Grokking Phenomenon Exploration

刘锦琪 2100011023

池方琦 2201110079

吴睿林 2200010914

Abstract

This report explores the grokking phenomenon described in the original paper [2], focusing on modular addition tasks. We reproduce key experiments from the paper, analyzing how different factors, such as data availability, model types (Transformer, MLP, LSTM), and training strategies (optimizers and regularization), affect learning and generalization. Additionally, we extend the study to multivariable modular addition to assess its complexity. Finally, we investigate the underlying mechanisms of grokking, focusing on the relationship between model parameter norms, the emergence of sparse subnetworks, and the role of input perturbations to accelerate grokking. This work sheds light on how models transition from overfitting to generalization under specific conditions.

1 Task 1: Reproducing the Results in Figure 1

1.1 Background and Settings

In this task, we investigate the grokking phenomenon as observed in the modular addition problem $(x, y) \to (x + y) \mod p$, with $x, y \in \mathbb{Z}_p$. The goal is to analyze how the training data fraction α affects the learning and generalization behavior of a Transformer model. Following the methodology in the original paper, the Transformer model is optimized using the AdamW optimizer.

For the experiments, we fix the modular base $p = 97$, and use the following hyperparameters:

- Learning rate $\text{lr} = 3 \times 10^{-3}$, with a decay factor $\text{lr\_gamma} = 0.99$.

- Weight decay is set to 1.0.

- The Transformer model consists of 2 layers with a sequence length $\text{seq\_len} = 2$ and 4 attention heads.

- Batch size is 512.

- The number of training epochs is at most 2000.

The integers are one-hot encoded, and the model architecture is designed to capture the modular addition operation efficiently.

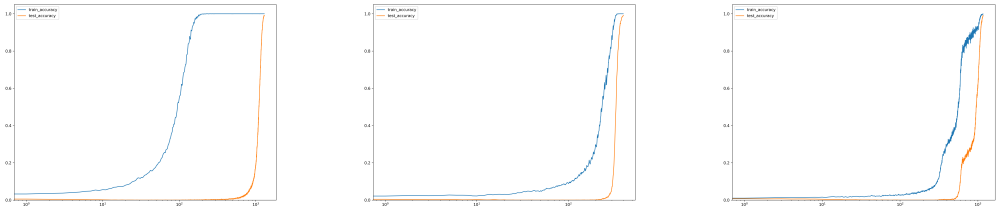
1.2 Experimental Design

The experiments are conducted by varying the training data fraction α, which represents the ratio of the training set size to the total dataset size. For each value of α, the Transformer model is trained and evaluated on both the training and test datasets. Accuracy and loss are recorded at each epoch.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

1.3 Results and Analysis

Figure 1 illustrate the curves of the training and test accuracy over the iterations of epochs, respectively, for different values of $\alpha = 0.3, 0.5, 0.7$. These curves provide insights into how the training and test performances evolve over time and how α influences the grokking phenomenon.



(a) $\alpha = 0.3$ (b) $\alpha = 0.5$ (c) $\alpha = 0.7$

Figure 1: Training and Test Accuracy Curves vs. Epochs

From the results:

- For smaller α (e.g., $\alpha = 0.3$), the model requires significantly more epochs to generalize, exhibiting a delayed grokking phenomenon.
- For larger α (e.g., $\alpha = 0.7$), the grokking time decreases, and the grokking phenomenon becomes less apparent.

1.4 Discussion

These results align with the findings of the original paper, demonstrating that the grokking phenomenon is highly sensitive to the training data fraction α. Smaller α values make it harder for the model to generalize, leading to delayed grokking. The results suggest that the amount of training data is a crucial factor for achieving generalization.

2 Task 2: Grokking on MLP and LSTM

2.1 Background and Settings

This task investigates the grokking phenomenon on MLP and LSTM architectures using the modular addition problem. Each model is trained on the same dataset under identical conditions, optimized with the AdamW optimizer (lr $= 3 \times 10^{-3}$) for a fixed number of epochs.

