When to Grow? A Fitting Risk-Aware Policy for Layer Growing in Deep Neural Networks

SPEAKER: NAGA SAI SIVANI, TUTIKA

AUTHOR: HAIHANG WU, WEI WANG, TAMASHA MALEPATHIRANA, DAMITH SANENAYAKE, DENNY OETOMO, SAMAN HALMUGE

Agenda

Introduction

- Deep Neural Networks (DNNs) have revolutionized computer vision, but training these large models is computationally intensive.
- Neural growth - growing a small network into a larger one - offers a promising solution for efficient training.
- While various aspects of neural growth have been studied, determining optimal growth timing remains largely unexplored.
- This paper introduces FRAGrow: a novel approach that automatically determines growth timing based on model's fitting risk.
- This method achieves up to 20% faster training while maintaining or improving accuracy compared to traditional approaches.

Objective

- This research makes three key contributions to the field of neural network growth:
- First, it aims to demonstrate that **neural growth** has an **inherent regularization effect** on network training, and show how the strength of this regularization is directly controlled by the timing of growth.
- Second, it proposes to develop a novel **fitting risk-aware policy (FRAGrow)** that can automatically adjust growth timing by evaluating the network's current fitting risks using the proposed overfitting risk level (ORL) metric.
- Third, it seeks to create a growth strategy that maintains model accuracy across different scenarios, specifically addressing both overfitting and underfitting cases that challenge existing growth approaches.

Neural Growth

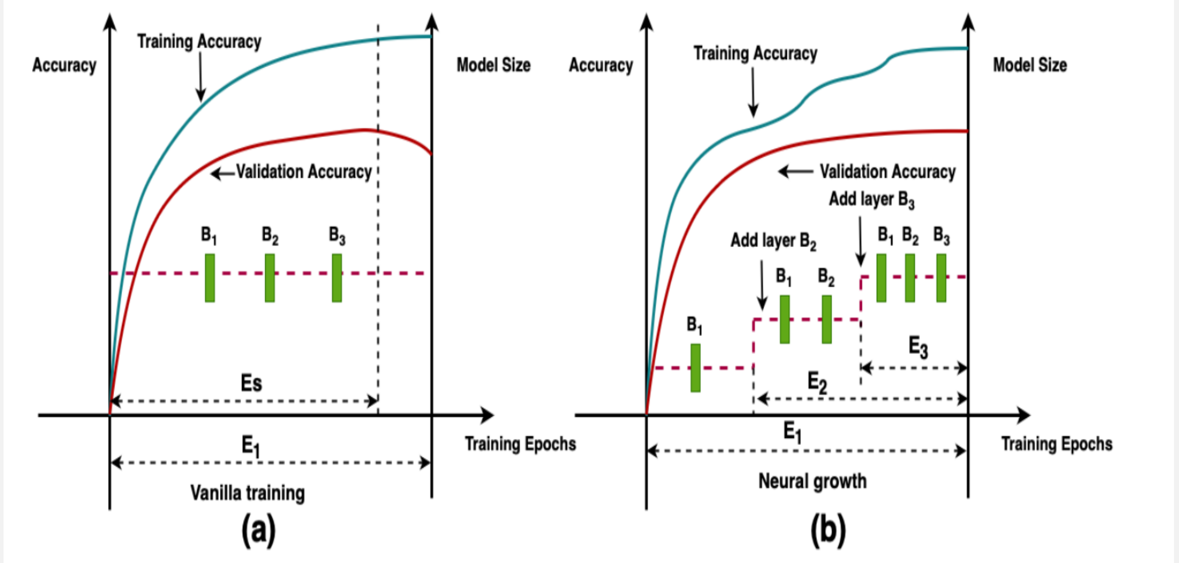


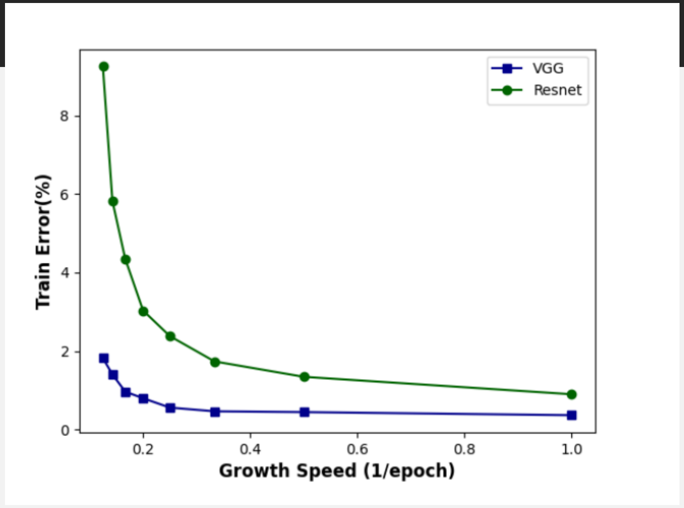
Figure 2: Comparison of vanilla training and neural growth. In standard training (vanilla), all layers undergo training for E_1 epochs. However, in the context of neural growth, only layer B_1 is trained for E_1 epochs, while the majority of layers (B_2 and B_3) are subjected to reduced training epochs (E_2 and E_3 respectively).

