Homework 1

Ping-Yeh Chou 113550901

1 Introduction

Image classification [Lu and Weng, 2007; Li *et al.*, 2018; Wang *et al.*, 2019], a fundamental challenge in computer vision, requires models to recognize objects and perform classification across diverse visual scenarios. This homework focuses on training a 100-class classifier using a phased optimization strategy with a Wide ResNet-50-2 [Zagoruyko and Komodakis, 2016; Zhang *et al.*, 2017] backbone. The approach prioritizes progressive adaptation of pre-trained features through sequential training phases while mitigating overfitting via data augmentation [Van Dyk and Meng, 2001]. The training process is specifically structured in two phases. Initially, only the new classification layer is trained with all other layers frozen, enabling rapid adaptation of decision boundaries. In the second phase, all layers are fine-tuned using a compound learning rate schedule to balance exploration and convergence. Ultimately, we achieved a 94% accuracy rate on the unknown test set. Code is available at https://github.com/cloud-zhoubingye/ResNet-Classification.git.

2 Method

2.1 Data Pre-processing

The data pre-processing pipeline is designed to handle 100-class RGB images. During training, images undergo real-time augmentation to enhance model robustness. Each input image is first converted to a tensor and dynamically resized to 224×224 pixels through a two-step process: center-cropping to the minimum dimension of the original image, followed by bilinear interpolation-based resizing to the target resolution.[Shorten and Khoshgoftaar, 2019; Shorten *et al.*, 2021; Maharana *et al.*, 2022]

For the training phase, the augmentation pipeline integrates geometric and photometric transformations to simulate diverse imaging conditions. Geometric perturbations include random affine transformations with rotation ($\pm 10^\circ$), translation ($\pm 10\%$ of image size), scaling (0.9–$1.1\times$), and shearing ($\pm 10^\circ$). Photometric variations are introduced through probabilistic color jittering (brightness: ± 0.25, contrast: ± 0.25, saturation: ± 0.25, hue: ± 0.1) and Gaussian blurring (kernel size 3, $\sigma = 0.1$–2.0), each applied with 50% probability. Random horizontal and vertical flips further expand pose variation, both activated with 30% probability.

Validation and testing phases employ a simplified preprocessing sequence, eliminating stochastic transformations to ensure reproducibility and stability of results. Images undergo identical resizing and center-cropping as training data, followed by normalization using ImageNet-derived statistics (mean$= [0.485, 0.456, 0.406]$, std$= [0.229, 0.224, 0.225]$). All tensor operations are executed on GPU hardware to accelerate processing, with images batched into groups of 72 samples during data loading. The pipeline maintains separate workers for training (4 threads on Unix systems) and validation/test sets, ensuring efficient I/O parallelism while preventing resource contention across phases.

2.2 Model Architecture

The proposed framework utilizes a Wide ResNet-50-2 backbone, a variant of the residual network [He *et al.*, 2016] family that emphasizes channel width expansion over extreme depth scaling. This architecture inherits the residual learning principle $F(x) = \mathcal{H}(x) + x$ while modifying the bottleneck blocks to double base channel dimensions compared to standard ResNet-50. Initial layers process input images through a 7×7 convolution with stride 2, followed by batch normalization and max-pooling to reduce spatial resolution to 56×56. Four subsequent residual stages employ widened bottleneck blocks, where the critical 3×3 convolutional layers operate on channels expanded by a factor of $4\times$ relative to the original ResNet design.

The core distinction between Wide ResNet and conventional ResNet lies in their trade-off between width and depth. Where ResNet-50 stacks 50 layers with base channel widths of 64–512, Wide ResNet-50-2 maintains identical depth but doubles channel counts (e.g., 128–1024) in intermediate layers. This width enhancement amplifies feature diversity while retaining the parametric efficiency of bottleneck structures: each 1×1–3×3–1×1 block first compresses channels, then applies wide convolutions, and finally restores dimensionality. The widened 3×3 convolution with $4\times$ expanded channels ($\mathcal{W}_2$ in Eq. 1) becomes the primary capacity booster. This design reduces gradient attenuation risks compared to ultra-deep networks, as wider layers facilitate smoother gradient flow during backpropagation.

The network terminates with global average pooling and a 100-unit fully connected layer initialized via He normal distribution. All convolutional weights transfer from ImageNet pre-training except the final classifier, which is trained from scratch.