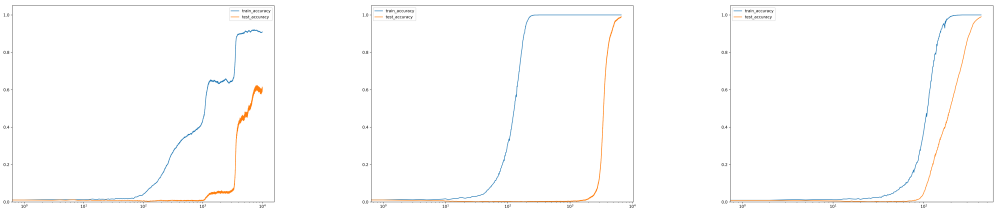
2.1.1 Model Architectures and Hyperparameters

- **MLP**:
  - Architecture: 5-layer feedforward network
  - Activation: GELU
  - Hidden dimensions: $512 \to 128 \to 64 \to 64$
- **LSTM**:
  - Number of layers: 2
  - Hidden dimensions: 128
  - Sequence length: 2

2.2 Results and Analysis

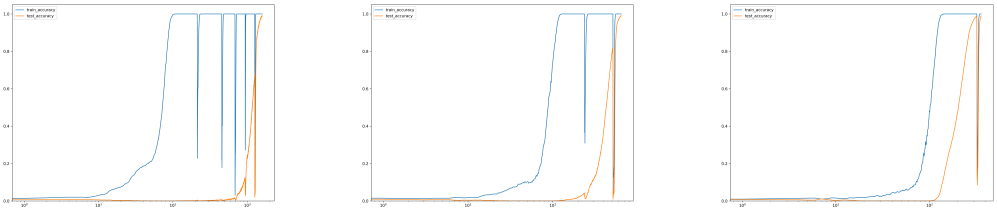
The experimental results for the MLP model are illustrated in Figure 2. A distinct grokking phenomenon is evident in the MLP model, with the training fraction showing a similar

trend to what is observed in the Transformer architecture. Similarly, Figure 3 presents the experimental outcomes for the LSTM model, which also exhibit the grokking phenomenon. Comparing these findings to the grokking behavior seen in the Transformer model, it is apparent that, given the same dataset and an equivalent number of parameters, both MLP and LSTM tend to grok later than the Transformer, even when trained with a larger weight decay. This observation suggests that the Transformer architecture may possess a notable advantage over the other two models.



(a) $\alpha = 0.3$ (b) $\alpha = 0.5$ (c) $\alpha = 0.7$

Figure 2: MLP Training and Test Accuracy Curves vs. Epochs



(a) $\alpha = 0.3$ (b) $\alpha = 0.5$ (c) $\alpha = 0.7$

Figure 3: LSTM Training and Test Accuracy Curves vs. Epochs

3 Task 3: Analyzing the Effects of Different Optimizers and Regularization Techniques

3.1 Background and Settings

The experiments focused on evaluating the Transformer architecture, with the task being the previously introduced modular addition problem ($q = 97$). The baseline model was trained using the AdamW optimizer. The optimization budget was capped at 15,000 steps, and training was repeated with random seeds 10, 11, and 12 to ensure robustness. The training data proportions ranged from 30% to 95% in steps of 5%.

The experiments evaluated various configurations, including:

- Full batch.
- Minibatch.
- Adam with a weight decay of 0.0003.
- AdamW with a weight decay of 0.3.
- AdamW with dropout (rate = 0.1).
- bigger learning rate
- smaller learning rate
- RMSProp

For each configuration, the best validation accuracy was recorded and plotted against the training data fraction.

3.2 Results and Discussion

Figure 4 illustrates the best validation accuracy for each combination of optimizer and regularization method across different training data fractions. Below, we discuss the key trends and observations from these experiments.

