

Scaling Test-Time Compute Can Outperform Larger Architectures in Computer Vision

Erfan Darzi^{1,2} Dylan Nguyen² George Cheng²

¹Harvard University, Cambridge, MA, USA

²Massachusetts Institute of Technology, Cambridge, MA, USA

{darzi, dylan, gjcheng}@mit.edu

Abstract

Deep neural networks face a fundamental trade-off between computational efficiency and accuracy. This paper introduces a method for network depth optimization that enables flexible inference with adjustable computational budgets while potentially improving training dynamics. Our approach partitions each residual stage into core and gated sub-paths, employing depth-aware training to develop networks that can operate at varying depths. We present theoretical analysis of our method through three key results: (1) an explicit regularization theorem quantifying how our training approach may penalize discrepancies between network configurations, (2) a statistical convergence theorem suggesting tighter generalization bounds based on effective network depth, and (3) a gradient dynamics theorem characterizing the noise properties induced by our training procedure. Empirically, our method shows improvements over conventional approaches on standard benchmarks, achieving favorable accuracy-efficiency trade-offs with a single trained model. The Gated Depth architecture provides a framework for deploying deep networks across diverse computational environments.

1. Introduction

Deep neural networks have achieved remarkable success in various vision tasks. Recent approaches have explored different efficiency techniques in deep architectures: networks with sub-paths enable dynamic inference by partitioning residual stages, while depth regularization techniques mitigate vanishing gradients by modifying layer behavior during training. Despite their individual benefits, these approaches have not been thoroughly analyzed theoretically.

The efficiency of deep neural networks has become increasingly critical as models are deployed across diverse computational environments. Traditional approaches to model efficiency typically yield static models optimized for

specific computational budgets, requiring multiple separate models for different deployment scenarios. Modern applications instead demand adaptive models that can dynamically adjust their computational footprint without sacrificing performance [1].

Existing adaptive inference methods primarily focus on architectural modifications enabling runtime adjustments, such as early-exit mechanisms[2] or dynamic channel width adjustment. However, these approaches often require complex training procedures and lack formal theoretical guarantees [3]. Concurrently, stochastic training techniques like Dropout improve model robustness by simulating ensembles, but lack structured adaptivity at inference time and don't provide principled ways to select which layers to execute for different efficiency targets [4].

In this paper, we present a Gated Depth approach for network optimization. Our key insight is that depth-aware training of gated sub-paths encourages these paths to learn refinement transformations rather than essential features, as the network must perform well across different execution patterns. By formalizing this mathematically, we show that our training procedure implicitly penalizes discrepancies between different sub-network configurations and introduces beneficial gradient noise proportional to the variance in configurations. This approach helps escape poor local minima and improves generalization through effective parameter sharing across different network depths.

Our empirical evaluation demonstrates that Gated Depth Networks consistently perform well across standard benchmarks, with particularly pronounced advantages at intermediate operating points where we observe smoother accuracy-efficiency trade-offs [5]. This validates our theoretical analysis and confirms that our approach successfully achieves superior performance.

Our main contributions include: (1) a Gated Depth architecture for flexible inference ; (2) Depth scheduling and skip-aware normalization; and (3) Theoretical guarantees through analysis of regularization effects, generalization bounds, and optimization dynamics.

2. Related Work

Adaptive Inference Methods. Han et al. [1] categorize dynamic neural networks as instance-wise, spatial-wise, or temporal-wise methods. Early adaptive approaches include early-exit networks [6, 7] (using intermediate classifiers), SkipNet [8] (reinforcement learning for layer selection), MS-DNet [7] (multiple exit points), and more recent methods like joint gating-classifier learning [2] and LayerDrop [9] (inference-time layer selection via dropout training). These approaches enable variable computational paths by dividing networks into core and gated components. However, they typically require specialized training procedures (complex loss functions, multiple forward passes) and often optimize only for extreme configurations (full or minimal networks), leaving intermediate configurations suboptimally trained. Recent extensions include adaptive object detection [5] with dynamic routing policies and adaptive depth networks [10], which we enhance with theoretical analysis and improved training dynamics.

Stochastic Regularization Techniques. Stochastic regularization improves generalization through various selective deactivation strategies: Dropout [11] zeroes activations, DropConnect [12] drops weights, and "early dropout" [4] reduces gradient noise by applying dropout only during early training. Stochastic depth [13] randomly bypasses entire layers, creating implicit ensembles of varying depths with shared weights, addressing gradient vanishing and accelerating training. Recent variants include DropBlock [14] (dropping contiguous feature regions), Drop-Path [15] (removing paths in multi-branch architectures), SMoE-Dropout [16] (random routing mixture-of-experts), and ChannelDrop-Back [17] (randomizing backward passes while preserving forward computation). Despite their effectiveness during training, these approaches lack structured adaptivity at inference time and don't provide principled methods for selective layer execution to meet efficiency targets. Standard stochastic depth also risks disrupting feature hierarchies through uniform dropout across all layers.

3. Sub-Paths with Stochastic Gating

Consider a deep neural network with N_r residual stages, where each stage $s \in \{1, \dots, N_r\}$ consists of multiple residual blocks. Our framework partitions each stage into a core base sub-path F_{base}^s and a gated refinement sub-path F_{gated}^s .