2.3 Training Strategy

The training process adopts a two-phase progressive optimization strategy [Lörincz and Buzsáki, 2000] to balance feature preservation and model adaptation. Phase I focuses on adapting the classification head while freezing pre-trained backbone parameters, initialized with 10 epochs of training using AdamW optimizer under 16-bit mixed precision. This phase establishes task-specific decision boundaries without destabilizing the ImageNet-pretrained feature extractors.

Algorithm 1 Two-Phase Training Process

Input: Training data $\mathcal{D}_{\text{train}}$, Validation data $\mathcal{D}_{\text{val}}$, Pretrained Wide ResNet-50-2 model M_0.
Output: Fine-tuned model M_{final} with checkpoints.

Phase 1: FC Layer Adaptation
Load M_0 weights excluding FC layer
Freeze all layers via `freeze_layers`(M_0)
Initialize FC layer with normal distribution
Set optimizer: AdamW (lr $= 5\times10^{-4}$, wd $= 10^{-4}$)
for epoch $= 1$ **to** 10 **do**
    Load batch $\mathcal{B}$ from $\mathcal{D}_{\text{train}}$ with augmentations
    Compute loss $\mathcal{L} = \text{CE}(M_0(\mathcal{B}_{\text{img}}), \mathcal{B}_{\text{label}})$
    Update FC layer weights via $\nabla\mathcal{L}$
end for
Save M_0 as M_{phase1}
Phase 2: Full Network Fine-tuning
Unfreeze all layers via `unfreeze_layers`(M_{phase1})
Set optimizer: NAdam (lr $= 5\times10^{-4}$, wd $= 10^{-4}$)
Configure scheduler: Linear warmup (5 epochs) $\rightarrow$ Cosine annealing (45 epochs)
for epoch $= 1$ **to** 125 **do**
    Adjust learning rate η_t per scheduling policy
    Clip gradients $\|\nabla\mathcal{L}\|_2 \leq 0.8$
    Train on $\mathcal{D}_{\text{train}}$ with mixed precision (FP16)
    if epoch $\mod 15 = 0$ **then**
        Save checkpoint M_{ckpt} with val_acc $\geq 82\%$
    end if
end for
return Best checkpoint M_{final}

Phase II unlocks the entire network through for full fine-tuning. A composite learning rate schedule combines linear warmup and cosine annealing, mitigating gradient instability during early fine-tuning. The optimizer is NAdam with weight decay and gradient clipping, preventing norm explosion. The pseudocode for our training process is shown in Algorithm 1.

Additionally, we have detailed the hyperparameter settings used in the project in Table 1.

3 Result

Our method achieves 90.31% classification accuracy on the validation set, with notably superior performance of 94% on the unknown test set. This accuracy discrepancy may arise from potential distributional simplicity of test samples compared to validation data.

Training dynamics visualized in Figure 1 demonstrate rapid convergence, where training losses decays within the first 10 epochs in phrase 1, gradually decreasing and stabilizing to plateaus in phrase 2. Correspondingly, in Figure 2, the progression of validation accuracy shows consistent improvement, ultimately attaining 90.31% in final training stage.

The normalized confusion matrix [Townsend, 1971] (Figure 5) reveals balanced discriminative capability across all 100

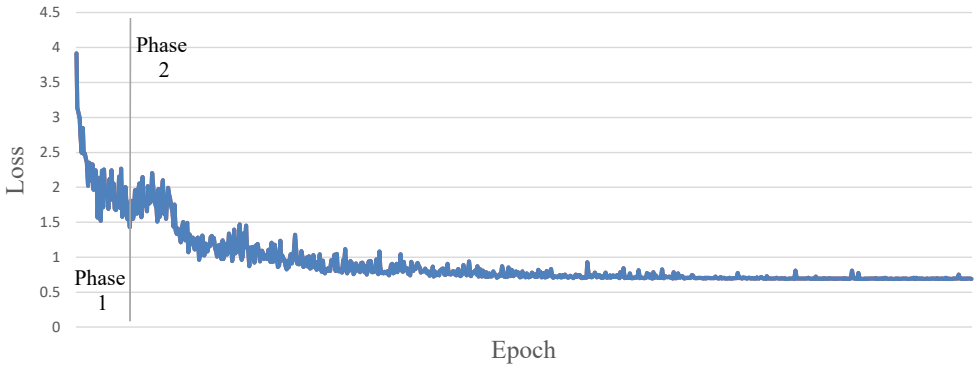

Figure 1: Training loss trajectories across epochs.