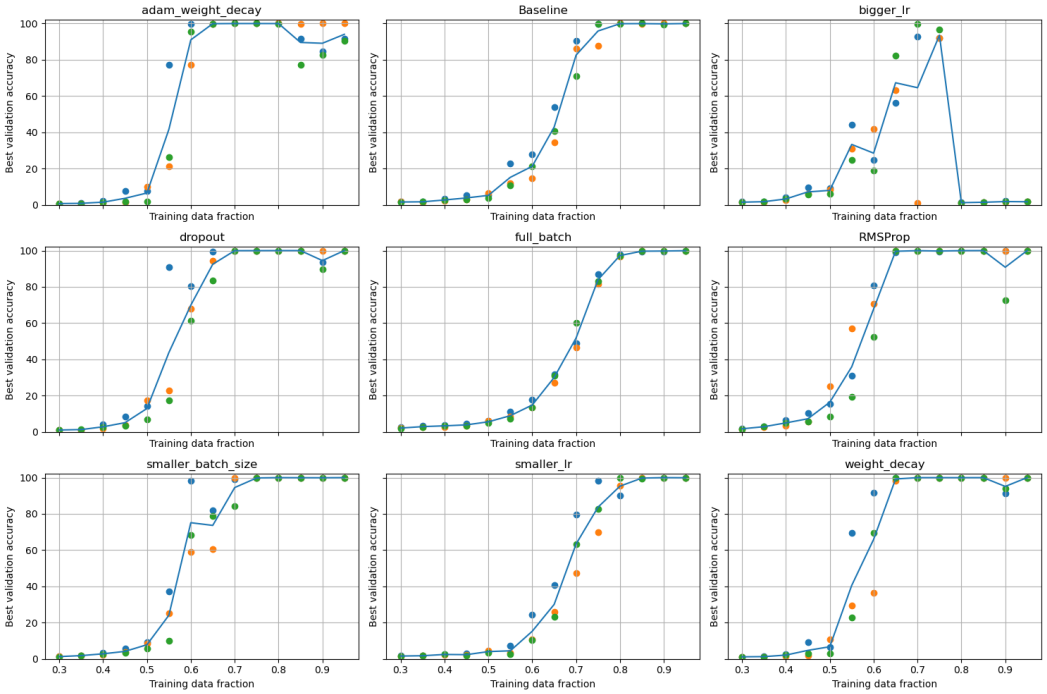


Figure 4: Best validation accuracy as a function of training data fraction for various optimizers and regularization methods. Each subplot corresponds to a specific experimental configuration.

We can see that weight decay plays a critical role in improving the generalization ability of the model. Additionally, using mini-batch training significantly enhances generalization, likely due to the inherent regularization effect of stochastic updates. And adding dropout can also improve the generalization ability. While leading to longer training time, a smaller learning rate cannot enhance performance. On the other hand, a too big learning rate will cause severe problem for optimization. For comparison of different optimizers, we use AdamW, RMSProp and Adam. Since AdamW is designed to decouple weight decay term in Adam, we use weight decay in Adam as well. After careful tuning, the three optimizers can achieve similar performance. However, it should be pointed out that RMSProp is the hardest to tune while AdamW is the easiest one, which is consistent with their development. Overall, weight decay and mini-batch training appear to be the most effective methods for improving generalization in this setup.

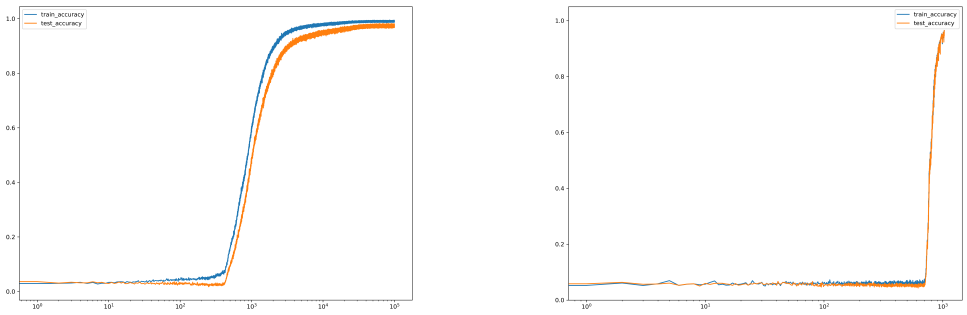
4 Task 4: Investigate the Impact of the Number of Summands

4.1 Background and Settings

In this section, we increase the difficulty of binary addition modulo operation to K-ary and observe the impact of K on grokking. Since the training set will increase exponentially with K, we use $p = 31, \alpha = 0.3$ for $K = 3$ and $p = 51, \alpha = 0.7$ for $K = 4$. This setup will slightly reduce the difficulty of the task and also alleviate the pressure on computational resources.