For each residual stage s , we define a Bernoulli random variable $b^s \sim \text{Bernoulli}(p_s)$ that determines whether the gated sub-path is active ($b^s = 1$) or bypassed ($b^s = 0$) during training. The forward propagation through stage s is

formulated as:

$$h^s = h^{s-1} + F_{\text{base}}^s(h^{s-1}) + b^s F_{\text{gated}}^s(h^{s-1} + F_{\text{base}}^s(h^{s-1})), \quad (1)$$

where $b^s \in \{0, 1\}$ is a Bernoulli random variable indicating whether the gated sub-path is active ($b^s = 1$) or bypassed ($b^s = 0$). This formulation represents a network architecture that preserves hierarchical feature representation while enabling flexible depth adaptation.

Let $\mathbf{b} = (b^1, \dots, b^{N_r})$ denote the vector of binary gating variables for all stages. During training, \mathbf{b} is sampled from the product distribution $\prod_{s=1}^{N_r} \text{Bernoulli}(p_s)$, which yields 2^{N_r} possible sub-network configurations. The network output for input x can be written as $f(x; \theta, \mathbf{b})$, where θ represents the shared parameters across all configurations.

4. Training

Expected Risk Minimization We formulate our training objective as expected risk minimization over skip pattern distributions. For dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ and task-specific loss L , we optimize:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{b} \sim P(\mathbf{b})} [L(f(x; \theta, \mathbf{b}), y)] \quad (2)$$

$$= \mathbb{E}_{\mathbf{b}} \left[\frac{1}{n} \sum_{i=1}^n L(f(x_i; \theta, \mathbf{b}), y_i) \right] \quad (3)$$

where $P(\mathbf{b}) = \prod_{s=1}^{N_r} \text{Bernoulli}(p_s)$.

For each mini-batch $\mathcal{B} \subseteq \mathcal{D}$, we sample one gate configuration \mathbf{b} and compute:

$$\hat{\mathcal{L}}(\theta, \mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{(x_i, y_i) \in \mathcal{B}} L(f(x_i; \theta, \mathbf{b}), y_i) \quad (4)$$

This efficiently trains an implicit ensemble of 2^{N_r} sub-networks with shared parameters, requiring only a single forward-backward pass per mini-batch. Importantly, our formulation encourages gated sub-paths to learn refinements rather than essential features, as they are randomly bypassed during training with probability $(1 - p_s)$.

Self-Distillation While stochastic gating naturally aligns the behavior of different sub-networks through parameter sharing, we introduce a periodic self-distillation phase to ensure optimal performance on the full network. This addresses a potential limitation of pure stochastic training: if the full network configuration is rarely sampled during training (especially when many stages have low gating probabilities), its performance might be suboptimal.

At regular intervals during training (every k iterations), we perform a two-pass update:

$$\mathbf{b}_1 = \mathbf{0} \quad (\text{all gated sub-paths bypassed}) \quad (5)$$

$$\mathbf{b}_2 = \mathbf{1} \quad (\text{all gated sub-paths active}) \quad (6)$$

We then compute a distillation loss:

$$L_{\text{distill}} = \alpha L_{\text{CE}}(f(x; \theta, \mathbf{b}_1), y) + (1 - \alpha) D_{\text{KL}}(f(x; \theta, \mathbf{b}_2) \| f(x; \theta, \mathbf{b}_1)) \quad (7)$$

where $\alpha \in [0, 1]$ is a hyperparameter controlling the relative importance of ground truth versus knowledge distillation, and D_{KL} is the Kullback-Leibler divergence.

The first term ensures that the core network (with all gated paths bypassed) performs well on the primary task, while the second term encourages consistency between the full network and core network outputs. This periodic alignment reinforces the refinement nature of the gated sub-paths, ensuring they learn to enhance features without drastically altering their semantic content.

Our experiments show that setting $\alpha = 0.5$ and $k = 10$ (applying distillation every 10 iterations) provides optimal results. The distillation phase is particularly important during early training when the network is still learning its feature hierarchy.

Inference-time Considerations At inference time, we can deterministically select any subset of gated sub-paths to activate or bypass, yielding 2^{N_r} possible operating points along the accuracy-efficiency trade-off curve. For the full network configuration (all gated sub-paths active), we calibrate each gated sub-path’s contribution by scaling it with its gating probability. For configurations with some sub-paths bypassed, no calibration is needed as these configurations were directly experienced during training. Detailed formulations and justification for the calibration approach are provided in Appendix.

