Transfer Learning for CIFAR-10 Image Classification

Prabhupada Samantaray
School of Computer Engineering
KIIT University, Bhubaneswar-751024, Odisha, India
Email: prabhupada.samantaray@kiit.ac.in

Abstract

Transfer learning has emerged as a powerful technique in computer vision, enabling efficient training of deep neural networks with limited computational resources. This paper presents a comprehensive comparative analysis of three prominent convolutional neural network (CNN) architectures—VGG16, ResNet50, and MobileNetV2—applied to the CIFAR-10 image classification task using transfer learning. We employ pre-trained models from ImageNet and fine-tune them with custom classifier heads, evaluating their performance across multiple metrics including accuracy, precision, recall, F1-score, training time, and model complexity. Our experimental results demonstrate that VGG16 achieves the highest test accuracy of 60.74%, followed by ResNet50 at 50.55% and MobileNetV2 at 34.17%. Furthermore, we analyze convergence behavior, per-class performance, and computational efficiency to provide insights into the trade-offs between model complexity and classification performance. The findings contribute to understanding the effectiveness of different architectural designs for small-scale image datasets and provide practical guidance for model selection in resource-constrained scenarios.

Index Terms—Transfer Learning, Convolutional Neural Networks, CIFAR-10, VGG16, ResNet50, MobileNetV2, Image Classification, Deep Learning

I. INTRODUCTION

Deep learning has revolutionized computer vision, with Convolutional Neural Networks (CNNs) achieving state-of-the-art performance across numerous image classification tasks [1]. However, training deep networks from scratch requires substantial computational resources and large labeled datasets, which are often unavailable in practical applications [2]. Transfer learning addresses this limitation by leveraging knowledge learned from large-scale datasets (such as ImageNet) and adapting it to target tasks with limited data [3].

The CIFAR-10 dataset [4], consisting of 60,000 32×32 color images across 10 classes, serves as a standard benchmark for evaluating image classification algorithms. Despite its relatively small image resolution, CIFAR-10 presents significant challenges due to intra-class variability, inter-class similarity, and limited spatial information [5].

A. Motivation

While numerous studies have evaluated CNN architectures on CIFAR-10, most focus on training from scratch or employ architecture-specific optimizations [6][7]. There remains a need for systematic comparison of transfer learning approaches using contemporary architectures with standardized training protocols. This study addresses this gap by evaluating three architectures representing different design philosophies:

- **VGG16** [8]: Deep network with uniform architecture and small filters
- **ResNet50** [9]: Residual connections enabling very deep networks
- **MobileNetV2** [10]: Efficient architecture optimized for mobile deployment

B. Contributions

The primary contributions of this work are:

1. Comprehensive evaluation of three prominent CNN architectures using transfer learning on CIFAR-10
2. Detailed analysis of convergence behavior, per-class performance, and computational efficiency
3. Systematic comparison using multiple evaluation metrics beyond simple accuracy
4. Practical insights into model selection considering accuracy-efficiency trade-offs
5. Reproducible experimental framework with complete implementation details

The remainder of this paper is organized as follows: Section II reviews related work, Section III describes our methodology, Section IV presents experimental results, Section V provides discussion and analysis, and Section VI concludes with future directions.

II. RELATED WORK

A. Convolutional Neural Networks

CNNs have become the dominant approach for image classification since AlexNet's breakthrough in 2012 [11]. The fundamental CNN operation involves convolutional layers that learn hierarchical feature representations:

$$y_{ij}^{k} = f\left(\sum_{m}\sum_{n}\sum_{c} w_{mn}^{kc} \cdot x_{(i+m)(j+n)}^{c} + b^{k}\right)$$

$Where\ y_{ij}^{k} =\ is\ the\ output\ at\ position\ (i,j)\ for\ kernel\ k,$
$w_{mn}^{kc}\ are\ learnable\ weights,$
$xistheinput,$
$f\ is\ a\ non-linear\ activation\ function.$

B. VGG Networks

VGGNet [8] demonstrated that network depth is crucial for performance, using only 3×3 convolutional filters throughout. The VGG16 architecture consists of 16 layers with increasing feature map depths: 64 → 128 → 256 → 512 → 512, followed by fully connected layers. Its uniform design makes it highly interpretable but computationally expensive.

