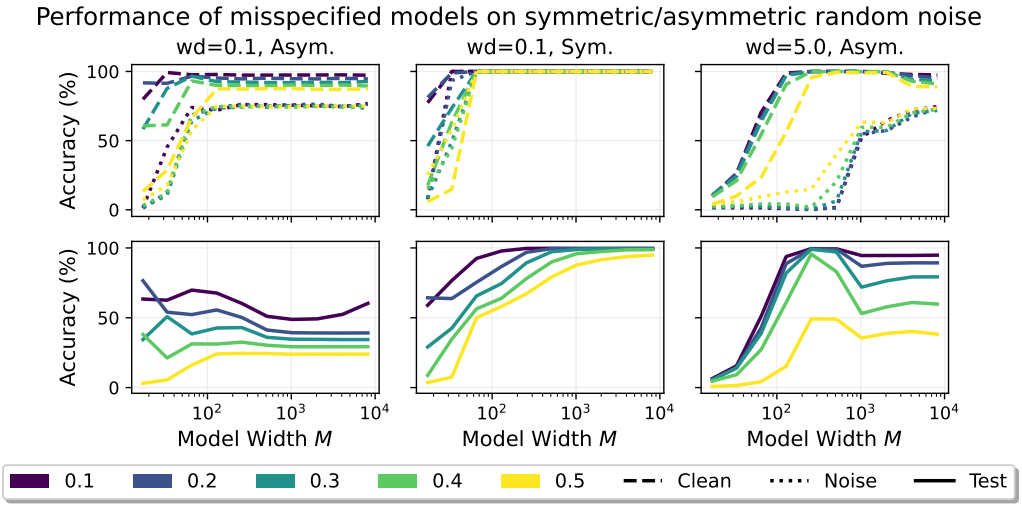


Figure 20. Performance of first-layer tied models under model misspecification. *(Left)* With asymmetric random noise, the tied model fails to fully interpolate noisy labels and exhibits irregular generalization behavior. *(Middle)* Under symmetric noise, the tied model successfully interpolates all noisy labels, and generalization improves with increasing model size. *(Right)* Stronger regularization mitigates the adverse effect of misspecification by encouraging the model to ignore noisy labels.

Under this symmetric noise, the tied model exhibits predictable behavior: noisy labels can be fully memorized once the model width exceeds a threshold (i.e., the coexistence phase appears), and generalization improves with increasing model size (see the middle panel in Figure 20).

When the noise structure is unknown, model misspecification becomes a potential risk. One practical mitigation strategy is to strengthen regularization so as to extend the inversion phase. The tied model is able to interpolate all clean samples, and increasing weight decay encourages the model to suppress/neglect noisy labels, thereby recovering good generalization performance (see the right panel in Figure 20).

C.3. Discussion on Result 3.2

We provide a gradient-based explanation for the counter-intuitive observation in Result 3.2. Let $\mathcal{D}_{\mathrm{c}} = \mathcal{D}_{\text{clean}}$ and $\mathcal{D}_{\mathrm{n}} = \tilde{\mathcal{D}}_{\text{noise}}$ denote the clean and noisy subsets of the training data, respectively. For a model parameterized by θ, we decompose the empirical loss as

$$L(\theta) = L_{\mathrm{c}}(\theta) + L_{\mathrm{n}}(\theta), \qquad L_{\mathrm{c}}(\theta) = \sum_{z \in \mathcal{D}_{\mathrm{c}}} \ell(z;\theta), \qquad L_{\mathrm{n}}(\theta) = \sum_{z \in \mathcal{D}_{\mathrm{n}}} \ell(z;\theta),$$

where ℓ is the cross-entropy loss on a single example. Under full-batch gradient descent, the update direction is

$$-\nabla L(\theta) = -\nabla L_{\mathrm{c}}(\theta) - \nabla L_{\mathrm{n}}(\theta).$$

For a sufficiently small step size η, the first-order decrease of the clean and noisy losses after one update is approximately

$$\Delta L_{\mathrm{c}} \approx -\eta \left\langle \nabla L_{\mathrm{c}}, \nabla L_{\mathrm{c}} + \nabla L_{\mathrm{n}} \right\rangle, \qquad \Delta L_{\mathrm{n}} \approx -\eta \left\langle \nabla L_{\mathrm{n}}, \nabla L_{\mathrm{c}} + \nabla L_{\mathrm{n}} \right\rangle.$$

Thus, the instantaneous learning speed of each subset is controlled by how well its own gradient aligns with the full-batch training direction. To compare the two subsets fairly, we normalize by their sizes and define the per-sample loss decrease rates

$$s_{\mathrm{c}} = \frac{\langle \nabla L_{\mathrm{c}}, \nabla L_{\mathrm{c}} + \nabla L_{\mathrm{n}} \rangle}{|\mathcal{D}_{\mathrm{c}}|}, \qquad s_{\mathrm{n}} = \frac{\langle \nabla L_{\mathrm{n}}, \nabla L_{\mathrm{c}} + \nabla L_{\mathrm{n}} \rangle}{|\mathcal{D}_{\mathrm{n}}|}.$$

Empirically, at the beginning of training, we observe that $\langle \nabla L_{\mathrm{c}}, \nabla L_{\mathrm{n}} \rangle < 0$, meaning that the clean and noisy subsets induce partially conflicting gradient directions. Nevertheless, both normalized progress measures remain positive, i.e., $s_{\mathrm{c}} > 0$ and $s_{\mathrm{n}} > 0$. This indicates that the full-batch update still decreases both the clean and noisy losses at the very beginning of

Table 2. Early-stage clean and noisy accuracy on MNIST with a three-layer neural network under full-batch training.

	$\alpha = 0.05$		$\alpha = 0.2$	
Period	Noisy acc.	Clean acc.	Noisy acc.	Clean acc.
Epoch 1	10.87%	2.86%	10.75%	2.89%
Epoch 2	6.47%	46.29%	5.71%	48.09%

training, although it does so at different per-sample rates. However, since the noisy subset is much smaller when the noise ratio α is small, the update direction can produce a larger *per-sample* decrease on the noisy subset. In other words, although the noisy labels are unstructured, each noisy example receives a stronger individual push from the full-batch gradient. This explains why the noisy-label accuracy can increase earlier than the clean-label accuracy, especially at lower noise ratios.

This gradient-based explanation is also consistent with additional experiments beyond modular arithmetic. To test whether the early noisy-over-clean learning order is purely induced by the algebraic structure of modular arithmetic, we further train a three-layer neural network on MNIST with randomly corrupted labels(hidden dimensions= $[1024, 256]$, using AdamW with learning rate 0.001). As shown in Table 2, under full-batch training, noisy-label accuracy is higher than clean-label accuracy at epoch 1 for both $\alpha = 0.05$ and $\alpha = 0.2$. In the next epoch, clean accuracy quickly catches up and becomes much larger, while noisy accuracy drops. These results suggest that the early advantage of noisy labels is not solely a consequence of the Fourier structure of modular arithmetic, but can also arise in a standard image classification task under a similar optimization protocol.