4.2 Results and Discussion

Figure 5 illustrates the training and testing accuracy for different K.



(a) $K = 3$ (b) $K = 4$

Figure 5: Training and Test Accuracy Curves vs. Epochs for different K

We can see that grokking phenomenon can hardly be observed for $K = 3$ and for $K = 4$, training and testing accuracy are nearly overlapping. This can be well explained by the result in Task 5. In fact, as the complexity of the problem increases, the solutions are likely to exhibit less sparsity. As a result, the grokking phenomenon vanishes.

5 Task 5: Explanation of Grokking Phenomenon

This section investigates the grokking phenomenon in the modular addition task. We analyze three aspects:

- the relationship between the l_2-norm of model parameters and grokking.
- the emergence of sparse subnetworks that enhance robustness and generalization.
- role of input perturbations in accelerating grokking.

Through theoretical insights and experimental results, we provide a comprehensive explanation of grokking in modular addition learning.

5.1 The Relationship Between l_2-Norm and Grokking

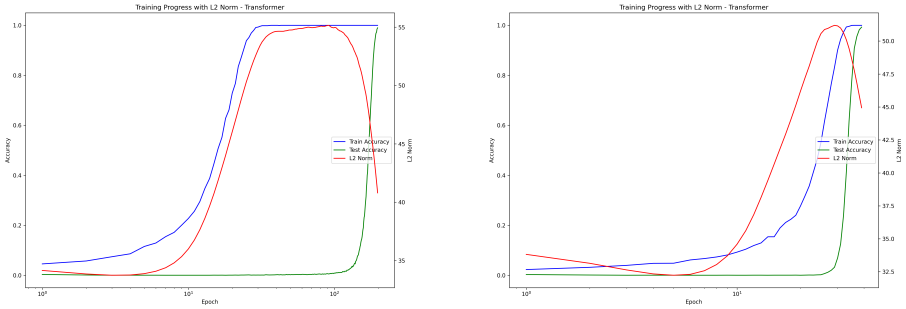
The l_2 norm of neural network parameters has been observed to correlate strongly with the grokking phenomenon. As shown in Figure 6, the training and test accuracy, along with the l_2-norm, evolve significantly over the course of training.

In the early stages of training, the l_2-norm increases steadily as the model learns to memorize the training data. During this period, the test accuracy remains close to zero, indicating that the model has not yet generalized. However, at the point where the l_2-norm begins to decrease, a corresponding rise in test accuracy is observed. This marks the onset of generalization and suggests that the reduction in the l_2 norm helps the model focus on more meaningful representations of the data.

During training process, the l_2-norm stabilizes, marking the grokking phase transition. This stabilization phase indicates that the model has shifted from memorization to generalization, with the l_2-norm constraining parameter magnitudes and enhancing robustness. By reducing the sensitivity of the model to noise and irrelevant features, the stabilization of the l_2-norm enables better generalization to unseen data.

These observations demonstrate a strong relationship between l_2-norm dynamics and test accuracy. Specifically, the onset of l_2-norm reduction aligns closely with the beginning of

generalization. This behavior aligns with prior research that highlights the importance of parameter norm control and weight decay method in facilitating generalization during grokking [3].