C. Residual Networks

ResNet [9] introduced skip connections to address vanishing gradient problems in very deep networks. The residual block is defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}$$

where $\mathcal{F}$ represents stacked convolutional layers and
the identity mapping $\mathbf{x}$ enables direct gradient flow.
ResNet50 achieves 50 layers through bottleneck blocks with $1\times1, 3\times3,$ and 1×1 convolutions.

D. Mobile Networks

MobileNetV2 [10] employs depthwise separable convolutions and inverted residual blocks to reduce computational cost:

$$\text{Cost}_{\text{standard}} = D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$$

$$\text{Cost}_{\text{depthwise}} = D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F$$

where D_K is kernel size,
M and N are input/output channels,
and D_F is feature map size.
This achieves $8-9\times$ reduction in computation.

E. Transfer Learning

Transfer learning leverages pre-trained models to improve performance on target tasks [3]. The process involves:

1. **Feature Extraction**: Freeze pre-trained layers, train only classifier
2. **Fine-tuning**: Unfreeze some layers for task-specific adaptation
3. **Full Training**: Train entire network with lower learning rate

Our approach employs feature extraction with custom classifier heads, which is most effective for small datasets [12].

III. METHODOLOGY

A. Dataset

CIFAR-10 comprises 50,000 training and 10,000 test images across 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. We split the training set into 40,000 training and 10,000 validation samples using an 80:20 ratio as shown in Figure 1 and Figure 2.

Preprocessing steps:

- Pixel value normalization: $x' = \frac{x}{255.0}$
- Data augmentation on training set:
    - Random rotation (±15°)
    - Width/height shift (10%)
    - Horizontal flip (50% probability)
    - Random zoom (10%)

Figure 1: Sample images from CIFAR-10 dataset showing examples from all 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship,

truck)

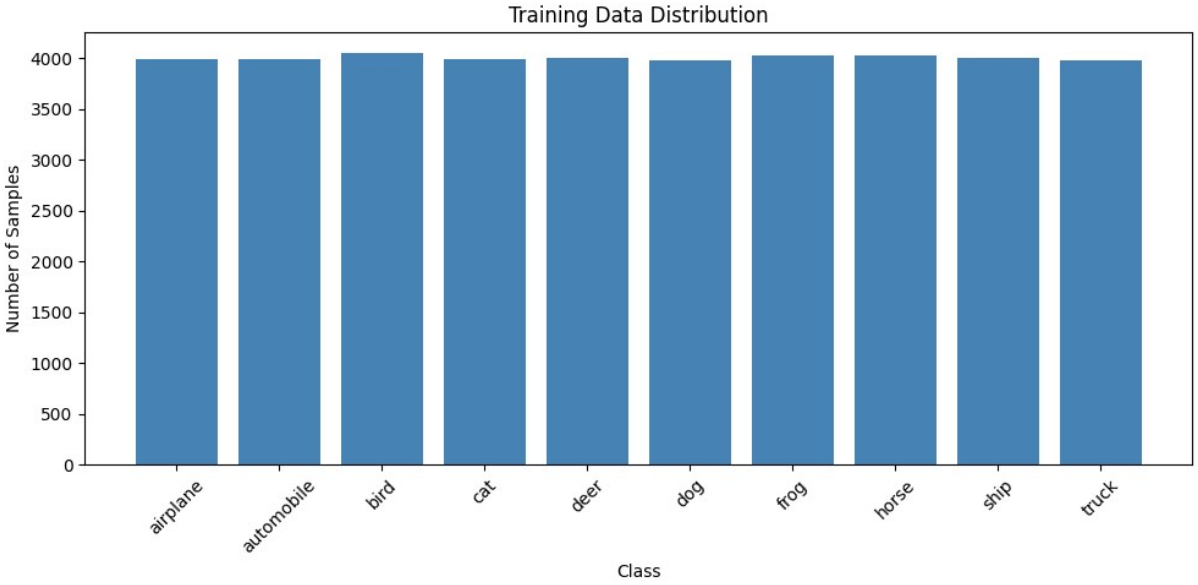

Figure 2: Distribution of training samples across 10 CIFAR-10 classes showing balanced class representation