The MNIST experiment also helps clarify the scope of the phenomenon. The effect is most visible at the very beginning of full-batch training and should not be interpreted as saying that noisy labels are ultimately easier to learn than clean labels. Instead, the first few updates may create a larger per-sample progress on the smaller noisy subset, while the clean structure dominates soon afterward. This is consistent with our gradient analysis: the initial quantities s_{c} and s_{n} can both be positive, with $s_{\mathrm{n}} > s_{\mathrm{c}}$, even though the clean and noisy gradients partially conflict.

This explanation also clarifies why the phenomenon depends on the training protocol. Under full-batch training, the gradient contribution of the noisy subset is accumulated coherently in every update, so the direction that benefits the current noisy labels is repeatedly applied. In contrast, under small-batch training, different mini-batches contain different noisy examples, and the gradient direction that helps memorize the noisy labels in one batch is generally not transferable to another batch. As a result, the per-sample push toward any specific corrupted label becomes less stable, and the early noisy-over-clean learning order is weakened or may disappear.

C.4. Accuracy learning curves

The accuracy learning curves capture when memorization and generalization emerge during training. We observe the grokking phenomenon even under noisy labels: the model quickly reaches 100% training accuracy, while test accuracy improves significantly only later. To illustrate this, we separately present the training dynamics on short and long time scales, highlighting memorization and generalization, respectively.

C.4.1. Memorization on a short time scale

Without noise in the training dataset, previous studies have found that larger models memorize the training data faster (Tirumala et al., 2022; Huang et al., 2024). This trend persists under noisy training data, although memorization speeds differ for clean and noisy subsets. We find that the model memorizes noisy labels faster than clean labels, and that the gap is larger at lower noise ratios. These phenomena are consistently observed across several optimizers, including Adam (Figure 21), AdamW (Figure 22), and Muon (Figure 23).

C.4.2. Generalization on a long time scale

In terms of test accuracy, we observe a general tendency that models under larger weight decay grok faster, although the trend is not equally pronounced across all hyperparameter settings. This trend is clear for AdamW (Figure 24) and Muon (Figure 26), but not evident for Adam (Figure 25). Across all of them, models under smaller noise ratio grok faster.

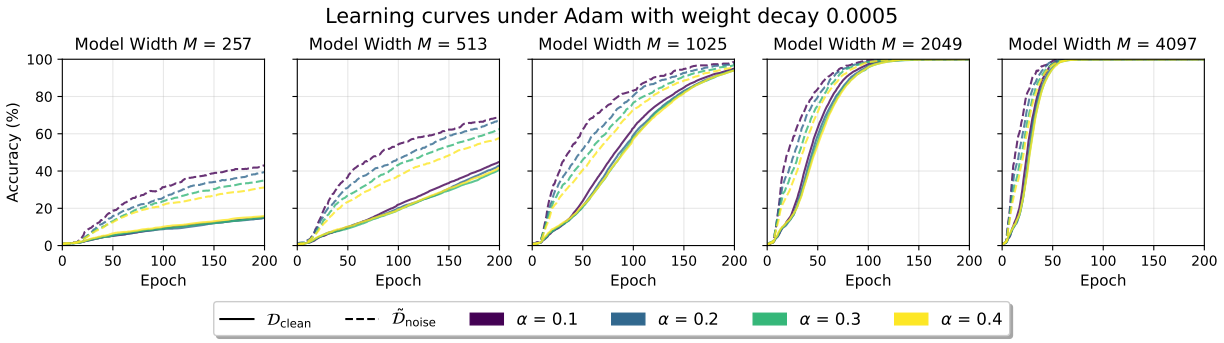


Figure 21. Memorization curves for the Adam optimizer. Larger models reach high training accuracy more quickly. Noisy labels are memorized faster than clean labels, and lower noise ratios further accelerate this effect.

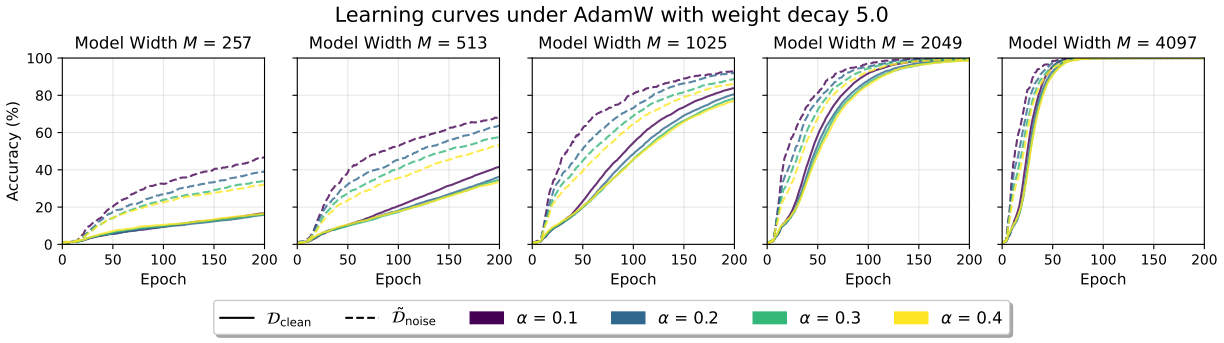


Figure 22. Memorization curves for AdamW optimizer.

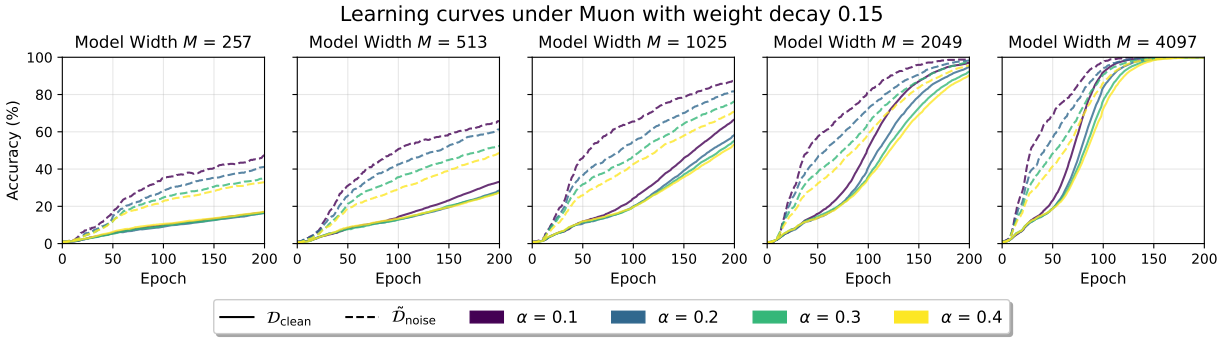


Figure 23. Memorization curves for Muon optimizer.