Category	Parameter	Value	Phase	Description
Training	Batch Size	72	Both	Training samples per iteration
	Phase 1 Epochs	10	I	Frozen backbone training
	Phase 2 Epochs	125	II	Full network fine-tuning
	Mixed Precision	16-bit	Both	FP16 acceleration
	Gradient Clip	0.8	II	L2 norm threshold
	Checkpoint Interval	15	II	Epochs between saves
Optimizer	Phase 1 Optimizer	AdamW	I	With weight decay
	Phase 1 Learning Rate	5×10^{-4}	I	Fixed schedule
	Phase 2 Optimizer	NAdam	II	With momentum
	Phase 2 Learning Rate	$5\times10^{-4}\rightarrow5\times10^{-6}$	II	Warmup + Cosine
Augmentation	Rotation Range	$\pm10^{\circ}$	Train	Random affine
	Translation	$\pm10\%$	Train	Horizontal/Vertical
	Scale Range	0.9–1.1	Train	Image scaling factor
	Shear Range	$\pm10^{\circ}$	Train	Shear deformation
	Color Jitter	0.25	Train	Brightness/Contrast/Saturation
	Hue Jitter	0.1	Train	Hue variation range
	Gaussian Blur σ	0.1–2.0	Train	Kernel intensity
Model	Input Resolution	224×224	Both	Resized crop
	Pretrained Weights	ImageNet	Both	Backbone initialization
	Num Classes	100	Both	Classification heads

Table 1: Hyperparameter configurations for the two-phase training pipeline.

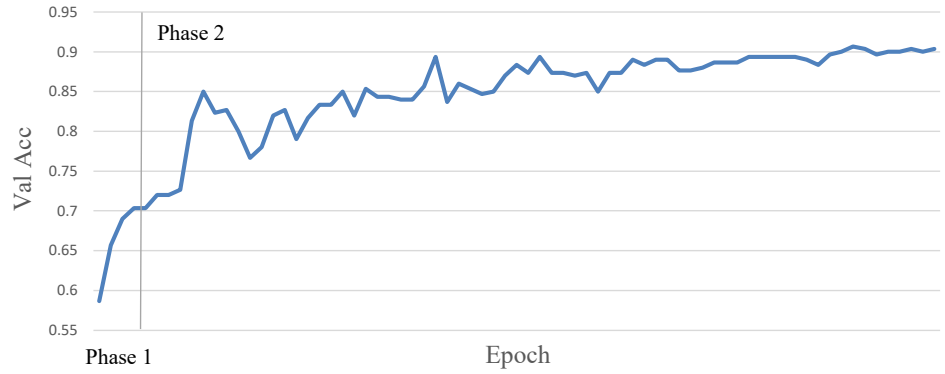

Figure 2: Evolution of validation set accuracy throughout the training epochs.

classes, suggesting that the phased optimization strategy and wide resnet model enhances generalization beyond validation benchmarks.

4 Additional Experiments

We employed the same training pipeline and experimental settings for five networks—ResNet-34, ResNet-50, ResNet-101, ResNeXt-50 [Xie *et al.*, 2017], and Wide ResNet-50—and observed varying accuracy rates on the test set, as illustrated in Figure 3. This analysis reveals three key trends. First, simply depth scaling (ResNet-34 to ResNet-101) for our specific task results in diminishing returns. The 44.5M-parameter ResNet-101 achieves nearly the same accuracy as ResNet-50, despite a 73.8% increase in parameters. Second, cardinality scaling through ResNeXt-50's grouped convolutions attains a 92.0% accuracy with a similar number of parameters as ResNet-50, demonstrating superior efficiency. Third, width scaling with Wide ResNet-50 delivers the highest performance by expanding channel dimensions, which improves gradient flow and feature reuse. This suggests that widening networks is more effective at alleviating information bottlenecks than simply increasing depth or cardinality.