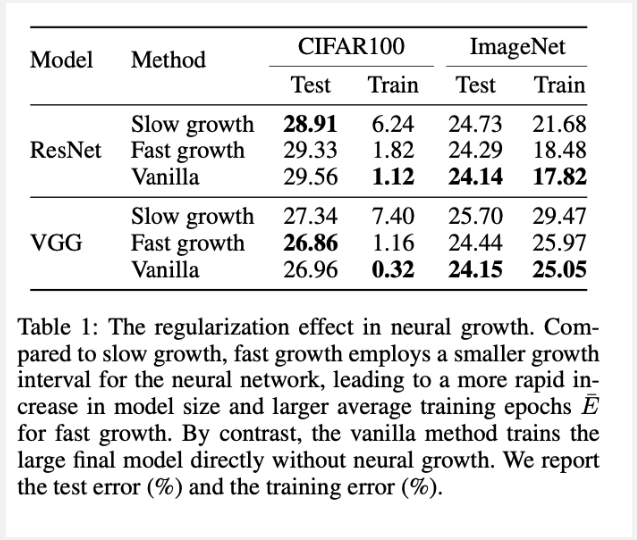
What is neural growth?
A process where a small neural network expands into a larger model, enhancing training efficiency.

How does it reduce training time?
New layers (B2, B3) inherit weights from previous layers and require fewer epochs to converge, speeding up training.

Why does it cause a regularization effect?
By training new layers for fewer epochs, neural growth reduces overfitting. It prevents noise memorization but may cause underfitting if growth is too slow, leading to decreased training accuracy while validation accuracy remains stable.

The Regularization Effect of Neural Growth





Model	Method	CIFAR100		ImageNet	
		Test	Train	Test	Train
ResNet	Slow growth	**28.91**	6.24	24.73	21.68
	Fast growth	29.33	1.82	24.29	18.48
	Vanilla	29.56	**1.12**	**24.14**	**17.82**
VGG	Slow growth	27.34	7.40	25.70	29.47
	Fast growth	**26.86**	1.16	24.44	25.97
	Vanilla	26.96	**0.32**	**24.15**	**25.05**

Table 1: The regularization effect in neural growth. Compared to slow growth, fast growth employs a smaller growth interval for the neural network, leading to a more rapid increase in model size and larger average training epochs $\bar{E}$ for fast growth. By contrast, the vanilla method trains the large final model directly without neural growth. We report the test error (%) and the training error (%).

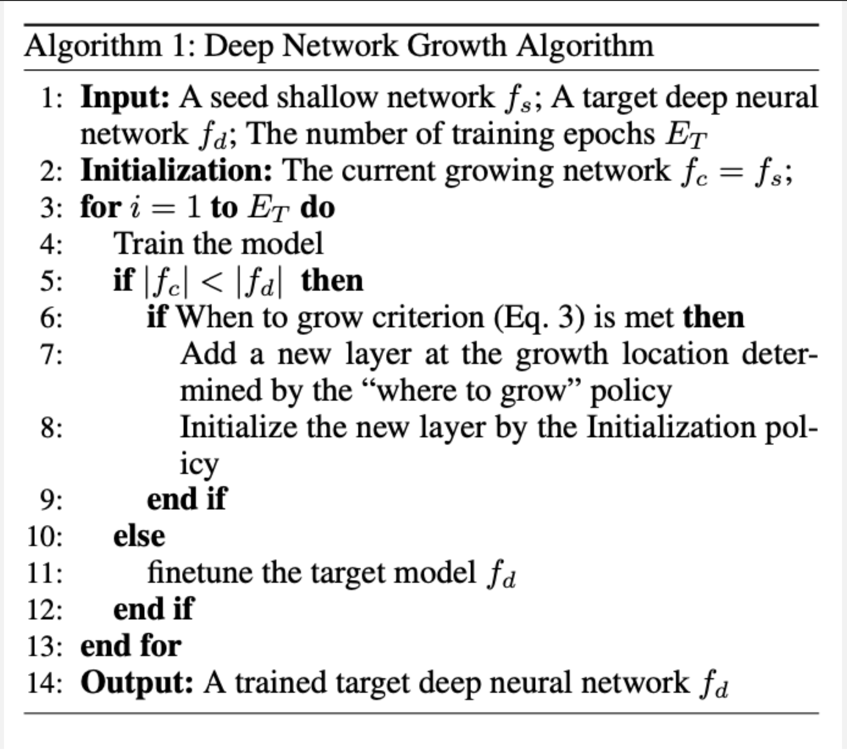
- Neural growth-induced regularization strength is determined by the average training epochs $\bar{E}$ as given in following equation.