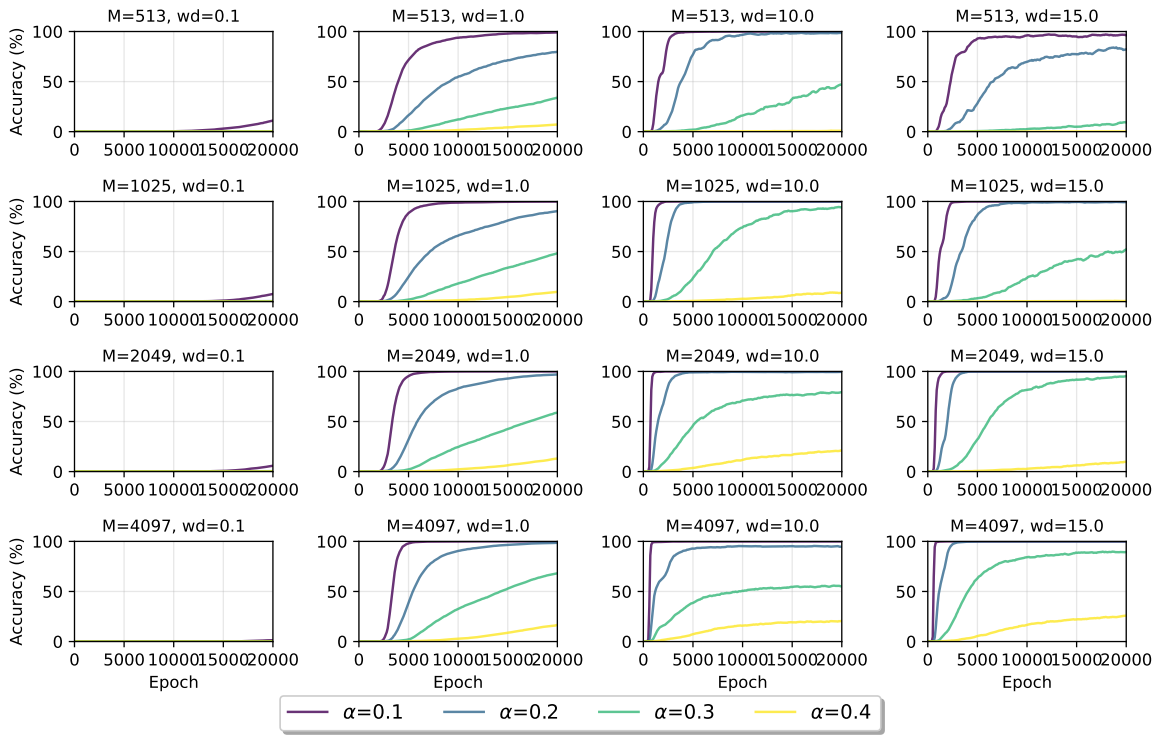


Figure 24. Training dynamics of test accuracy across various weight decays for AdamW. Small weight decay (wd=0.1) leads to delayed and slow grokking.

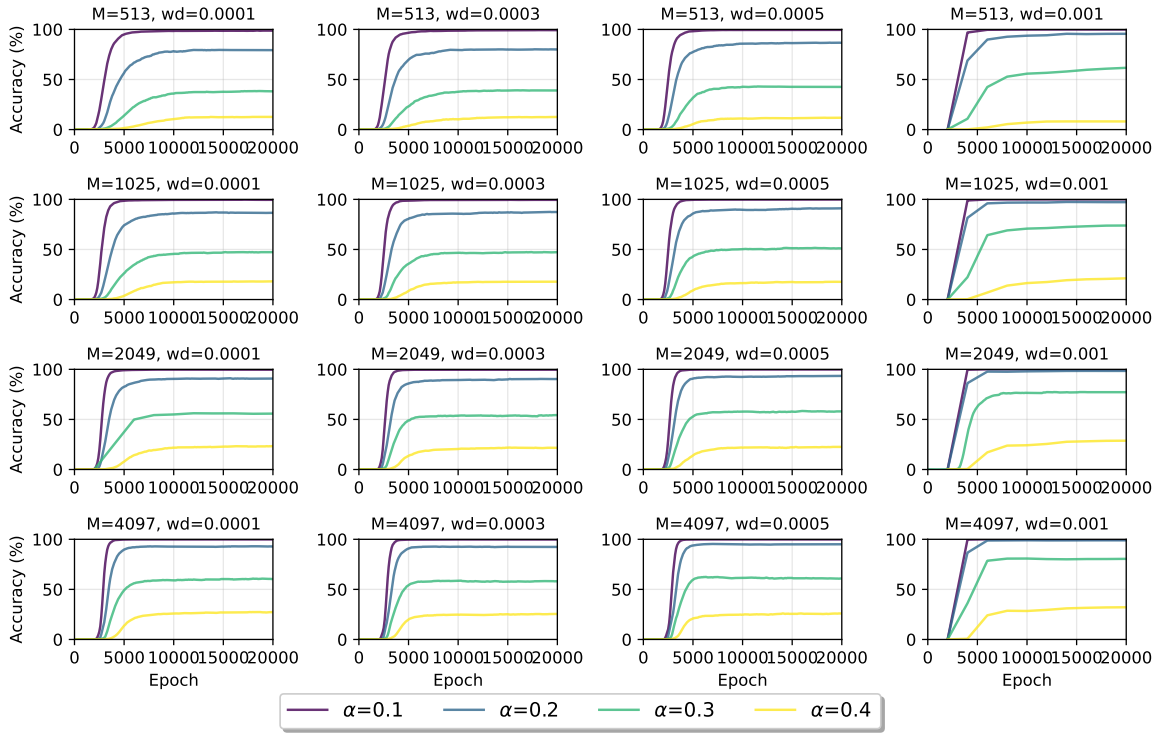


Figure 25. Training dynamics of test accuracy across various weight decays for Adam.