Another ablation experiment, Figure 4, reveals NAdam [Dozat, 2016]'s dominance over standard optimizers, achieving highest accuracy. The improvement can be attributed to the anticipatory updates from Nesterov momentum, which reduce overshooting in loss landscapes, adaptive learning rates that prevent aggressive parameter updates in high-gradient regions, and

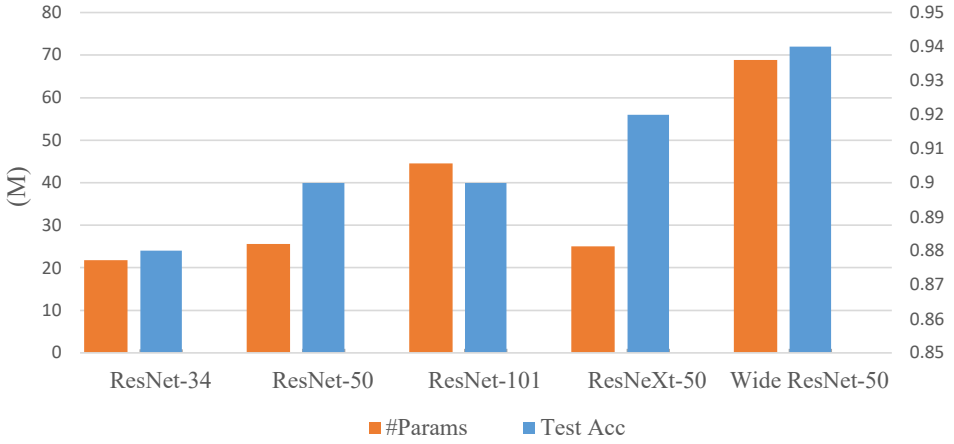

Figure 3: Comparations of network architectures (ResNet-34/50/101, ResNeXt-50, Wide ResNet-50).

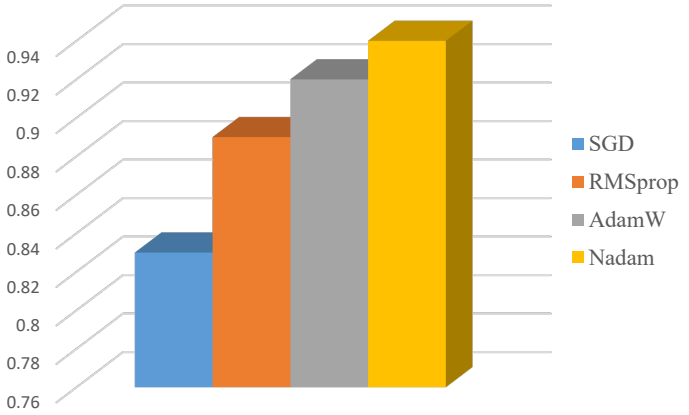

Figure 4: Optimizer dynamics comparisons.

momentum decay scheduling that stabilizes late-stage training. While AdamW [Zhou *et al.*, 2024] addresses Adam [Kingma and Ba, 2014]'s weight decay misalignment, it lacks NAdam's dual mechanisms for escaping local minima, a critical advantage when training wide networks with complex loss surfaces.