$$\bar{E} = \frac{1}{n}\sum_{i=1}^{n} E_i$$

- From the graph faster growth (larger $\bar{E}$) weakens regularization, reducing training error, while slower growth (smaller $\bar{E}$) strengthens regularization, increasing training error.
- The same effect is seen in table1, where ResNet and VGG were trained with slow growth rate, fast growth rate and the standard method.
- In overfitting scenarios (CIFAR100), neural growth improves test accuracy due to regularization. In underfitting scenarios (ImageNet), it harms performance by reducing the model's learning capability.

The regularization effect of neural growth helps mitigate overfitting but can worsen underfitting, highlighting the need for a balanced growth policy like FRAGrow.

Proposed - When to Grow Policy



Algorithm 1: Deep Network Growth Algorithm

```
1:  Input: A seed shallow network $f_s$; A target deep neural network $f_d$; The number of training epochs $E_T$
2:  Initialization: The current growing network $f_c = f_s$;
3:  for $i = 1$ to $E_T$ do
4:      Train the model
5:      if $|f_c| < |f_d|$ then
6:          if When to grow criterion (Eq. 3) is met then
7:              Add a new layer at the growth location determined by the "where to grow" policy
8:              Initialize the new layer by the Initialization policy
9:          end if
10:     else
11:         finetune the target model $f_d$
12:     end if
13: end for
14: Output: A trained target deep neural network $f_d$
```

$$ORL = \text{train accuracy} - \text{validation accuracy} \qquad (2)$$

$$I = \frac{I_{max}}{1 + e^{\alpha - ORL}} \qquad (3)$$

$$I_{max} = \frac{E_T - E_F^{min}}{n} \qquad (4)$$

- A "when to grow" policy determines the growth timing t1,...,tn for n added blocks, which in turn influences the average training epochs $\bar{E}$ and controls the regularization strength.
- We use ORL to determine the risk of overfitting or underfitting.
- Our algorithm appends network layers after the dynamic growth interval I specified by Equation 3 where Imax is the maximum allowed growth interval, and α is a hyperparameter.
- **High ORL** indicates overfitting, while **low ORL** suggests underfitting.
- The algorithm uses Eq 3 to decide when to grow a neural network. Handling the overfitting and underfitting problems.

Experiments - Datasets

Dataset	Classes	Training Images	Validation Images	Testing Images	Image Resolution
ImageNet	1000	1.27 million	10,000	50,000	224x224
CIFAR-10	10	49,500	500	10,000	32x32
CIFAR-100	100	49,500	500	10,000	32x32