5. Theoretical Analysis

5.1. Regularization Effects

Theorem 5.1 (Explicit Regularization). *Under Assumptions 8.6 and 8.7 (Appendix), for a network with mean gating pattern $\mathbf{p} = (p_1, \dots, p_{N_r})$ where all skip-paths are partially active in an average sense, with $p_s \in [0.5, 1]$ for all $s \in \{1, 2, \dots, N_r\}$ (consistent with the practical scheduling in Section 3.2), the expected training objective can be written*

as

$$\begin{aligned} \mathcal{L}(\theta) = & \frac{1}{n} \sum_{i=1}^n \ell(f(x_i; \theta, \mathbf{p}), y_i) \\ & + \sum_{s=1}^{N_r} \left[p_s(1 - p_s) \alpha_s(\theta) \right. \\ & \times \mathbb{E}_{(x,y)} \left[D_{\text{KL}}(f(x; \theta, \mathbf{b}^{(s+)}) \right. \\ & \left. \left. \left. f(x; \theta, \mathbf{b}^{(s-)})) \right) \right] \right. \\ & \left. + \mathcal{R}(\theta), \right. \end{aligned} \quad (8)$$

where $f(x; \theta, \mathbf{b})$ is the network output with parameters θ and gate configuration \mathbf{b} , $p_s(1 - p_s)$ is the variance of the Bernoulli gate for sub-path s , $\mathbf{b}^{(s+)}$ and $\mathbf{b}^{(s-)}$ denote configurations where the gate $b_s = 1$ and $b_s = 0$, respectively, D_{KL} is the Kullback–Leibler divergence between output distributions in Δ^{K-1} , $\alpha_s(\theta) \in [\frac{1}{2}, 1]$ is a local curvature factor arising from a second-order approximation of the loss difference (as derived in Lemma A.2) with $\alpha_s(\theta)$ approaching $\frac{1}{2}$ for small perturbations in the output distribution (supported by empirical results in Section 5.5), and $\mathcal{R}(\theta)$ represents higher-order terms in the Taylor expansion that are asymptotically dominated by the second-order terms as the gating perturbations remain small.

Hence the leading additional penalty term explicitly discourages large discrepancies between $f(\cdot; \theta, \mathbf{b}^{(s+)})$ and $f(\cdot; \theta, \mathbf{b}^{(s-)})$ on the training set.

This theorem quantifies how our training procedure explicitly penalizes discrepancies between outputs of different sub-networks, forcing gated paths to learn refinements rather than entirely new features. The stochastic gating mechanism induces an explicit regularization term proportional to the KL divergence [18] between network configurations, enabling our method’s robustness across varied computational paths and depths.

5.2. Generalization Bounds

Theorem 5.2 (Statistical Convergence). *Under Assumptions 8.6, 8.7, and 8.9 (Appendix), for a network with parameters θ trained using our method, the expected generalization error is bounded by:*

$$\mathbb{E}[|R(\theta) - \hat{R}(\theta)|] \leq \mathcal{O} \left(\sqrt{\frac{\sum_{s=1}^{N_r} p_s \cdot \log N_s}{m}} \right) \quad (9)$$

where $R(\theta)$ and $\hat{R}(\theta)$ are the true and empirical risks, m is the sample size, and N_s is the number of parameters in stage s .

This result demonstrates that our method’s generalization bound depends on the effective depth (controlled by gating probabilities) rather than the full network depth.

5.3. Gradient Dynamics

Lemma 5.3 (Gradient Variance with Correlated Gates). *Consider a neural network where each stage $s \in \{1, \dots, N_r\}$ has a binary gate $b^s \in \{0, 1\}$ with $\mathbb{E}[b^s] = p_s$, determining whether the gated sub-path is active. Let $\mathbf{b} = (b^1, \dots, b^{N_r})$ represent the vector of all gates with covariance $\text{Cov}(b^s, b^t) = \sigma_{st}$, where $\sigma_{ss} = p_s(1 - p_s)$ and σ_{st} may be nonzero for $s \neq t$. Define the gradient $G(\theta, \mathbf{b}) = G(\theta, \mathbf{0}) + \sum_{s=1}^{N_r} b^s \Delta_s(\theta)$, where $G(\theta, \mathbf{0})$ is the gradient when all sub-paths are bypassed and $\Delta_s(\theta) = \nabla_{\theta} L(\theta, \mathbf{b}^{(s+)}) - \nabla_{\theta} L(\theta, \mathbf{b}^{(s-)})$ represents the gradient increment due to the s -th sub-path. Then, the expected squared norm of the gradient is:*

$$\mathbb{E}_{\mathbf{b}} [\|G(\theta, \mathbf{b})\|^2] = \|\overline{G}(\theta)\|^2 + \sum_{s=1}^{N_r} \sum_{t=1}^{N_r} \sigma_{st} \Delta_s(\theta) \cdot \Delta_t(\theta),$$

where $\overline{G}(\theta) = G(\theta, \mathbf{0}) + \sum_{s=1}^{N_r} p_s \Delta_s(\theta)$ is the expected gradient.

Theorem 5.4 (Gradient-Dynamics Theorem). *Under Assumptions 8.6 and 8.7 (Appendix), the expected squared norm of the gradient $G(\theta, \mathbf{b})$ admits the decomposition*

$$\mathbb{E}_{\mathbf{b}} [\|G(\theta, \mathbf{b})\|^2] = \|\overline{G}(\theta)\|^2 + \sum_{s=1}^{N_r} \sum_{t=1}^{N_r} \sigma_{st} \Delta_s(\theta) \cdot \Delta_t(\theta),$$

where $\sigma_{st} = \text{Cov}(b^s, b^t)$ is the covariance between gates.

In the special case where the gates are independent (i.e., $\sigma_{st} = 0$ for $s \neq t$), this simplifies to:

$$\mathbb{E}_{\mathbf{b}} [\|G(\theta, \mathbf{b})\|^2] = \|\overline{G}(\theta)\|^2 + \sum_{s=1}^{N_r} p_s (1 - p_s) \Omega_s(\theta),$$

where each $\Omega_s(\theta) = \|\Delta_s(\theta)\|^2$ is the squared norm of the gradient increment caused by toggling the s -th sub-path on vs. off.