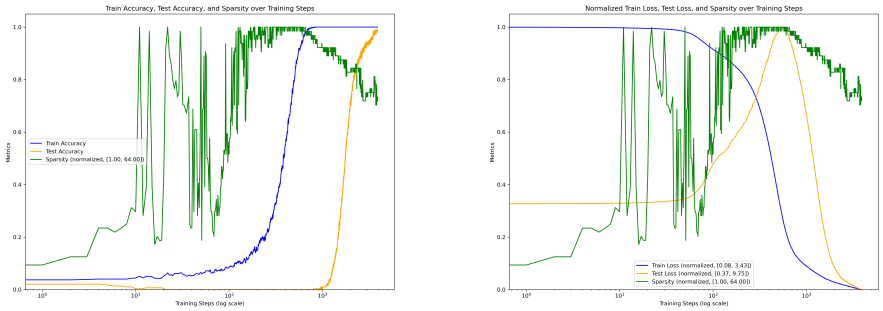
(a) $\alpha = 0.3$ (b) $\alpha = 0.5$

Figure 6: Training / Test Accuracy and l_2-Norm Curves vs. Epochs

5.2 The Role of Sparse Subnetworks in Generalization

As the model transitions to the generalization phase, a sparse subnetwork emerges and begins to dominate the network's predictions. Figure 7,illustrates the changes in network sparsity (measured as the ratio of active neurons in the sparse subnetwork to the total number of neurons [1]) during training. At the early stages of training, the decrease in training loss is accompanied by a decline in sparsity, where the dense subnetwork dominates, and the model primarily focuses on memorization. In contrast, the decrease in test loss is associated with an increase in sparsity, indicating that the model has discovered a sparse subnetwork, leading to improved generalization.

This phenomenon supports the hypothesis that the model progressively focuses on the most critical features of the modular addition task by utilizing a sparse subnetwork. Sparse subnetworks, by reducing redundancy and ignoring irrelevant features, enhance robustness and enable effective generalization. These findings are consistent with last section and prior work [1] that highlights the role of sparse structures in grokking.



(a) Network Sparsity and Accuracy (b) Network Sparsity and Loss

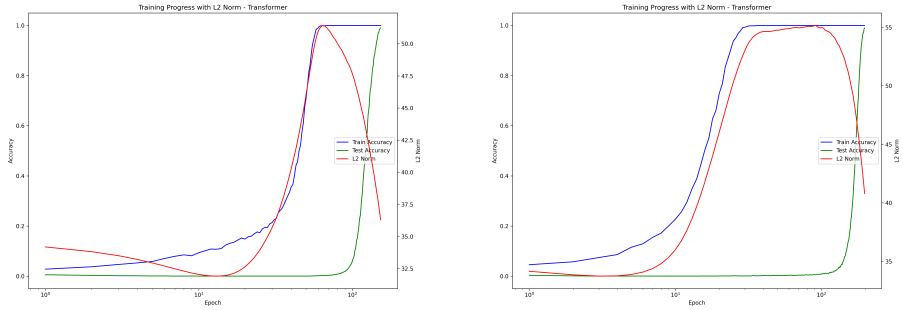
Figure 7: Accuracy, Loss and Sparsity of a 2-layer MLP model. The decrease in test loss is associated with an increase in sparsity

5.3 Accelerating Grokking Through Perturbation

Finally, we explored the use of perturbation-based methods to accelerate grokking. By injecting Gaussian noise into the input data during training, we observed that the test

accuracy improved significantly earlier than in standard training. Figure 8, present a comparison of the accuracy and l_2-norm of model parameters with respect to epochs under the same settings, between the method with added perturbations and the method without perturbations.

Perturbations encourage the model to focus on invariant features that are critical for solving the modular addition task. This aligns with previous studies [3] that demonstrate how robustness-enhancing techniques improve generalization by reducing reliance on spurious correlations.



(a) added perturbations (b) without perturbations

Figure 8: Training / Test Accuracy and l_2-Norm Curves vs. Epochs

5.4 Conclusion

In this section, we investigated the grokking phenomenon in the modular addition task from three perspectives. First, we demonstrated the relationship between the l_2-norm and grokking, emphasizing the role of robustness in generalization. Second, we highlighted the emergence of sparse subnetworks as a mechanism for improving robustness and generalization. Finally, we showed that perturbation-based methods can effectively accelerate grokking. These findings provide new insights into the dynamics of grokking and suggest strategies for optimizing training in neural networks.

References

[1] William Merrill, Nikolaos Tsilivis, and Aman Shukla. A tale of two circuits: Grokking as competition of sparse and dense subnetworks, 2023.

[2] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022.

[3] Zhiquan Tan and Weiran Huang. Understanding grokking through a robustness viewpoint, 2024.