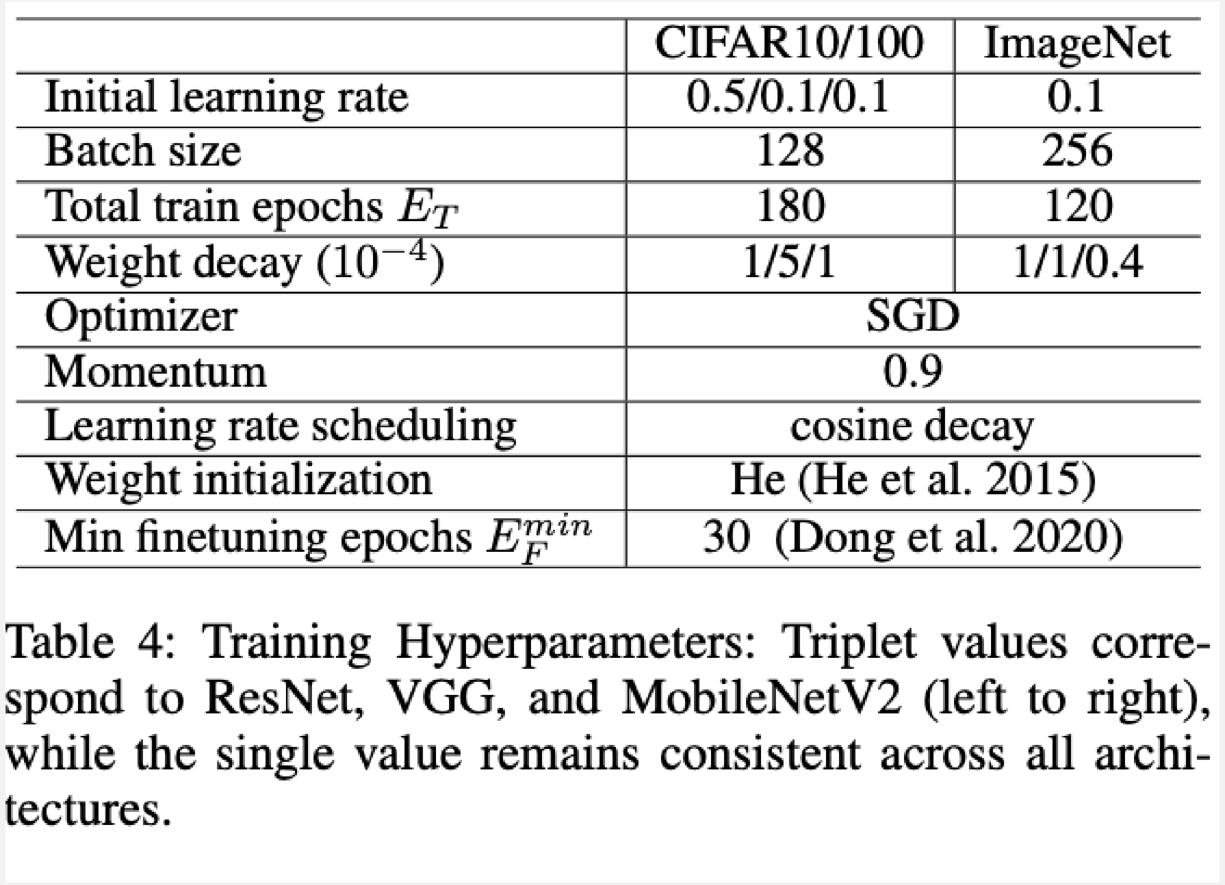
Experiments - Models

Model Type	Base Model (seed shallow network)	Architecture	Stages	Target Deep Model
Single-Branch	VGG-1-1-1-1-2	VGGNet	5	VGG-2-2-4-4-4
Multi-Branch	ResNet-2-2-2-2	ResNet	4	ResNet-8-8-8-8
Multi-Branch	MobileNetV2-1-1-1-1-1	MobileNetV2	5	MobileNetV2-2-3-4-3-3

Experiments – Growth Methods

Method	Description
Periodic Growth	Inserts new layers at fixed intervals of I_{max}.
Convergent Growth	Adds blocks only when accuracy stagnates.
LipGrow	Grows the model when the Lipschitz constant exceeds a threshold.
Proposed	FRAGrow model.

Experiments – Hyperparameters Used



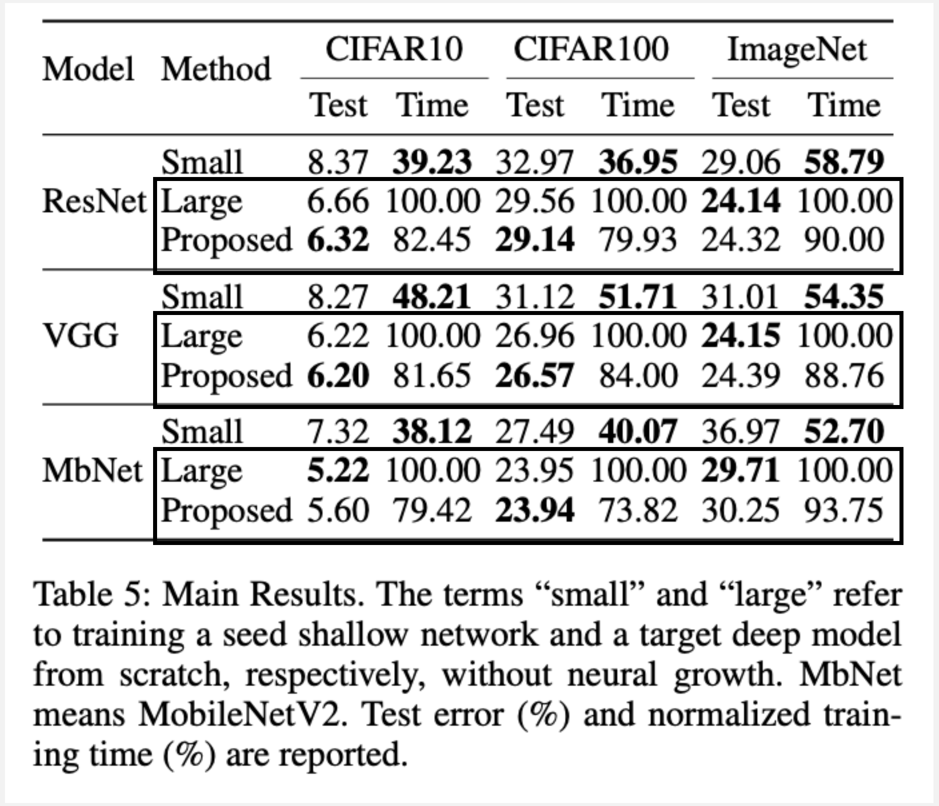
	CIFAR10/100	ImageNet
Initial learning rate	0.5/0.1/0.1	0.1
Batch size	128	256
Total train epochs E_T	180	120
Weight decay (10^{-4})	1/5/1	1/1/0.4
Optimizer	SGD	
Momentum	0.9	
Learning rate scheduling	cosine decay	
Weight initialization	He (He et al. 2015)	
Min finetuning epochs E_F^{min}	30 (Dong et al. 2020)	