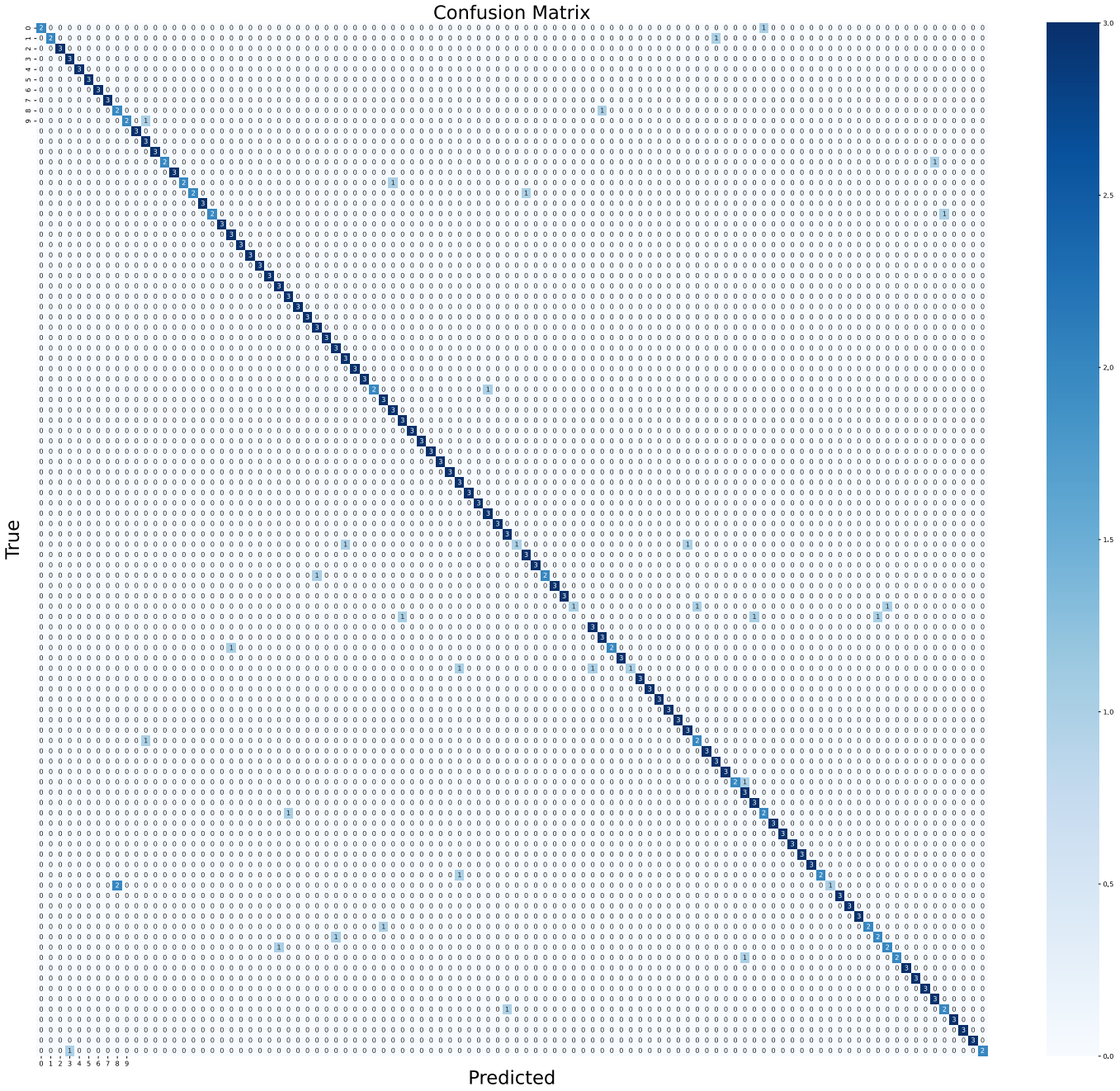

Figure 5: Confusion matrix depicting predicted versus true class relationships across all 100 categories.

References

[Dozat, 2016] Timothy Dozat. Incorporating nesterov momentum into adam. 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Li *et al.*, 2018] Ying Li, Haokui Zhang, Xizhe Xue, Yenan Jiang, and Qiang Shen. Deep learning for remote sensing image classification: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(6):e1264, 2018.

[Lörincz and Buzsáki, 2000] András Lörincz and György Buzsáki. Two-phase computational model training long-term memories in the entorhinal-hippocampal region. *Annals of the New York Academy of Sciences*, 911(1):83–111, 2000.

[Lu and Weng, 2007] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[Maharana *et al.*, 2022] Kiran Maharana, Surajit Mondal, and Bhushankumar Nemade. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, 3(1):91–99, 2022.

[Shorten and Khoshgoftaar, 2019] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[Shorten *et al.*, 2021] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8(1):101, 2021.

[Townsend, 1971] James T Townsend. Theoretical analysis of an alphabetic confusion matrix. *Perception & Psychophysics*, 9:40–50, 1971.

[Van Dyk and Meng, 2001] David A Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.

[Wang *et al.*, 2019] Wei Wang, Yujing Yang, Xin Wang, Weizheng Wang, and Ji Li. Development of convolutional neural network and its application in image classification: a survey. *Optical Engineering*, 58(4):040901–040901, 2019.

[Xie *et al.*, 2017] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[Zagoruyko and Komodakis, 2016] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[Zhang *et al.*, 2017] Ke Zhang, Miao Sun, Tony X Han, Xingfang Yuan, Liru Guo, and Tao Liu. Residual networks of residual networks: Multilevel residual networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(6):1303–1314, 2017.

[Zhou *et al.*, 2024] Pan Zhou, Xingyu Xie, Zhouchen Lin, and Shuicheng Yan. Towards understanding convergence and generalization of adamw. *IEEE transactions on pattern analysis and machine intelligence*, 2024.