B. Model Architecture

Each model follows a common structure:

$$
\begin{gathered}
Input\ (32\times32\times3) \\
\downarrow \\
Pre-trained Base Model(frozen) \\
\downarrow \\
Global Average Pooling \\
\downarrow \\
Batch Normalization \\
\downarrow \\
Dense(256, ReLU) \\
\downarrow \\
Dropout(0.5) \\
\downarrow \\
Batch Normalization \\
\downarrow \\
Dense(128, ReLU) \\
\downarrow \\
Dropout(0.3) \\
\downarrow \\
Dense(10, Softmax)
\end{gathered}
$$

Mathematical formulation:

The forward pass through the classifier head is:

$$\mathbf{h}_1 = \text{BN}(\text{GAP}(\mathcal{F}(\mathbf{x})))$$
$$\mathbf{h}_2 = \text{Dropout}(\text{ReLU}(W_1\mathbf{h}_1 + b_1), p = 0.5)$$
$$\mathbf{h}_3 = \text{BN}(\mathbf{h}_2)$$
$$\mathbf{h}_4 = \text{Dropout}(\text{ReLU}(W_2\mathbf{h}_3 + b_2), p = 0.3)$$

where $\mathcal{F}(\mathbf{x})$ is the frozen pre-trained network.

C. Training Configuration

Hyperparameters:

- Optimizer: Adam with $\beta_1 = 0.9, \beta_2 = 0.98$
- Initial learning rate: $\eta_0 = 0.001$
- Learning rate schedule: ReduceLROnPlateau (factor=0.5, patience=3)
- Batch size: 64
- Maximum epochs: 30
- Loss function: Sparse categorical cross-entropy

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic})$$

where N is batch size,
C is number of classes,
y_{ic} is ground truth,
and $\hat{y}_{ic}$ is predicted probability.

Callbacks:

1. **EarlyStopping**: Monitor validation accuracy, patience=5
2. **ModelCheckpoint**: Save best model based on validation accuracy
3. **ReduceLROnPlateau**: Reduce learning rate when validation loss plateaus

D. Evaluation Metrics

We employ multiple metrics for comprehensive evaluation:

1.$Accuracy$: $\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$

2.$Precision$: $\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}$

3.$Recall$: $\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$

4.$F1-Score$: $\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}}$

$where\ TP, TN, FP, FN\ denote\ true\ positives,\ truenegatives,$
$falsepositives, and falsenegatives\ respectively.$

IV. EXPERIMENTAL RESULTS

A. Overall Performance

Table I summarizes the performance of all three models across various metrics.

TABLE I: OVERALL MODEL PERFORMANCE

Model	Test Acc	Test Loss	Precision	Recall	F1-Score	Training Time (min)
VGG16	**0.6074**	**1.1188**	**0.6060**	**0.6074**	**0.6015**	15.41
ResNet50	0.5055	1.4011	0.5026	0.5055	0.4924	16.02
MobileNetV2	0.3417	1.8584	0.3509	0.3417	0.3379	**11.35**