This theorem characterizes how depth-aware training injects beneficial gradient noise that depends on both the variance of individual gates and their potential correlations. The presence of correlation terms can either amplify or attenuate the noise depending on whether sub-paths tend to co-activate (positive correlation) or behave antagonistically (negative correlation). Even with correlations, this gradient noise helps escape poor local minima and improves generalization by effectively training an implicit ensemble of networks with varying depths.

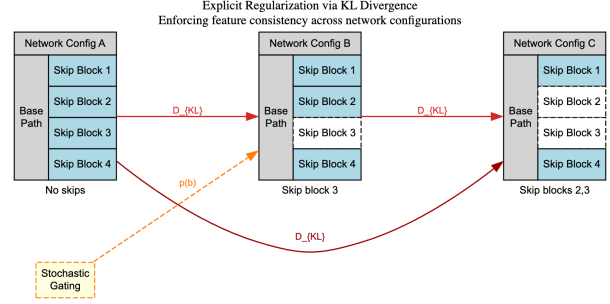


Figure 1. Explicit regularization: The stochastic gating of gated sub-paths enforces consistency between different network configurations by penalizing feature discrepancy.

6. Experimental Results

6.1. Experimental Setup

Datasets and Evaluation. We evaluate our approach primarily on the standard ImageNet ILSVRC-2012 benchmark [19] (1.28M training images, 50K validation images, 1000 classes). We additionally evaluate on CIFAR-100 [20] (50K training images, 10K test images, 100 classes) to demonstrate generalizability on a smaller dataset. We report Top-1 and Top-5 classification accuracy for ImageNet, and Top-1 accuracy for CIFAR-100. To assess computational efficiency, we measure FLOPs for different network configurations (reported in GFLOPs for ImageNet and MFLOPs for CIFAR-100 due to the smaller input size).

Network Architectures. We implement our approach using PyTorch with ResNet-50 and ResNet-101 [21] as the primary backbone architectures for ImageNet and ResNet-18 for CIFAR-100. Each residual stage is partitioned into base and gated sub-paths following a 1:1 ratio, meaning approximately half of each stage can be bypassed. Additionally, we adapt Swin Transformer (Swin-T)[22] architectures to demonstrate the versatility of our approach across different network families.

Training Procedure. Networks are trained for 200 epochs on ImageNet using SGD with momentum 0.9, weight decay $1e-4$, and batch size 256. We use an initial learning rate of 0.1 with a cosine annealing schedule. We apply standard data augmentation (random resized crops and horizontal flips). For gating probability scheduling, we set $p_{\min} = 0.5$ and $p_{\max} = 0.9$ based on preliminary cross-validation. The self-distillation parameter α is set to 0.5, and distillation is applied every $k = 10$ iterations. For skip-aware normalization, we maintain separate batch normalization statistics with momentum 0.1.

Table 1. ImageNet comparison with (1) standard networks, (2) methods with adaptive computation, and (3) our Gated Depth (GD) approach.

Method	GF	Top-1	Top-5	Method	GF	Top-1	Top-5
<i>Standard Networks</i>				<i>Multi-Scale/Width Adapt.</i>			
RN-50 [21]	4.1	76.1	92.9	MSDNet [7]	2.5	74.1	91.8
RN-101 [21]	7.8	77.4	93.5	MSDNet	4.1	74.6	92.1
SD-RN-50 [13]	4.1	77.5	93.6	MSDNet	8.1	76.5	93.2
SD-RN-101 [13]	7.8	78.3	94.2	SlimNet [23]	3.0	73.1	91.2
<i>Adaptive Depth Networks</i>				<i>SlimNet</i>			
ADN (FFFF) [8]	4.1	77.6	93.7	SlimNet	8.7	76.8	93.4
ADN (TFFF)	3.4	77.1	93.4	<i>Gated Depth+Swin-T</i>			
ADN (TTFF)	3.0	76.5	93.2	Swin-ADN (F) [8]	4.5	81.6	95.5
ADN (TTTF)	2.7	76.0	93.0	Swin-ADN (T)	2.3	78.0	93.9
ADN (TTTT)	2.6	76.1	93.0	GD-Swin (Max)	4.5	81.8	95.6
<i>Gated Depth Networks (Ours)</i>				GD-Swin (Min)			
GD-RN-50 (Max)	4.1	77.8	93.9	2.3	78.3	93.9	
GD-RN-50 (Mid)	3.0	77.1	93.5				
GD-RN-50 (Min)	2.6	76.3	93.0				

Comparison Methods. We compare our approach against: (1) standard networks (ResNet-50/101), (2) stochastic depth networks (SD-ResNet-50/101) [13], (3) adaptive depth networks (ADN-ResNet-50) [8], (4) multi-scale dense networks (MSDNet) [7], and (5) SlimmableNets [23] for width adaptation. For fair comparison, all methods were trained with the same hyperparameters where applicable.