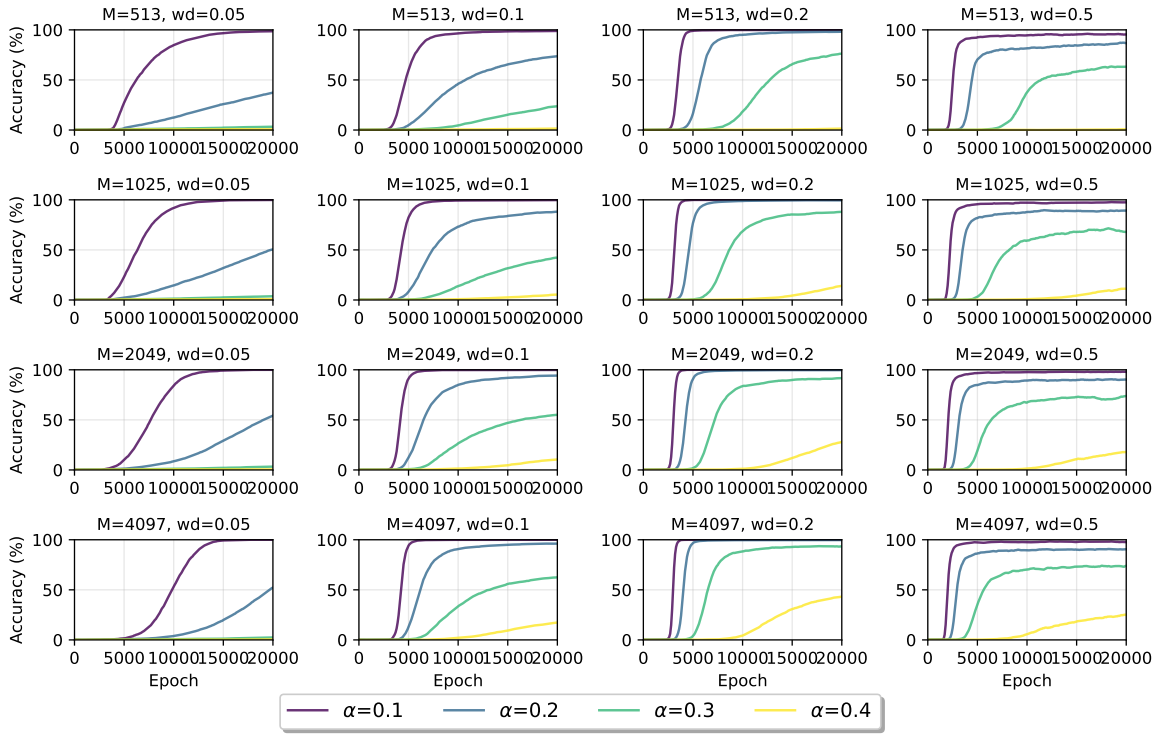


Figure 26. Training dynamics of test accuracy across various weight decays for Muon.

D. Supplementary Materials for Section 4

D.1. Details of frequency filtration

In this section, we provide more details about the frequency filtration. That is to say, the details about decomposing $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$, the mth neuron, into $\left(\boldsymbol{u}_m^G, \boldsymbol{v}_m^G, \boldsymbol{w}_m^G\right)$ and $\left(\boldsymbol{u}_m^R, \boldsymbol{v}_m^R, \boldsymbol{w}_m^R\right)$.

Let $\omega = e^{-2\pi i/P}$. Define the Fourier transform matrix $F \in \mathbb{C}^{P\times P}$ by

$$F_{k,n} = \frac{1}{\sqrt{P}}\omega^{kn} = \frac{1}{\sqrt{P}}e^{-2\pi ikn/P}.$$

For any vector $\boldsymbol{x} \in \mathbb{R}^{P\times 1}$, denote its Fourier transform as $\tilde{\boldsymbol{x}} = \boldsymbol{F}\boldsymbol{x}$. $\boldsymbol{F}$ is a unitary matrix, so its conjugate transpose of $\boldsymbol{F}$, denoted as $\boldsymbol{F}^H$, gives $\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{v} = \boldsymbol{v}$. Without loss of generality, we assume $\boldsymbol{x}$ is a vector satisfying that $\arg\max_k |\tilde{x}_k|$ is a singleton.

We define the main frequency retrieval operator (for generalization part) as $\boldsymbol{F}^H\boldsymbol{F}_G(\boldsymbol{x})$, filtering the frequency with maximum magnitude, namely,

$$F_G(\boldsymbol{x})_{k,n} = \begin{cases} F_{k,n}, & k \in \arg\max_{k'} |\tilde{x}_{k'}| \\ 0, & k \notin \arg\max_{k'} |\tilde{x}_{k'}| \end{cases},$$

and the remaining frequency retrieval operator (for memorization) as $\boldsymbol{F}^H\boldsymbol{F}_R(\boldsymbol{x})$, where $\boldsymbol{F}_R(\boldsymbol{x}) = \boldsymbol{F} - \boldsymbol{F}_G(\boldsymbol{x})$.

For the mth neuron, let $\boldsymbol{P}_{G,m} = \boldsymbol{F}^H\boldsymbol{F}_G(\boldsymbol{w}_m)$ and $\boldsymbol{P}_{R,m} = \boldsymbol{F}^H\boldsymbol{F}_R(\boldsymbol{w}_m)$. Then, we define the decomposition to neuron m as

$$\begin{aligned} \boldsymbol{u}_m^G &= \text{Real}(\boldsymbol{P}_{G,m}\boldsymbol{u}_m),\ \boldsymbol{u}_m^R = \text{Real}(\boldsymbol{P}_{R,m}\boldsymbol{u}_m), \\ \boldsymbol{v}_m^G &= \text{Real}(\boldsymbol{P}_{G,m}\boldsymbol{v}_m),\ \boldsymbol{v}_m^R = \text{Real}(\boldsymbol{P}_{R,m}\boldsymbol{v}_m), \\ \boldsymbol{w}_m^G &= \text{Real}(\boldsymbol{P}_{G,m}\boldsymbol{w}_m),\ \boldsymbol{w}_m^R = \text{Real}(\boldsymbol{P}_{R,m}\boldsymbol{w}_m). \end{aligned}$$

It is easy to check $\boldsymbol{P}_{G,m} + \boldsymbol{P}_{R,m} = \boldsymbol{I}$, so

$$(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m) = \left(\boldsymbol{u}_m^G, \boldsymbol{v}_m^G, \boldsymbol{w}_m^G\right) + \left(\boldsymbol{u}_m^R, \boldsymbol{v}_m^R, \boldsymbol{w}_m^R\right).$$

We provide the visualization of a neuron under such frequency filtration in Figure 27.

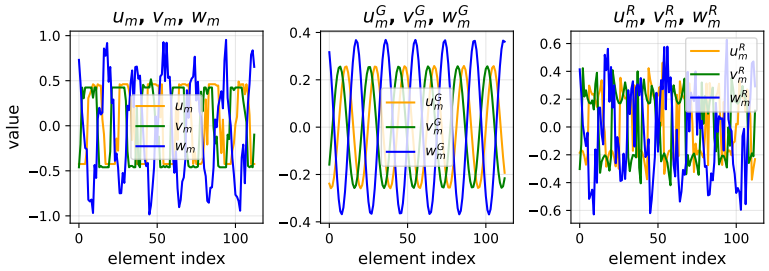


Figure 27. Visualization of applying frequency filtration to a neuron. $(\boldsymbol{u}_m^G, \boldsymbol{v}_m^G, \boldsymbol{w}_m^G)$ retains the frequency component with the maximum magnitude in $(\boldsymbol{u}_m, \boldsymbol{v}_m, \boldsymbol{w}_m)$.