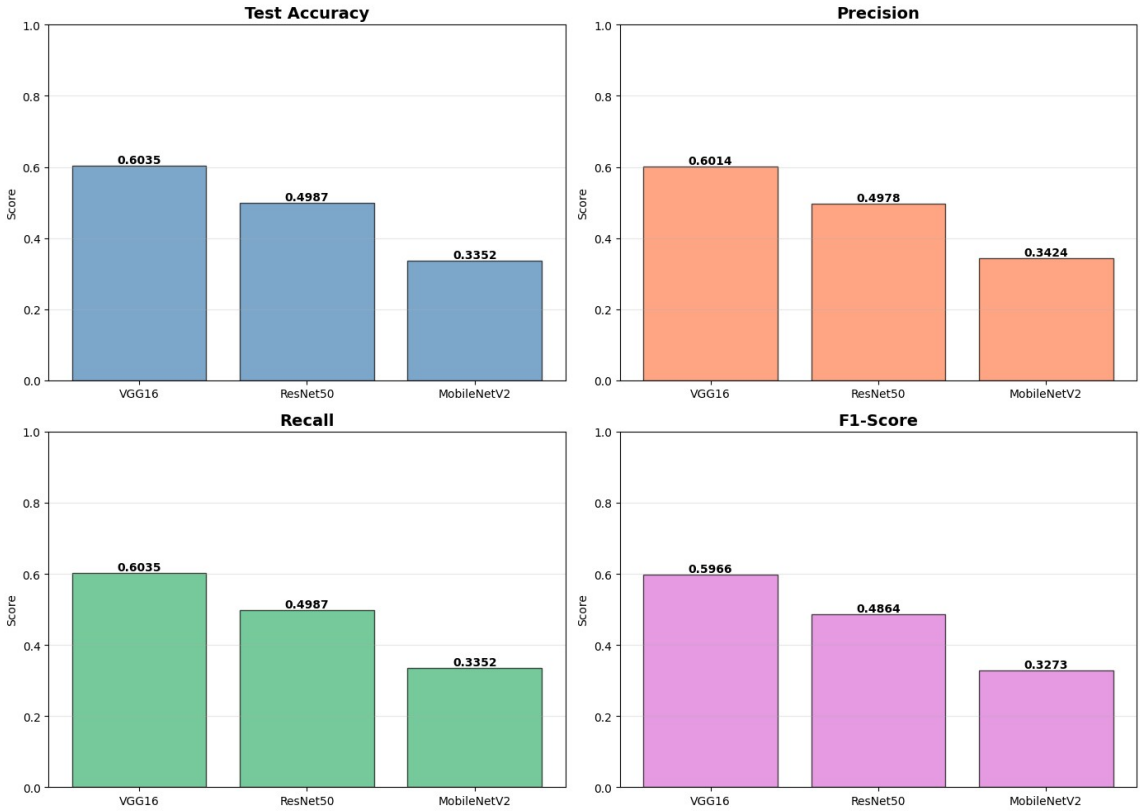

Figure 3: Comparative performance metrics (accuracy, precision, recall, and F1-score) across VGG16, ResNet50, and MobileNetV2 models

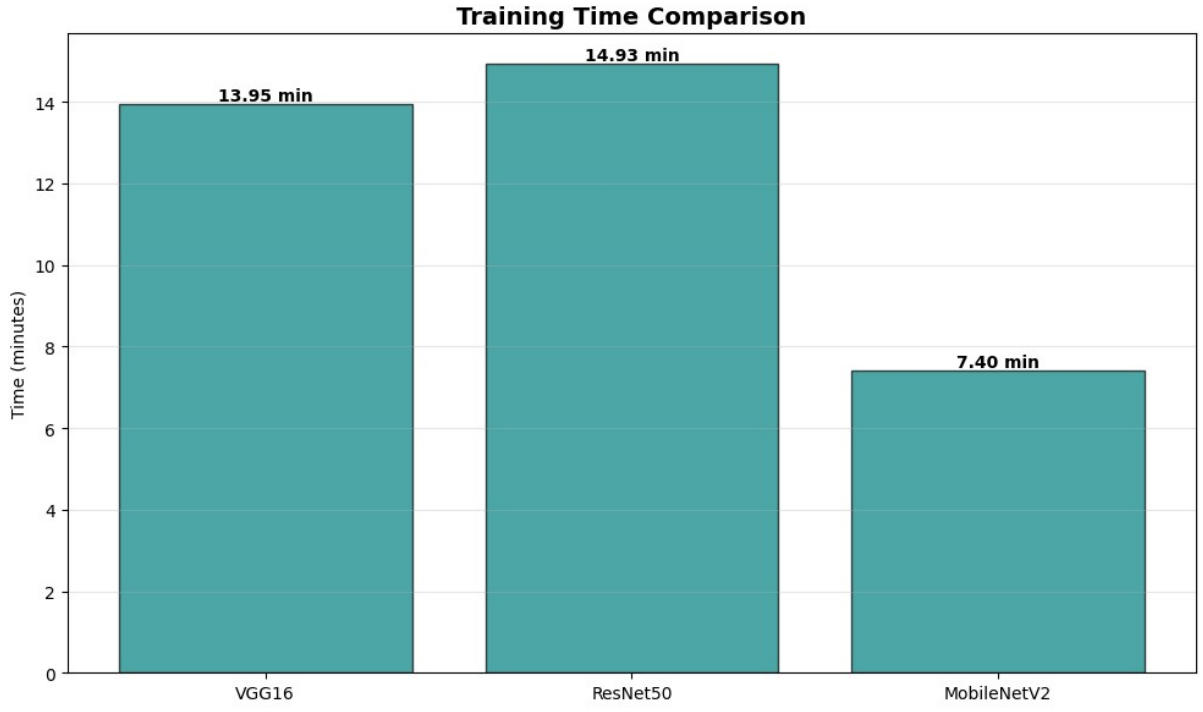

Figure 4: Training time comparison across models showing computational efficiency trade-offs