Table 4: Training Hyperparameters: Triplet values correspond to ResNet, VGG, and MobileNetV2 (left to right), while the single value remains consistent across all architectures.

Learning Rate

- Two learning rates were investigated: constant large learning rate and cosine annealing with restart.
- Constant large learning rate was selected as it outperformed cosine annealing.

Results – small, large & proposed

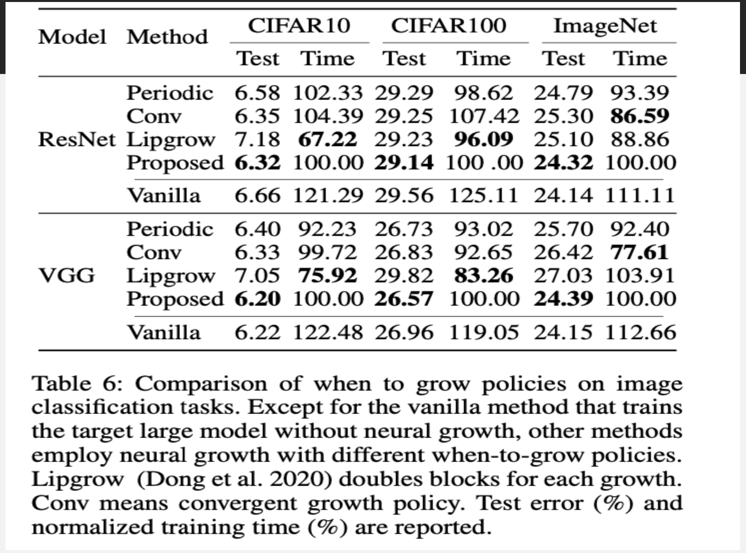


Model	Method	CIFAR10		CIFAR100		ImageNet	
		Test	Time	Test	Time	Test	Time
ResNet	Small	8.37	**39.23**	32.97	**36.95**	29.06	**58.79**
	Large	6.66	100.00	29.56	100.00	**24.14**	100.00
	Proposed	**6.32**	82.45	**29.14**	79.93	24.32	90.00
VGG	Small	8.27	**48.21**	31.12	**51.71**	31.01	**54.35**
	Large	6.22	100.00	26.96	100.00	**24.15**	100.00
	Proposed	**6.20**	81.65	**26.57**	84.00	24.39	88.76
MbNet	Small	7.32	**38.12**	27.49	**40.07**	36.97	**52.70**
	Large	**5.22**	100.00	23.95	100.00	**29.71**	100.00
	Proposed	5.60	79.42	**23.94**	73.82	30.25	93.75

Table 5: Main Results. The terms "small" and "large" refer to training a seed shallow network and a target deep model from scratch, respectively, without neural growth. MbNet means MobileNetV2. Test error (%) and normalized training time (%) are reported.