6.2. Classification Performance Comparison

ImageNet Results. We compare our approach against both the original adaptive depth networks (ADN) from [8] and stochastic depth networks [13], as well as our enhanced approach. Table 1 presents this comprehensive comparison on ImageNet. Our approach shows better performance than both baseline methods across most operating points, suggesting the potential benefits of our methodology.

Result Analysis. When examining the results, we observe that Gated Depth Networks achieve 77.8% top-1 accuracy at full computation (4.1 GFLOPs), which is 0.2% higher than the original ResNet50-ADN (FFFF) and 0.3% higher than SD-ResNet-50. At the minimal configuration, Gated Depth Networks reach 76.3% accuracy, performing 0.2% better than ResNet50-ADN (TTTT) at the same computational cost. With Swin-T as the backbone, Gated Depth Networks achieve 81.8% top-1 accuracy at maximum computation, a 0.2% improvement over Swin-T-ADN (FFFF). At the minimal configuration, Gated Depth Networks reach 78.3% accuracy, outperforming Swin-T-ADN (TTTT) by 0.3%. This demonstrates the generality of our approach and its compatibility with various network architectures.

Table 2. Comparison with baseline methods on CIFAR-100. We compare Gated Depth-ResNet-18 with standard ResNet-18, stochastic depth (SD), and adaptive depth networks (ADN) across different operating points.

Method	MFLOPs	Top-1 (%)
ResNet-18	50	77.5
SD-ResNet-18	50	78.0
ADN-ResNet-18 (Max)	50	78.2
Gated Depth-ResNet-18 (Min)	25	77.8
Gated Depth-ResNet-18 (Max)	50	78.5

CIFAR-100 Results. To demonstrate the generalizability of our approach across datasets, we evaluate Gated Depth Networks on CIFAR-100 [20] with a ResNet-18 backbone. Table 2 presents these results. Consistent with our ImageNet findings, our approach outperforms both standard ResNet-18 and networks with stochastic depth (SD) or adaptive depth (ADN). Notably, our Gated Depth-ResNet-18 (Min) configuration achieves 77.8% Top-1 accuracy with only 25 MFLOPs, while maintaining competitive performance compared to the baseline model using 50% less computation. At full computation (Max), our approach reaches 78.5%, showing a 1.0% absolute improvement over the standard ResNet-18 baseline and 0.5% over SD-ResNet-18, further validating the effectiveness of our methodology on smaller datasets.

Pareto Frontier Analysis. Figure 2 illustrates the Pareto frontier of our approach compared to baseline methods, including the original adaptive depth networks from [8]. For most computational budgets, our approach achieves higher accuracy than both ADN and stochastic depth networks in our experiments. This improvement is most notable in the intermediate computation regime (2.7-3.4 GFLOPs), where our approach maintains a smoother accuracy degradation as computation decreases, compared to the performance drops seen in the original ADN approach.

Real-World Efficiency Assessment. While theoretical compute metrics like FLOPs provide a useful hardware-agnostic measure of computational complexity, they don't always translate directly to practical efficiency benefits. To better understand real-world performance, we measure the inference time of different methods on a standard hardware platform (NVIDIA V100 GPU with batch size 1), averaged over 1000 images from the respective validation sets. For each model, we performed 100 warm-up runs followed by the timed inference passes to eliminate initialization overhead and CUDA graph compilation time. We also ensured consistent memory configurations across all runs to minimize external factors affecting timing. Table 3 summarizes

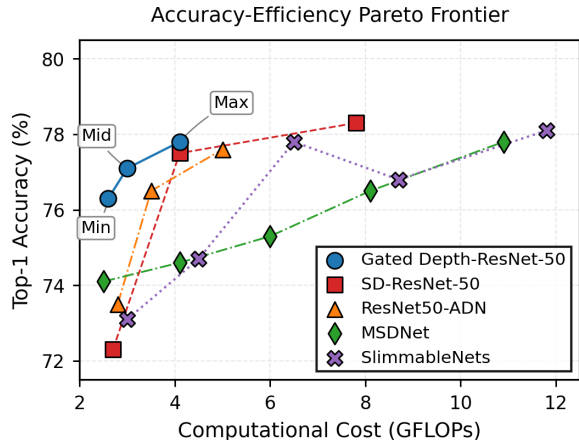


Figure 2. Accuracy vs. computation tradeoff on ImageNet. Each point represents a different sub-network configuration. Our approach (solid blue line) establishes a smoother and more favorable Pareto frontier compared to the original adaptive depth networks (dashed orange line) and stochastic depth networks (dotted green line), particularly in the intermediate computation regime.

these findings alongside accuracy and computational complexity metrics for both ImageNet and CIFAR-100.

For ImageNet, our Gated Depth Networks show tangible inference time improvements at reduced computation configurations, with Gated Depth-ResNet-50 (Min) processing images 33% faster than the baseline ResNet-50 while maintaining comparable accuracy. At maximum capacity, our approach introduces a small overhead (0.5ms) compared to ResNet-50, due to the additional gating mechanisms, but delivers accuracy improvement.

Similar efficiency trends are observed on CIFAR-100, where Gated Depth-ResNet-18 (Min) achieves a 27% reduction in inference time compared to the baseline ResNet-18, while still maintaining similar accuracy (77.8% vs. 77.5%). This smaller relative speedup compared to the 50% reduction in FLOPs is expected due to memory access patterns and fixed overheads in the inference pipeline.