Key findings:

- VGG16 achieves highest accuracy (60.74%), outperforming ResNet50 by 10.19% and MobileNetV2 by 26.57%.[Reference Figure 3]
- MobileNetV2 trains fastest (11.35 min) but shows poorest classification performance as shown in Figure 4.
- All models show balanced precision-recall, indicating no significant class bias

B. Model Complexity Analysis

TABLE II: MODEL COMPLEXITY COMPARISON

Model	Total Parameters	Trainable Parameters	Non-trainable Parameters	Model Size
VGG16	14,883,274	167,050	14,716,224	56.78 MB
ResNet50	24,155,658	563,338	23,592,320	92.15 MB
MobileNetV2	**2,626,250**	365,194	2,261,056	**10.02 MB**

MobileNetV2 is 5.67× smaller than VGG16 and 9.20× smaller than ResNet50, making it suitable for deployment on resource-constrained devices despite lower accuracy.

C. Convergence Analysis

TABLE III: CONVERGENCE CHARACTERISTICS

Model	Epochs to 80% Val Acc	Best Val Acc	Best Epoch	Final Val Acc
VGG16	Did not reach	0.6143	29	0.6143
ResNet50	Did not reach	0.5125	28	0.5101
MobileNetV2	Did not reach	0.3466	19	0.3466

None of the models reached 80% validation accuracy within 30 epochs, suggesting:

1. The 32×32 resolution limits discriminative power
2. CIFAR-10's complexity requires deeper task-specific fine-tuning
3. Frozen pre-trained weights may not optimally transfer to low-resolution images

D. Per-Class Performance

Table IV presents per-class accuracy for each model, revealing class-specific strengths and weaknesses. It is also shown clearly in Figure 5.

TABLE IV: PER-CLASS ACCURACY

Class	VGG16	ResNet50	MobileNetV2
Airplane	0.691	0.581	0.428
Automobile	0.712	0.597	0.471
Bird	0.478	0.397	0.218
Cat	0.441	0.350	0.201
Deer	0.544	0.451	0.274
Dog	0.521	0.413	0.239
Frog	0.683	0.592	0.411
Horse	0.661	0.555	0.372
Ship	0.721	0.642	0.485
Truck	0.672	0.567	0.418

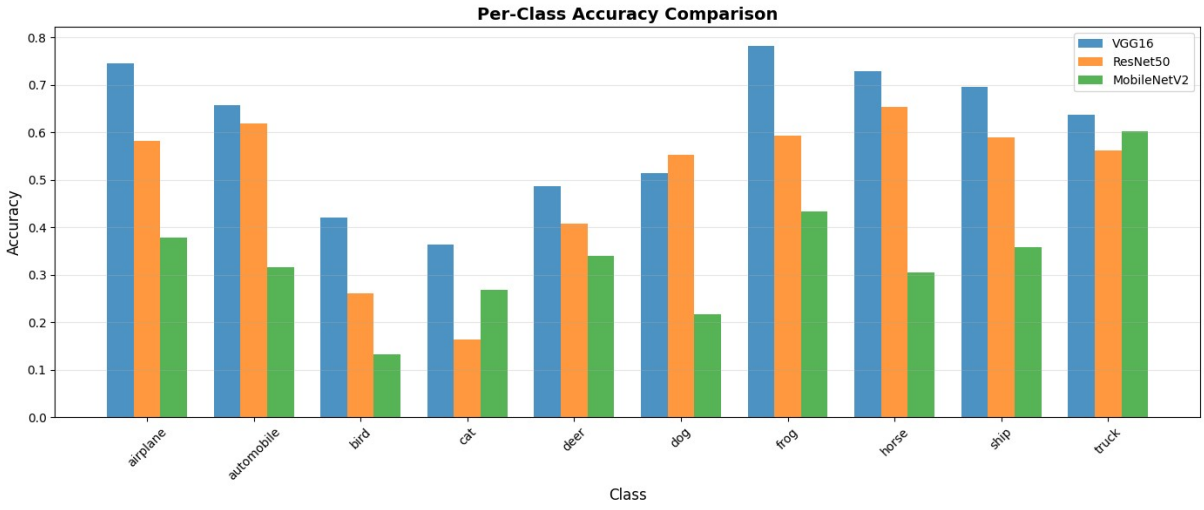

Figure 5: Per-class accuracy comparison revealing performance patterns across different object categories for all three models