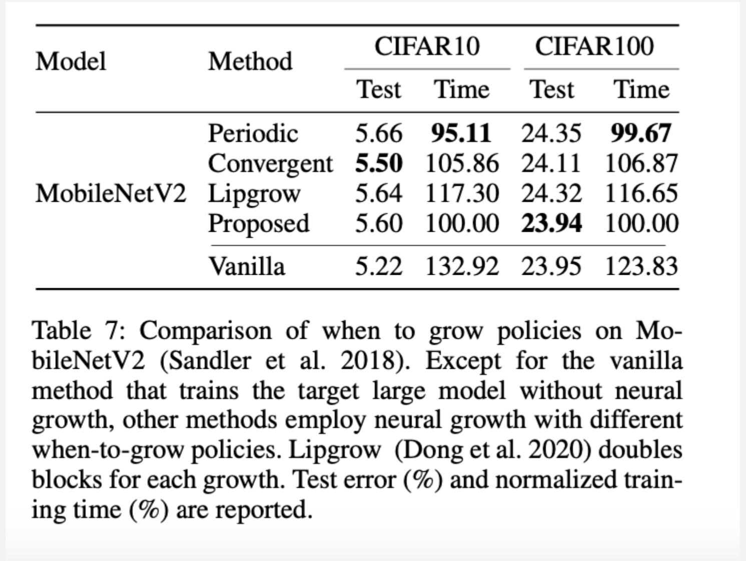
- First experiment, we compared **neural growth method** with traditional methods that train small or large models directly, results are in Table 5
- Proposed model shows **comparable accuracy** to large models while requiring **less training time**.
- On **CIFAR10/100**, neural growth improves **test error by ~0.3% for ResNet and VGG** while reducing **training time by ~20%**, due to its **regularization effect** on overfitting models like **VGG-2-2-4-4-4 and ResNet-8-8-8-8**.
- On **ImageNet**, the method results in a **minor accuracy drop (~0.32%)** but still saves **~10% training time** compared to directly training large models.
- **CIFAR10/100 benefits more from time savings** than ImageNet because the method **automatically adjusts growth speed**—slower for **overfitted datasets (CIFAR10/100)** and faster for **underfitted datasets (ImageNet)**, maximizing efficiency.

Results – Periodic, Convergent, LipGrow



Model	Method	CIFAR10		CIFAR100		ImageNet	
		Test	Time	Test	Time	Test	Time
ResNet	Periodic	6.58	102.33	29.29	98.62	24.79	93.39
	Conv	6.35	104.39	29.25	107.42	25.30	**86.59**
	Lipgrow	7.18	**67.22**	29.23	**96.09**	25.10	88.86
	Proposed	**6.32**	100.00	**29.14**	100 .00	**24.32**	100.00
	Vanilla	6.66	121.29	29.56	125.11	24.14	111.11
VGG	Periodic	6.40	92.23	26.73	93.02	25.70	92.40
	Conv	6.33	99.72	26.83	92.65	26.42	**77.61**
	Lipgrow	7.05	**75.92**	29.82	**83.26**	27.03	103.91
	Proposed	**6.20**	100.00	**26.57**	100.00	**24.39**	100.00
	Vanilla	6.22	122.48	26.96	119.05	24.15	112.66

Table 6: Comparison of when to grow policies on image classification tasks. Except for the vanilla method that trains the target large model without neural growth, other methods employ neural growth with different when-to-grow policies. Lipgrow (Dong et al. 2020) doubles blocks for each growth. Conv means convergent growth policy. Test error (%) and normalized training time (%) are reported.



Model	Method	CIFAR10		CIFAR100	
		Test	Time	Test	Time
MobileNetV2	Periodic	5.66	**95.11**	24.35	**99.67**
	Convergent	**5.50**	105.86	24.11	106.87
	Lipgrow	5.64	117.30	24.32	116.65
	Proposed	5.60	100.00	**23.94**	100.00
	Vanilla	5.22	132.92	23.95	123.83

Table 7: Comparison of when to grow policies on MobileNetV2 (Sandler et al. 2018). Except for the vanilla method that trains the target large model without neural growth, other methods employ neural growth with different when-to-grow policies. Lipgrow (Dong et al. 2020) doubles blocks for each growth. Test error (%) and normalized training time (%) are reported.

- In the next experiment, we compare our proposed model with three baseline methods: **periodic**, **convergent**, and **LipGrow**, along with the **vanilla method**. The results are shown in **Table 6** and **Table 7**.
- On the **overfitting CIFAR-10/100 dataset**, all methods except **LipGrow (Dong et al., 2020)** achieve similar accuracy with the target model, indicating balanced growth speed and regularization.
- The proposed method outperforms **periodic growth** by **0.47% for ResNet** and **1.31% for VGG** on **ImageNet** due to its **adaptive growth speed**, reducing excessive regularization.
- **Periodic** and **convergent growth** are too slow for **ImageNet**, leading to underfitting and lower accuracy, while faster growth is crucial for **VGG** due to its higher underfitting risk.
- However, our method requires **longer training time** than periodic growth, as the growth interval never exceeds that of periodic growth, resulting in faster growth but less time saved overall.