D.2. Frequency filtration benefits generalization under varying setups

Frequency filtration also disentangles generalization and memorization for the modular *subtraction* task (Figure 28) and for models with *quadratic* activations (Figure 29). We observe substantial improvement in generalization after frequency filtration: ReLU models recover test accuracy under severe label noise, while quadratic models exceed previously observed accuracy limits. Test accuracy also increases for models trained on noise-free data (Figure 30).

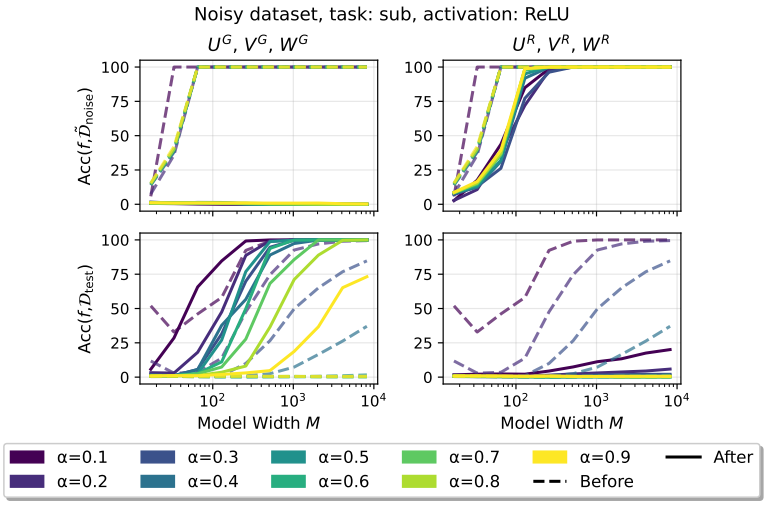


Figure 28. Performance before (dashed) and after (solid) the frequency filtration on the modular *subtraction* task.

Frequency filtration on transformers. To test whether the frequency-based filtering idea is restricted to two-layer networks, we also apply FF to a Transformer trained on modular addition with label noise. The mechanism of the Transformer for the modular addition task on *clean* data has been explored in Nanda et al. (2023). Specifically, after training, we keep only the top 20 frequencies in the unembedding head and remove the remaining frequency components. This simple intervention substantially reduces the model's ability to memorize noisy labels while improving its test accuracy, suggesting that the frequency-structured rule representation is not exclusive to the two-layer setting. (Table 3) These results provide preliminary evidence that FF can extend beyond two-layer networks.

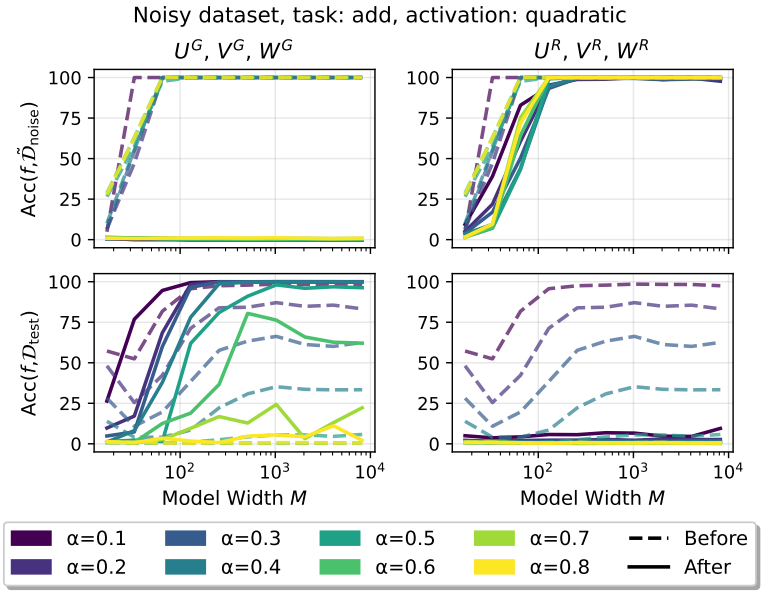


Figure 29. Performance before (dashed) and after (solid) the frequency filtration on models with *quadratic* activation. We remark that frequency filtration also breaks the ceilings of test accuracy.

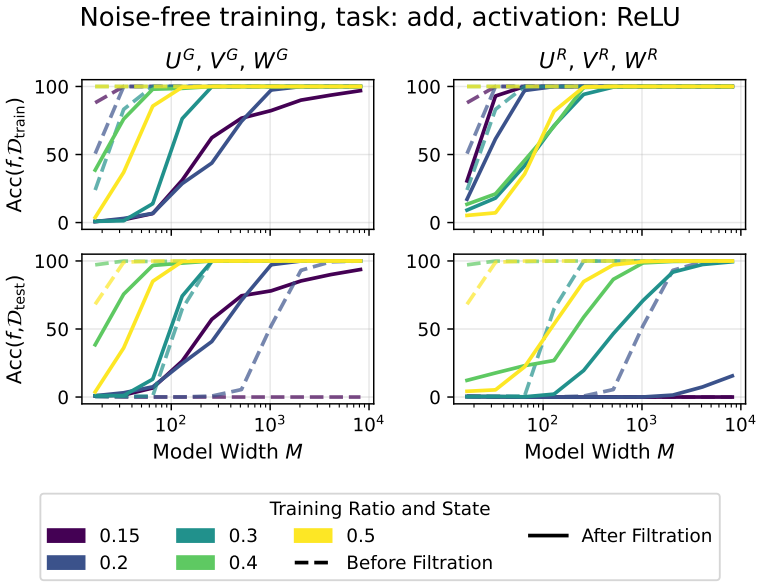


Figure 30. Performance before (dashed) and after (solid) frequency filtration for models trained on the clean modular addition data. The models use ReLU activation and are trained with AdamW. Overall, test accuracy after filtration (U^G, V^G, W^G) is generally higher, particularly for large models.

Table 3. Applying frequency filtration to a Transformer trained on modular addition. FF keeps only the top 20 frequencies in the unembedding head.