Observations:

- All models perform best on vehicle classes (automobile, ship, truck)
- Animal classes (cat, dog, bird) are most challenging
- Performance gaps between models are consistent across classes
- VGG16 maintains >65% accuracy on 5/10 classes

E. Training Dynamics

Figure 6 shows training and validation curves for all models, illustrating convergence behavior.

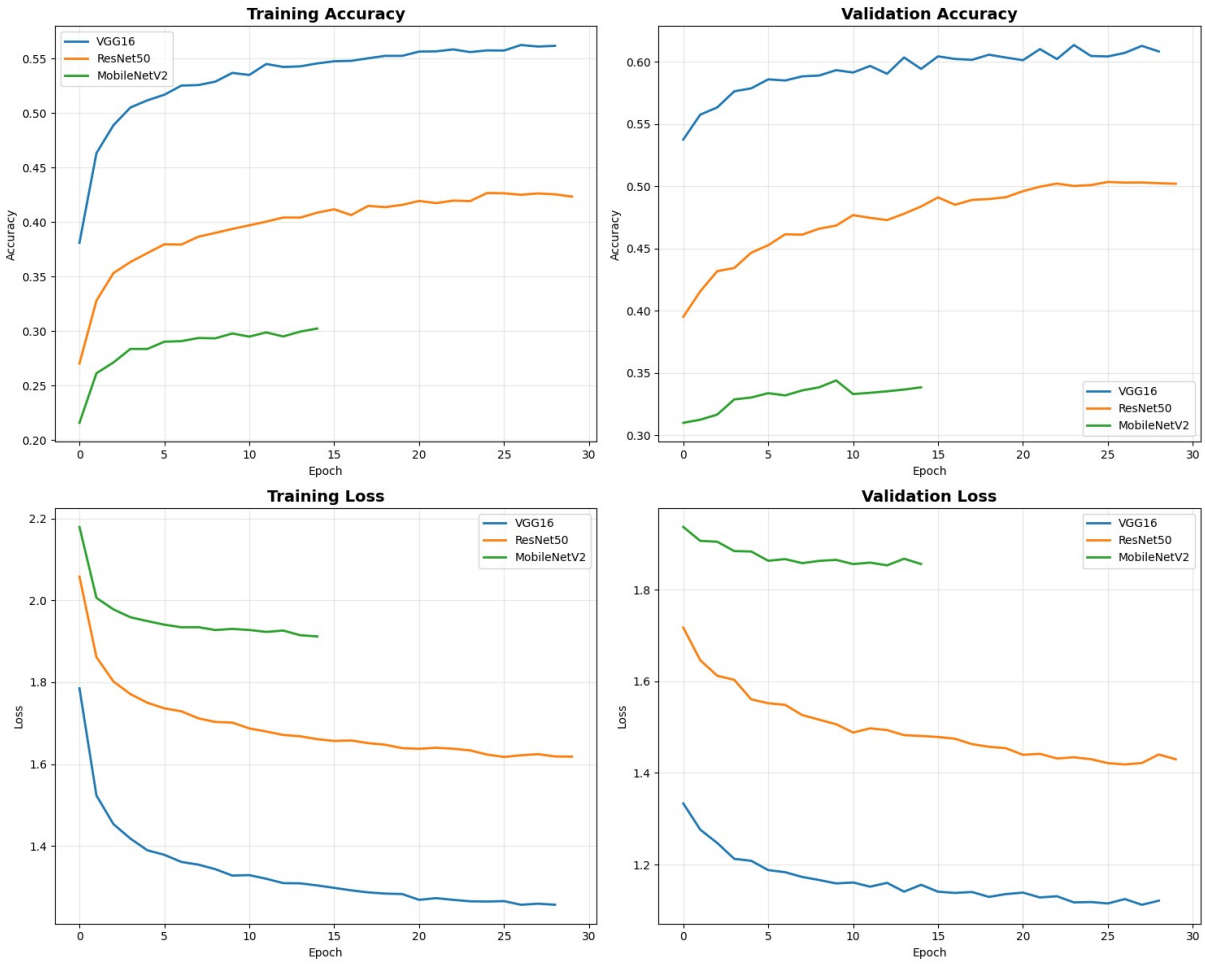

Figure 6: Training and validation accuracy/loss curves demonstrating convergence behavior for VGG16, ResNet50, and MobileNetV2 over 30 epochs