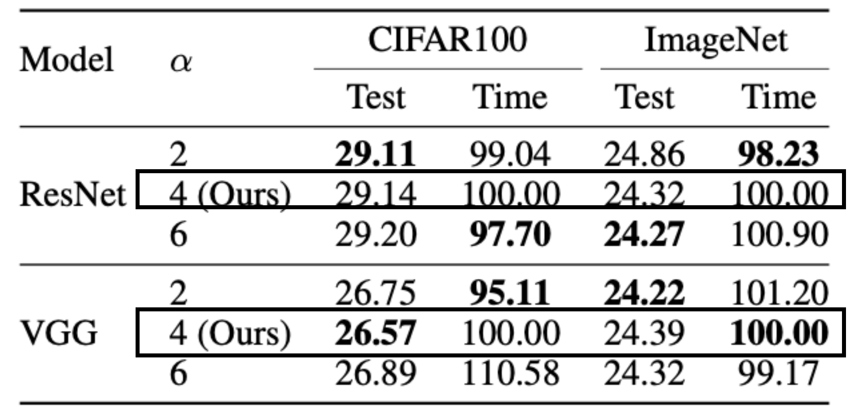
In conclusion, the proposed method outperformed for CIFAR 10/100 datasets for VGG and ResNet. It didn’t beat other methods in training time but did give better accuracy in most cases.

Ablation Study

How does the choice of hyperparameters impact the performance of FRAGrow?

How robust is our growth timing policy to growth order and initialization of new layers of neural growth?

Effect of Hyperparameter α



Model	α	CIFAR100		ImageNet	
		Test	Time	Test	Time
ResNet	2	**29.11**	99.04	24.86	**98.23**
	4 (Ours)	29.14	100.00	24.32	100.00
	6	29.20	**97.70**	**24.27**	100.90
VGG	2	26.75	**95.11**	**24.22**	101.20
	4 (Ours)	**26.57**	100.00	24.39	**100.00**
	6	26.89	110.58	24.32	99.17

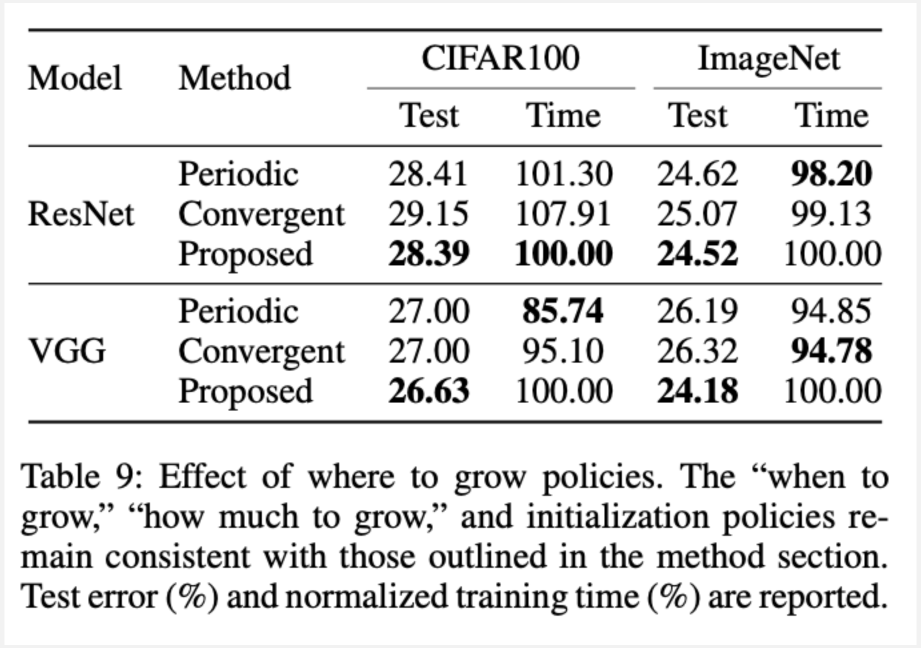
Table 8: Effect of alpha. All neural growth policies are consistent with those described in the method section. Test error (%) and normalized training time (%) are reported.

$$I = \frac{I_{max}}{1 + e^{\alpha - ORL}} \tag{3}$$

$$I_{max} = \frac{E_T - E_F^{min}}{n} \tag{4}$$

- **α in Equation 3** controls the **growth interval**, affecting training speed and regularization strength.
- **Reducing α from 6 to 2** caused **accuracy drops (~0.6%) on ImageNet (ResNet)** due to **slower growth and stronger regularization**, which negatively impacts underfitting datasets.
- **Smaller α values** improve **training efficiency** by reducing computation time, as observed with **VGG on CIFAR100**.
- An **optimal α** balances **accuracy and efficiency**, with **α = 4** performing best across datasets.