Method	Clean Acc.	Noisy Acc.	Test Acc.
Before FF	100%	100%	76.8%
After FF	98.1%	36.8%	87.8%

D.3. Relationship among the phases $\varphi_m^{(a)}$, $\varphi_m^{(b)}$, and $\varphi_m^{(c)}$

Suppose the dominant frequency of neuron m has the form

$$u_{mi}^G = \lambda_u \cos\left(\frac{2\pi}{P}\omega_m i + \varphi_{m,G}^{(a)}\right), \tag{7a}$$

$$v_{mj}^G = \lambda_v \cos\left(\frac{2\pi}{P}\omega_m j + \varphi_{m,G}^{(b)}\right), \tag{7b}$$

$$w_{mk}^G = \lambda_w \cos\left(\frac{2\pi}{P}\omega_m k + \varphi_{m,G}^{(c)}\right). \tag{7c}$$

For models trained with either ReLU or quadratic activations, there exists $q \in \mathbb{Z}$ such that the following equations are approximately satisfied:

$$\varphi_{m,G}^{(a)} + \varphi_{m,G}^{(b)} = \varphi_{m,G}^{(c)} + 2q\pi \qquad \text{(modular addition)}, \tag{8}$$

$$\varphi_{m,G}^{(a)} - \varphi_{m,G}^{(b)} = \varphi_{m,G}^{(c)} + 2q\pi \qquad \text{(modular subtraction)}. \tag{9}$$

We quantify the empirical deviation of these relationships using the mean squared error (MSE). Specifically, for modular addition:

$$\text{Phase MSE} = \frac{1}{M}\sum_{m=1}^{M} \min_{q\in\mathbb{Z}} \left(\varphi_{m,G}^{(a)} + \varphi_{m,G}^{(b)} - \varphi_{m,G}^{(c)} + 2q\pi\right)^2,$$

and for modular subtraction, the $+$ is replaced by $-$ in the definition above.

We visualize these relationships in Figures 31 (addition) and 32 (subtraction), and the Phase MSE trends with respect to model size and noise ratio are shown in Figure 33. We observe that higher noise ratios generally lead to larger Phase MSE, i.e., larger deviation to the analytical relationship.

D.4. Uniform distribution of the frequency

Figure 34 shows that each frequency is covered by at least one neuron after frequency filtration (only dominant frequency left for each neuron). Figure 35 also shows that the distribution of the frequency magnitude after filtration is close to uniform at larger model width. Here, let $\tilde{\boldsymbol{W}}^G$ is the Fourier transform of $\boldsymbol{W}^G$ with each column constituted by $\{\boldsymbol{w}_m\}_{m=1}^M$, so Figure 35 plots the norm of each row vector in $\tilde{\boldsymbol{W}}^G$ with index in $\left\{1, \cdots, \frac{P-1}{2}+1\right\}$.

E. Supplementary Materials for Section 5

E.1. Performance of Algorithm 1 using Str. or IPR

In this section, we examine generalization improvements across different tasks achieved through neuron selection. Figure 36 reports the improvement obtained using IPR for neuron selection, while Figure 37 presents the corresponding results for Str.

Comparing the results on the modular multiplication task illustrates the generality of the method using Str. This task lacks periodic patterns relevant for generalization, so measuring neuron importance via IPR offers limited improvement.

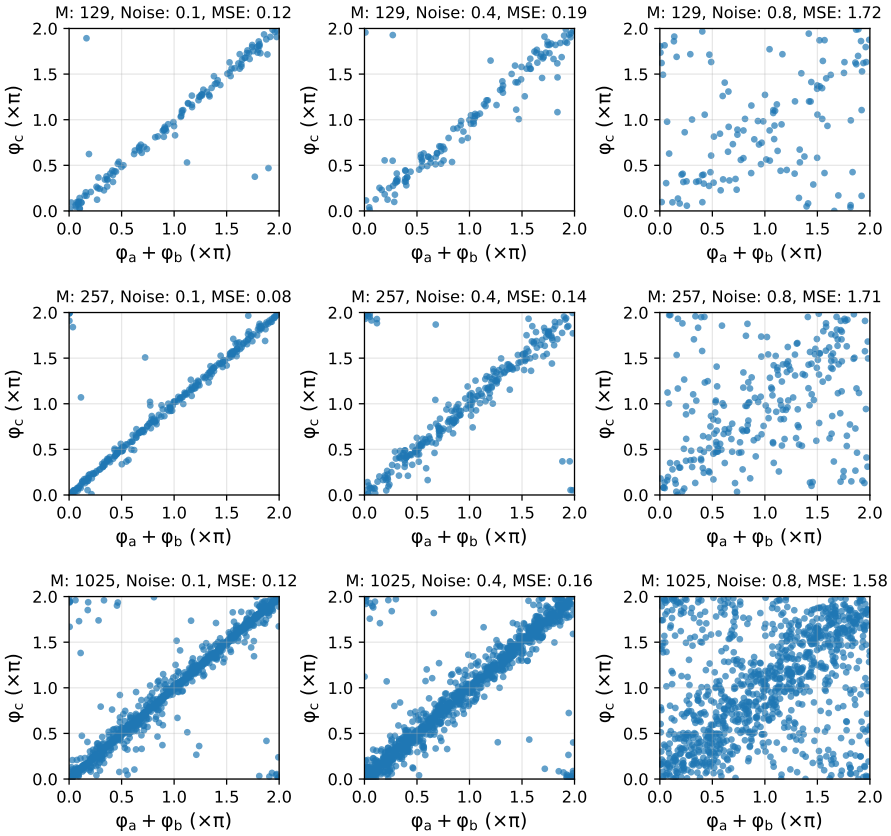


Figure 31. Scatter plot showing the relationship of phases on modular addition tasks. Each point represents a neuron, where φ_a, φ_b, and φ_c are the phases of $\boldsymbol{u}^G$, $\boldsymbol{v}^G$, and $\boldsymbol{w}^G$, respectively.