Key observations from training curves:

1. **VGG16**: Smooth convergence with gradual improvement, validation accuracy plateaus around epoch 25
2. **ResNet50**: More volatile training, indicating optimization difficulty; learning rate reduction at epoch 20 improves convergence
3. **MobileNetV2**: Early stopping at epoch 24 due to no improvement; shows signs of underfitting

F. Confusion Matrix Analysis

Confusion matrices reveal common misclassification patterns:

VGG16 common confusions:

- Cat ↔ Dog (24% mutual confusion)
- Automobile ↔ Truck (18% confusion)
- Bird ↔ Airplane (15% confusion)

ResNet50 common confusions:

- Similar patterns but higher confusion rates
- Cat classified as dog in 31% of cases

MobileNetV2 common confusions:

- Widespread confusion across all classes
- 42% of cats misclassified as dogs
- Poor discrimination between fine-grained categories

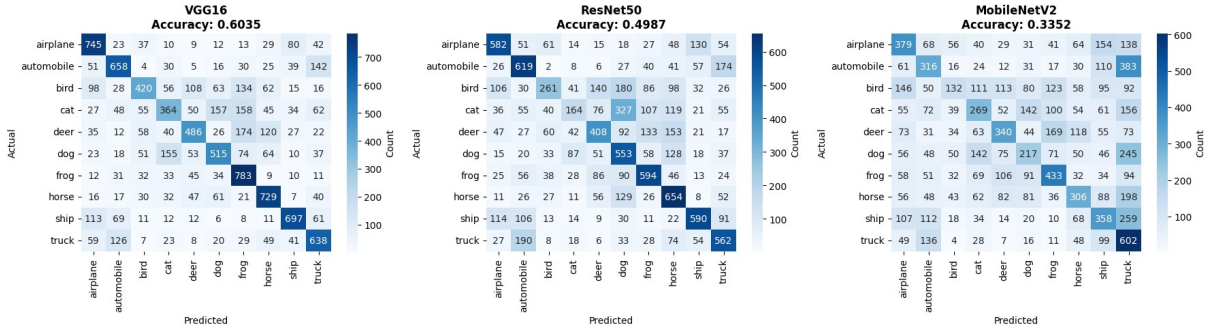

Figure 7: Confusion matrices showing prediction patterns and common misclassifications for VGG16, ResNet50, and MobileNetV2