Effect of Growth Order

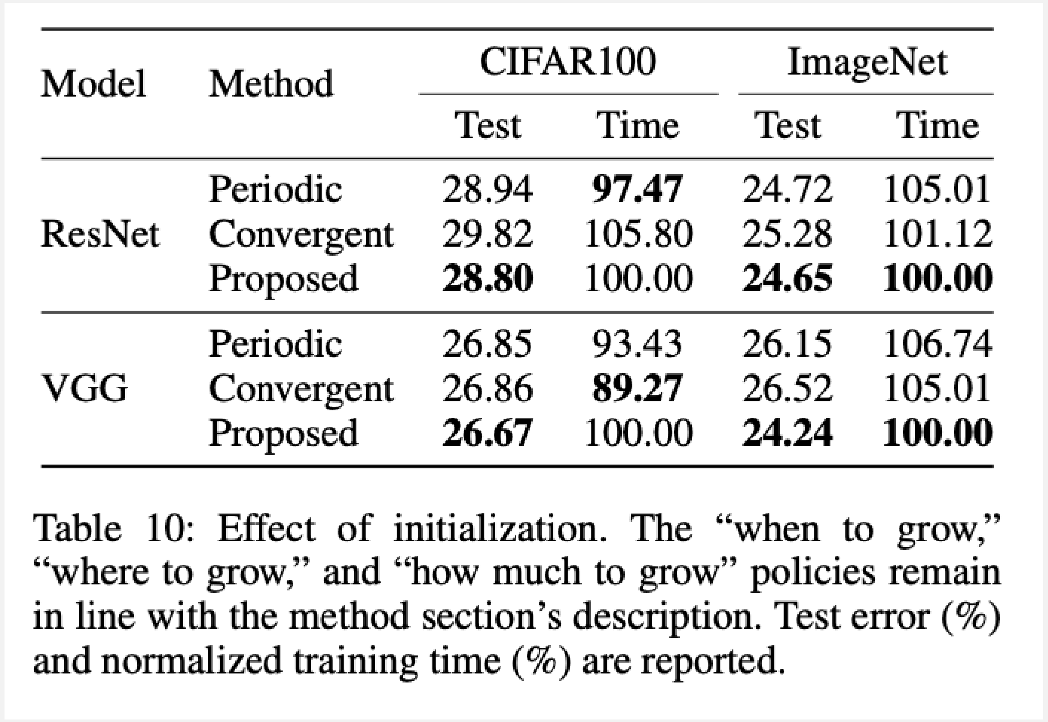


Model	Method	CIFAR100		ImageNet	
		Test	Time	Test	Time
ResNet	Periodic	28.41	101.30	24.62	**98.20**
	Convergent	29.15	107.91	25.07	99.13
	Proposed	**28.39**	**100.00**	**24.52**	100.00
VGG	Periodic	27.00	**85.74**	26.19	94.85
	Convergent	27.00	95.10	26.32	**94.78**
	Proposed	**26.63**	100.00	**24.18**	100.00

Table 9: Effect of where to grow policies. The "when to grow," "how much to grow," and initialization policies remain consistent with those outlined in the method section. Test error (%) and normalized training time (%) are reported.

- **Replaces layer-wise growth** with **circulation growth** (Wen et al. 2020), where new blocks are added as stages are visited.
- Demonstrates **robustness to growth order**, achieving comparable or higher accuracy than baseline methods.
- This method improves VGG accuracy by ~2% on underfitting ImageNet.
- Although accuracy improved, the training time is slightly more than other methods.

Effect of Initialization



Model	Method	CIFAR100		ImageNet	
		Test	Time	Test	Time
ResNet	Periodic	28.94	**97.47**	24.72	105.01
	Convergent	29.82	105.80	25.28	101.12
	Proposed	**28.80**	100.00	**24.65**	**100.00**
VGG	Periodic	26.85	93.43	26.15	106.74
	Convergent	26.86	**89.27**	26.52	105.01
	Proposed	**26.67**	100.00	**24.24**	**100.00**

Table 10: Effect of initialization. The "when to grow," "where to grow," and "how much to grow" policies remain in line with the method section's description. Test error (%) and normalized training time (%) are reported.

- To evaluate the influence of initialization, they used **Moment growth** (Li et al. 2022), it is a weight initialization strategy for expanding neural networks by copying weights from preceding layers and using historical moving averages. Ensuring smooth training.
- While it performs similarly to baseline methods on CIFAR-100, it achieves **~1% higher accuracy** on the underfitting ImageNet dataset.
- Despite the added computational overhead from tracking weight averages, the faster model expansion on ImageNet offsets this, leading to slightly reduced training time.

Summary

Conclusion

- Neural growth introduces a regularization effect, which can be controlled by growth timing.
- FRAGrow dynamically adjusts growth timing to address underfitting and overfitting risks.
- And achieves superior accuracy on underfitting-prone models while maintaining performance on overfitting-prone models.

Future Scope

- Extend FRAGrow to other vision tasks (e.g., dense prediction tasks).
- Improve training efficiency for larger-scale models.

Thank you!