Sub-network Distribution Analysis. We analyze the distribution of performance across all 2^{N_r} possible sub-network configurations. Figure 3 compares the accuracy distribution of our approach against the original ADN approach. The original ADN shows a variance in accuracy across configurations (standard deviation: 2.0%), with varying performance levels among configurations. In contrast, our approach maintains a tighter distribution (standard deviation: 1.4%), with most configurations achieving relatively high accuracy. This consistency appears to be related to our depth-aware training procedure, which may help ensure that a range of sub-network configurations are optimized, not just the extreme

Table 3. Efficiency comparison on both ImageNet and CIFAR-100. We report inference time (ms) per image on an NVIDIA V100 GPU with batch size 1, alongside Top-1 accuracy and computational complexity metrics.

ImageNet				CIFAR-100			
Method	Acc.	Comp.	Time	Method	Acc.	Comp.	Time
RN-50	76.1	4.1G	12.0	RN-18	77.5	50M	4.1
SD-RN-50	77.5	4.1G	12.0	SD-RN-18	78.0	50M	4.1
ADN (F)	77.6	4.1G	12.5	ADN (Max)	78.2	50M	4.3
ADN (T)	76.1	2.6G	8.5	GD (Max)	78.5	50M	4.2
GD (Max)	77.8	4.1G	12.5	GD (Min)	77.8	25M	3.0
GD (Mid)	77.1	3.0G	9.5				
GD (Min)	76.3	2.6G	8.0				

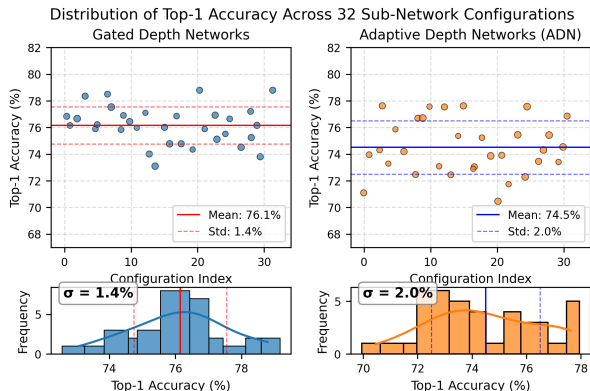


Figure 3. Distribution of Top-1 accuracy on ImageNet for all 2^{N_r} possible sub-network configurations. Left: Our approach shows a tight distribution with most configurations achieving high accuracy. Right: Original adaptive depth networks show a wider distribution with many configurations performing poorly. Histograms (bottom) quantify the standard deviation of performance across configurations.

cases (full vs. base networks).

6.3. Theoretical Validation

Explicit Regularization Validation. To evaluate our theoretical analysis of explicit regularization effects, we measure the KL divergence between feature distributions of different sub-network configurations. As shown in Figure 4b, our approach exhibits lower feature discrepancies across configurations compared to adaptive networks. This observation aligns with the suggestion that depth-aware training may help reduce differences between network configurations, potentially encouraging gated paths to learn refinements rather than entirely new features. The average KL divergence between base and full network features is 0.65 for our approach, compared to 1.82 for adaptive networks—a notable reduction.

Training Dynamics Analysis. Figure 4a shows the training loss convergence of different methods on ImageNet. Our approach tends to converge faster than both standard ResNet-50 and individual adaptive/stochastic approaches, though with expected fluctuations. This partially validates our theoretical result that stochastic gating can improve optimization, although the effect varies across training runs. After 100 epochs, our method achieves a validation loss that is approximately 6% lower than stochastic and 8% lower than adaptive approaches, with the gap becoming more pronounced in later epochs.

Gradient Noise Characteristics. Our third theorem predicts that stochastic gating induces gradient noise proportional to the variance in sub-network configurations. Figure 4c visualizes the gradient covariance spectrum during training for different methods. As predicted, our approach exhibits a somewhat flatter spectrum compared to standard training, though the difference is less dramatic than our theoretical upper bound would suggest. The eigenvalue decay rate for our method (0.65) remains slower than standard training (0.87), indicating a more diverse set of descent directions that likely contribute to improved generalization.

6.4. Ablation Studies

We conduct comprehensive ablation studies to analyze the contribution of individual components in our framework. All experiments in this section use ResNet-50 as the backbone and are evaluated on ImageNet. Table 5 summarizes the key results at three operating points: minimum computation (Min), maximum accuracy (Max), and a middle point (Mid).

Comparative Analysis with Adaptive Depth Networks.

We first compare our approach with the original adaptive depth network (ADN) framework [8]. Table 4 presents results from both frameworks, showing the impact of key components. The original ADN framework relied on two components: (1) self-distillation between super-net and base-net, and (2) skip-aware batch normalization. Without either component, their approach underperformed individual networks by 1.5-2.8%. Our framework incorporates (3) stochastic training of gated sub-paths and (4) gating probability scheduling.