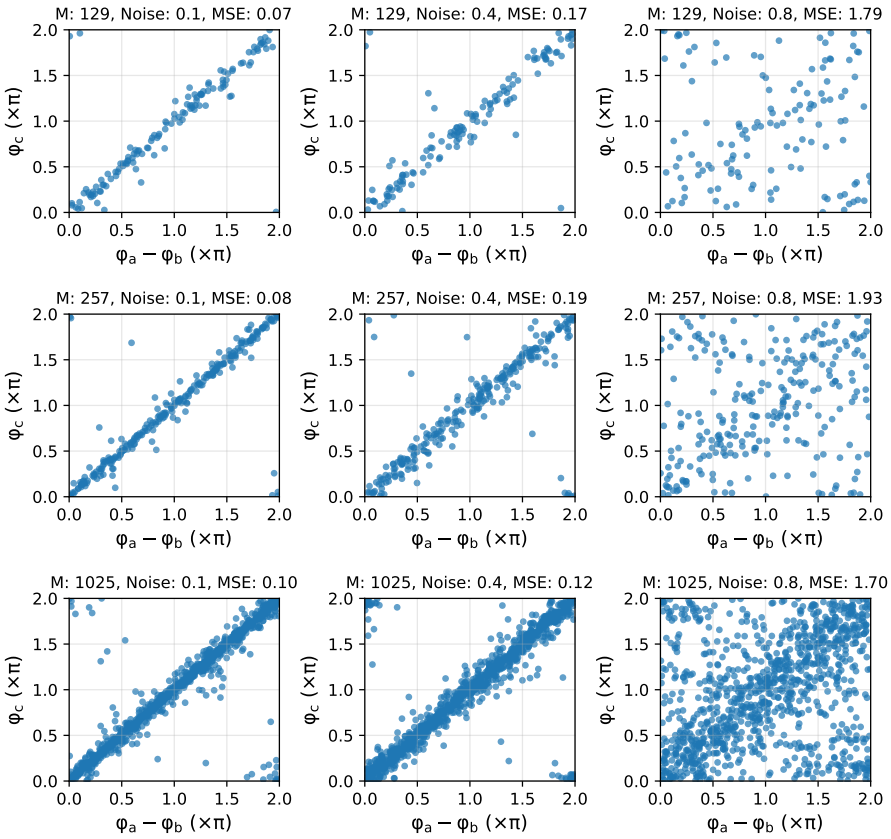


Figure 32. Scatter plot showing the relationship of phases on modular subtraction tasks. Each point represents a neuron, where φ_a, φ_b, and φ_c are the phases of $\boldsymbol{u}^G$, $\boldsymbol{v}^G$, and $\boldsymbol{w}^G$, respectively.

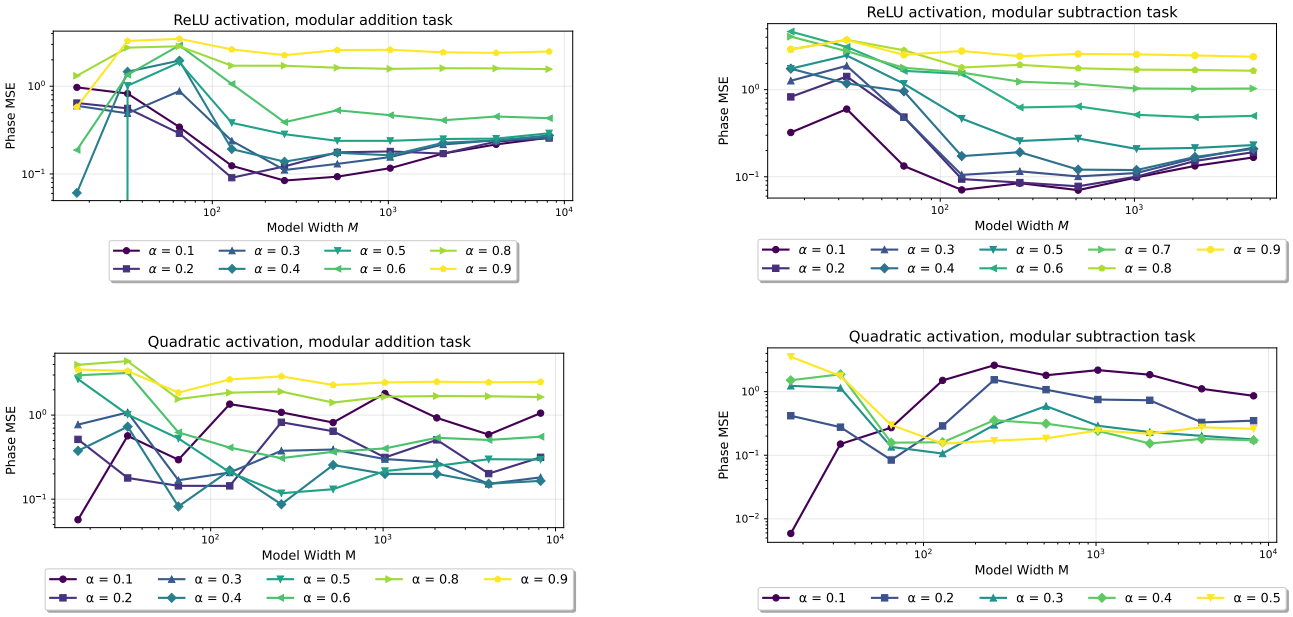


Figure 33. Phase MSE across different tasks, activation functions, model widths, and noise ratios. Higher noise ratios increase Phase MSE for ReLU models, and a similar trend is observed for quadratic models on the modular addition task.

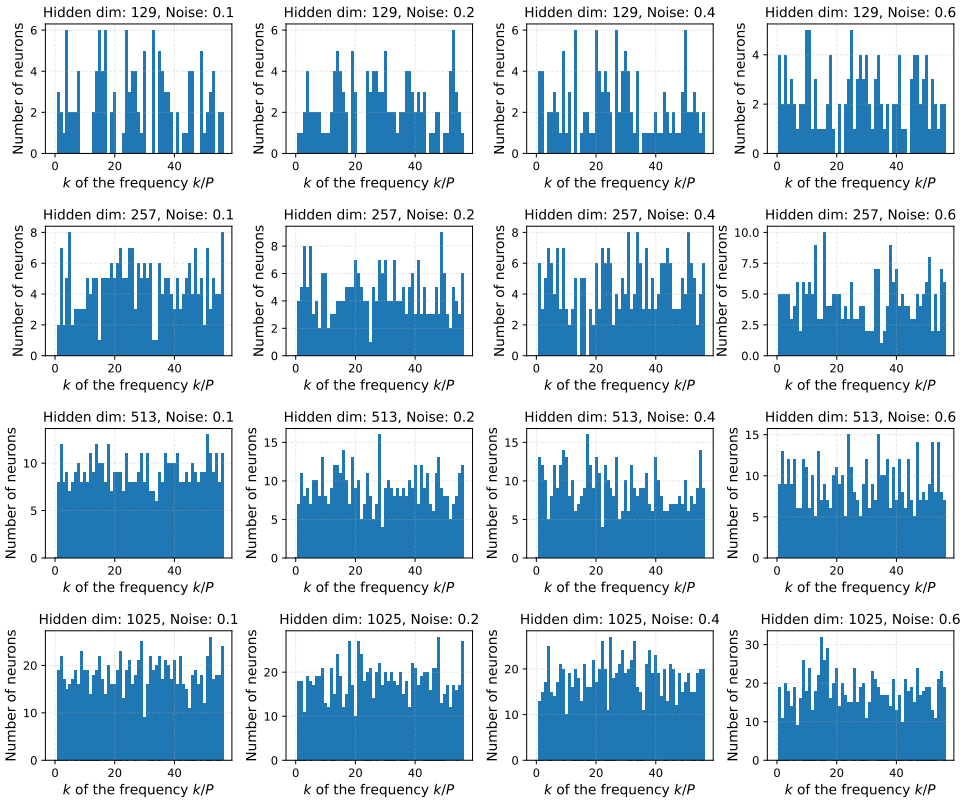


Figure 34. Each frequency $\omega \in \left\{1, \cdots, \frac{P-1}{2}\right\}$ is covered by at least a neuron when the model width is no less than 513.