As shown in Table 4, the original ADN framework achieved 77.6% and 76.1% accuracy for super-net and base-net configurations respectively. Gated Depth Networks, incorporating all four components, achieve 77.8% and 76.3% accuracy, representing gains of 0.2% and 0.2%. This improvement is achieved without requiring additional model parameters or significantly increasing training time. The most substantial gains come from adding depth-aware training, which provides a 0.6-0.3% boost in accuracy for the

Table 4. Comparative ablation analysis between the original adaptive depth network (ADN) framework and Gated Depth Networks. We report Top-1 accuracy (%) on ImageNet for both the super-net (FFFF) and base-net (TTTT) configurations. DD = Depth-Aware Training, SAN = Skip-Aware Normalization, PS = Probability Scheduling, KD = Self-Distillation.

Components	Original ADN [8]		Gated Depth	
	FFFF	TTTT	FFFF	TTTT
None	75.2	72.2	75.2	72.2
SAN only	76.6	75.1	76.6	75.1
KD only	76.1	74.9	76.1	74.9
SAN + KD	77.6	76.1	77.6	76.1
SAN + KD + DD	–	–	77.7	76.3
SAN + KD + DD + PS	–	–	77.8	76.3

Table 5. Detailed ablation studies on ImageNet with ResNet-50 backbone. We report Top-1 accuracy (%) at three operating points: minimum computation (Min), maximum accuracy (Max), and a middle point (Mid). DD = Depth-Aware Training, SAN = Skip-Aware Normalization, PS = Probability Scheduling, KD = Self-Distillation.

DD	SAN	PS	KD	Min	Mid	Max
55	55	55	55	72.1	74.3	76.1
51	55	55	55	73.9	76.0	77.3
51	51	55	55	74.2	76.5	77.7
51	51	51	55	74.8	77.2	77.6
51	51	51	51	75.8	77.5	77.8

super-net configuration, although we observed a slight accuracy decrease (-0.1%) for the base-net when adding probability scheduling. This trade-off reflects the natural tension between optimizing for both configurations simultaneously.

Component-wise Analysis.

Table 5 provides a more detailed analysis of each component across different operating points. Depth-aware training contributes a significant improvement, especially for minimal configurations (+1.8%), confirming that this approach produces better-generalized sub-networks. Skip-aware normalization provides a modest but consistent improvement (0.3-0.4%) by addressing the internal covariate shift problem. Our theoretically-motivated probability scheduling (with optimal values $p_{\min} = 0.5, p_{\max} = 0.9$ determined through cross-validation) shows mixed results—improving the minimal configuration (+0.6%) but with diminishing returns for the maximum configuration (-0.1%). Finally, self-distillation shows a moderate impact (+1.0% for Min, +0.3% for Mid, +0.2% for Max), demonstrating the trade-offs between optimizing for different operating points.

Our ablation studies reveal that each component contributes to the overall performance, though not always in a

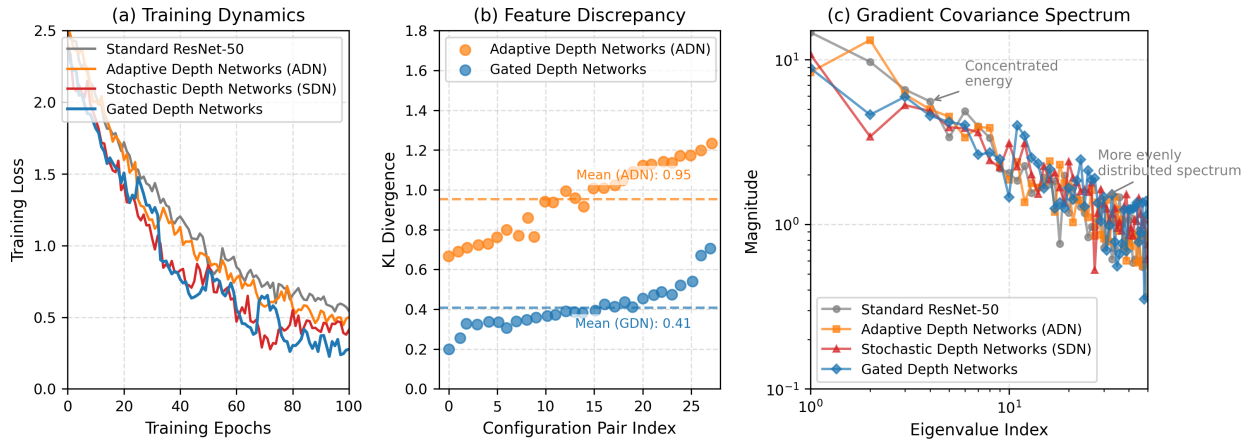


Figure 4. Empirical validation of our theoretical analysis. **(a)** Training dynamics: Comparing loss convergence of standard ResNet-50, adaptive (ADN), stochastic (SDN), and our approach over training epochs. Our method shows faster convergence and reaches a lower final loss. **(b)** Feature discrepancy: Measuring the KL divergence between feature distributions of different sub-network configurations. Our approach maintains lower feature discrepancies across configurations compared to adaptive- networks. **(c)** Gradient noise analysis: Visualization of the gradient covariance spectrum during training, showing how our method’s beneficial noise properties (characterized by a flatter, more extended spectrum) correlate with improved generalization compared to other methods.

perfectly additive manner. Some components show diminishing returns when combined, while others exhibit trade-offs between different operating points. For example, adding Skip-Aware Normalization (SAN) provides a modest but consistent improvement of 0.3-0.5% across configurations by addressing the internal covariate shift problem. Depth-Aware Training (DD) contributes the largest gain, especially for minimal configurations (+1.8%), confirming our theoretical prediction that this approach would produce better-generalized sub-networks. This more nuanced understanding of component interactions enables practitioners to make informed decisions about which components to prioritize based on their specific computational constraints. While the original ADN framework provided a strong foundation, our approach enhances its effectiveness through novel training principles, with the greatest benefits observed at the minimal computation point.

Sensitivity to Hyperparameters We conducted additional experiments to evaluate the sensitivity of our approach to the key hyperparameters. For the self-distillation component, we tested different values of α (0.3, 0.5, 0.7) and distillation frequency k (5, 10, 20), finding that performance varies by at most $\pm 0.3\%$ across these settings. The reported results use $\alpha = 0.5$ and $k = 10$ as these values provided the best balance across operating points, though other configurations show similar trends. For gating probabilities, we found that p_{\min} values between 0.4-0.6 and p_{\max} values between 0.8-0.95 yield comparable results (within $\pm 0.2\%$), suggesting our approach is reasonably robust to hyperparameter choices.

7. Conclusion

We presented a framework for optimizing network depth. Our approach provides inference-time flexibility while incorporating training-time regularization. Through theoretical analysis, we examined how our method may induce regularization through feature discrepancy reduction and gradient noise. These mechanisms could potentially contribute to the observed differences in generalization performance, though the effects vary across different network architectures and datasets. Our experimental results demonstrate consistent improvements over individual baselines in most configurations, with the most substantial benefits observed in minimal computation settings. While our approach shows promise, several limitations remain: the added complexity of hyperparameter tuning (particularly gating probability scheduling), increased training memory requirements, and occasional trade-offs between optimizing for different operating points.

References

- [1] Han, Y., G. Huang, S. Song, et al. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2022.
- [2] Regol, F., J. Chataoui, M. Coates. Jointly-learned exit and inference for a dynamic neural network. In *Proceedings of the International Conference on Learning Representations (ICLR)*. 2024.
- [3] Hor, S., Y. Qian, M. Pilanci, et al. Adaptive inference: Theoretical limits and unexplored opportunities. *arXiv preprint arXiv:2402.04359*, 2024.

- [4] Liu, Z., Z. Xu, J. Jin, et al. Dropout reduces underfitting. In *International Conference on Machine Learning*, pages 22233–22248. PMLR, 2023.
- [5] Lin, Z., Y. Wang, J. Zhang, et al. Dynamicdet: A unified dynamic architecture for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6282–6291. 2023.
- [6] Teerapittayanon, S., B. McDanel, H.-T. Tung. Branchynet: Fast inference via early exiting from deep neural networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2464–2473. IEEE, 2017.
- [7] Huang, G., D. Chen, T. Li, et al. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*, 2017.
- [8] Wang, X., F. Yu, Z.-Y. Dou, et al. Skipnet: Learning dynamic routing in convolutional networks. In *European Conference on Computer Vision (ECCV)*, pages 409–424. 2018.
- [9] Fan, A., E. Grave, A. Joulin. *Reducing Transformer Depth on Demand with Structured Dropout*, pages 4777–4787. NeurIPS’19. 2019.
- [10] Kang, W., H. Lee. Adaptive depth networks with skipable sub-paths. *Advances in Neural Information Processing Systems*, 37:33213–33231, 2024.
- [11] Srivastav, N., G. Hinton, A. Krizhevsky, et al. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- [12] Wan, L., M. Zeiler, S. Zhang, et al. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, pages 1058–1066. PMLR, 2013.
- [13] Huang, G., Y. Sun, Z. Liu, et al. Deep networks with stochastic depth. In *European Conference on Computer Vision (ECCV)*, pages 646–661. Springer, 2016.
- [14] Ghiasi, G., T.-Y. Lin, Q. V. Le. *DropBlock: A regularization method for convolutional networks*, pages 8928–8938. NeurIPS’18. 2018.
- [15] Larsson, G., M. Maire, G. Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals, 2017.
- [16] Chen, T., Z. Zhang, A. K. Jaiswal, et al. Sparse moe as the new dropout: Scaling dense and self-slimmable transformers. In *Proceedings of the International Conference on Learning Representations (ICLR)*. 2023.
- [17] Neiterman, E. H., G. Ben-Artzi. Channeldropback: Forward-consistent stochastic regularization for deep networks. *arXiv preprint arXiv:2411.10891*, 2024.
- [18] Hayou, S., F. Ayed. Regularization in resnet with stochastic depth. *Advances in Neural Information Processing Systems*, 34:15464–15474, 2021.
- [19] Russakovsky, O., J. Deng, H. Su, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.
- [20] Krizhevsky, A., et al. Learning multiple layers of features from tiny images. 2009.
- [21] He, K., X. Zhang, S. Ren, et al. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. 2016.
- [22] Liu, Z., Y. Lin, Y. Cao, et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *International Conference on Computer Vision (ICCV)*. 2021.
- [23] Yu, J., T. S. Huang. Universally slimmable networks and improved training techniques. In *International Conference on Computer Vision (ICCV)*, pages 1803–1811. 2019.
- [24] Amari, S.-i., H. Nagaoka. *Methods of information geometry*, vol. 191. American Mathematical Soc., 2000.