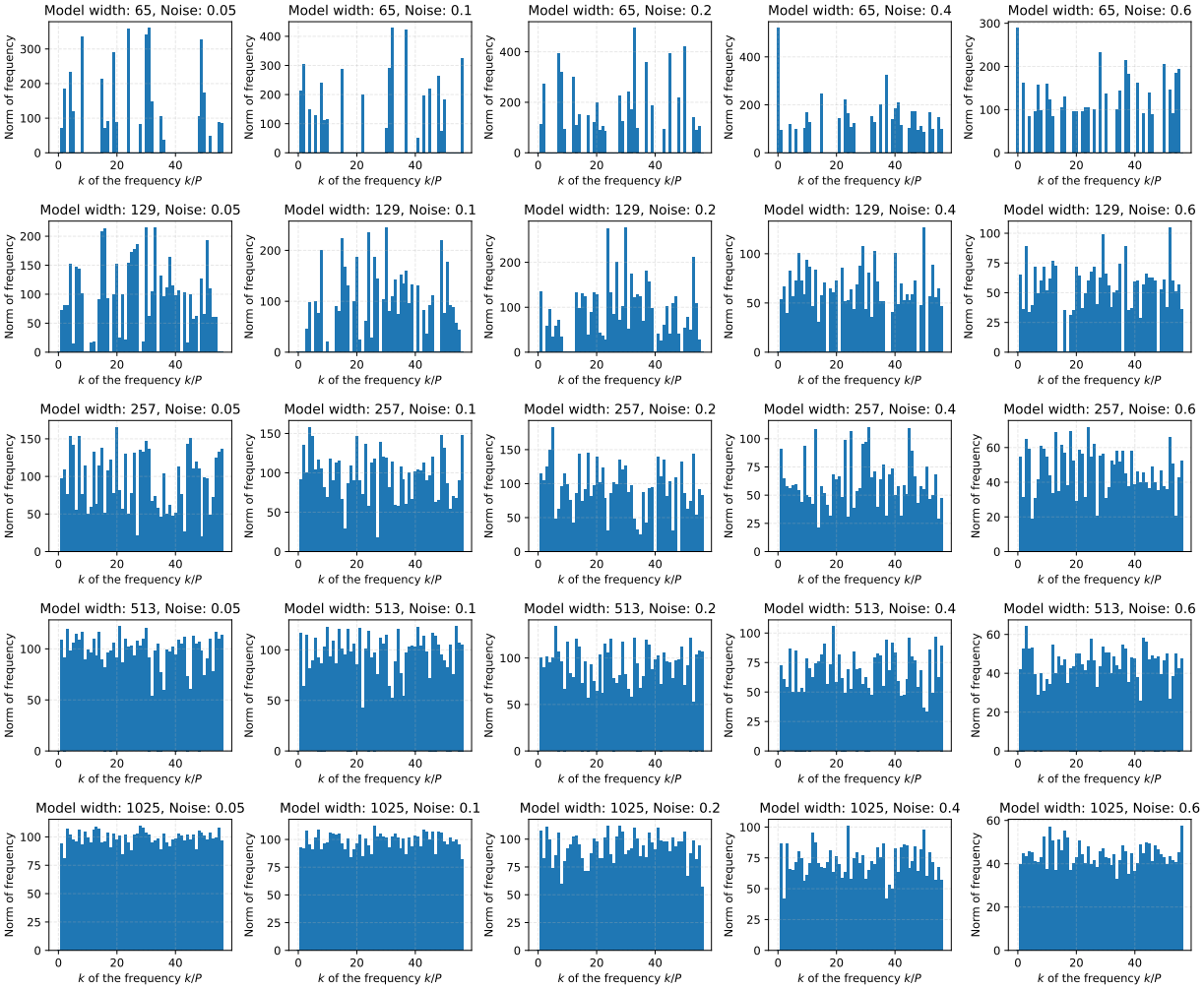


Figure 35. The magnitude (norm) of each frequency $\omega \in \left\{1, \cdots, \frac{P-1}{2}\right\}$ of $\boldsymbol{W}^G$ after frequency filtration to each neuron.

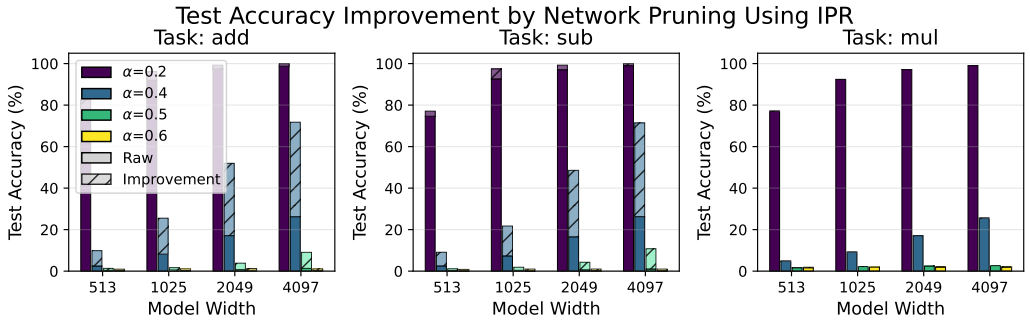


Figure 36. Generalization improvement across various tasks by neuron selection/pruning using IPR. No noticeable improvement is observed for the modular multiplication task.

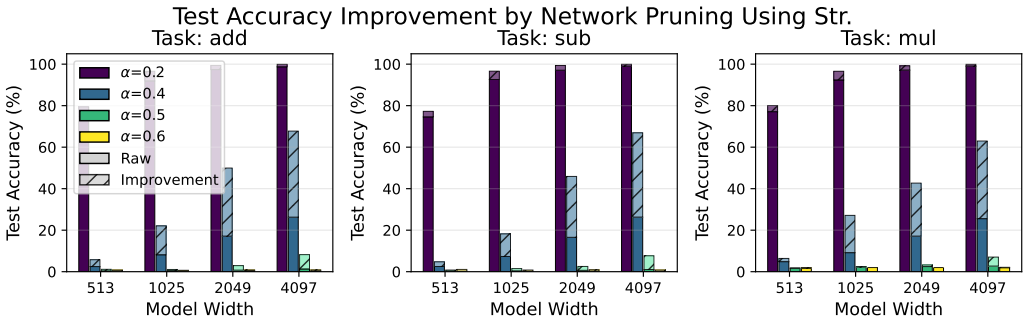


Figure 37. Generalization improvement across various tasks by neuron selection/pruning using Str. Significant improvement is also observed for the modular multiplication task.