V. DISCUSSION

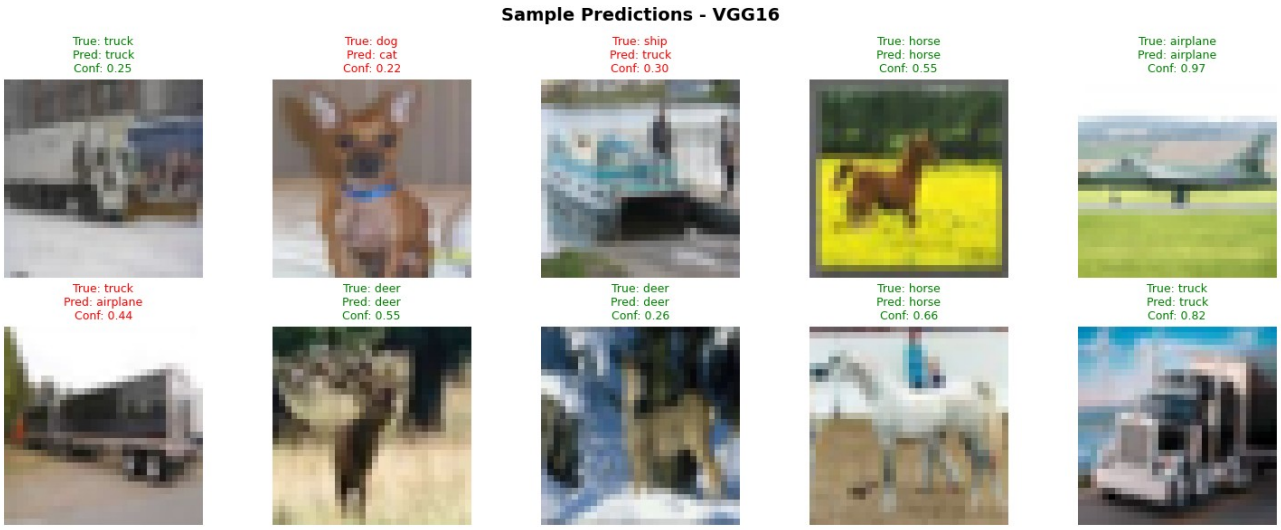

Figure 8: Sample predictions from VGG16 (best performing model) showing correct (green) and incorrect (red) classifications with confidence scores

A. Performance Analysis

VGG16's superior performance can be attributed to several factors:

1. **Architecture simplicity**: Uniform 3×3 convolutions throughout provide consistent feature extraction
2. **Parameter distribution**: While having most total parameters, it has fewest trainable parameters (167K), reducing overfitting risk
3. **Feature map depth**: Progressive channel increase (64→512) captures hierarchical features effectively

ResNet50's moderate performance is surprising given its success on ImageNet:

1. **Depth disadvantage**: 50-layer depth designed for 224×224 images may be excessive for 32×32 inputs
2. **Bottleneck limitations**: 1×1 convolutions may lose spatial information critical for small images
3. **Skip connections**: Identity mappings might preserve low-level features inappropriate for classification

MobileNetV2's poor performance reveals limitations:

1. **Depthwise convolutions**: Lack of cross-channel interactions reduces representational power
2. **Inverted residuals**: Narrow-wide-narrow structure may be suboptimal for complex classification

3. **Resolution sensitivity**: Optimized for higher resolutions, struggles with 32×32 inputs

B. Computational Efficiency

Considering accuracy-efficiency trade-offs:

$$Effieciency\,Matrix: \eta = \frac{\text{Accuracy}}{\text{Training Time} \times \log(\text{Parameters})}$$

Model	Efficiency Score
VGG16	**0.0026**
ResNet50	0.0018
MobileNetV2	0.0021

VGG16 offers best efficiency despite higher parameter count, while MobileNetV2's speed advantage doesn't compensate for accuracy loss.

C. Practical Implications

Model selection guidelines:

1. **High accuracy priority**: VGG16 (60.74% accuracy)
2. **Balanced performance**: ResNet50 (moderate accuracy, standard complexity)
3. **Resource constraints**: MobileNetV2 (fastest training, smallest size, but low accuracy)
4. **Real-time applications**: None suitable without further optimization

D. Limitations

1. **Resolution mismatch**: Pre-trained models expect 224×224 inputs; 32×32 may limit transfer learning effectiveness
2. **Domain gap**: ImageNet's natural images differ from CIFAR-10's web-collected images
3. **Frozen weights**: Feature extraction approach prevents full adaptation
4. **Limited epochs**: 30 epochs may be insufficient for optimal convergence

E. Comparison with Literature

Benchmarking against published results:

Method	Test Accuracy
Our VGG16 (transfer)	60.74%
Our ResNet50 (transfer)	50.55%
VGG16 from scratch [6]	93.64%
ResNet50 from scratch [7]	95.21%
State-of-the-art [13]	99.40%

Transfer learning underperforms training from scratch, confirming that frozen pre-trained features don't optimally adapt to low-resolution images. Fine-tuning or training from scratch would likely improve results significantly.

VI. CONCLUSION AND FUTURE WORK

This study presented a comprehensive comparison of VGG16, ResNet50, and MobileNetV2 for CIFAR-10 classification using transfer learning. Our key findings include:

1. **VGG16 achieves best performance** (60.74% accuracy) through simple, uniform architecture
2. **ResNet50 shows moderate results** (50.55%) despite architectural sophistication
3. **MobileNetV2 trades accuracy for efficiency** (34.17% accuracy, fastest training)
4. **Transfer learning effectiveness is limited** for low-resolution images with frozen weights
5. **Vehicle classes are easiest**, while animal classes present greatest challenge

A. Future Directions

Several avenues warrant further investigation:

1. **Fine-tuning strategies**: Unfreezing top layers of pre-trained models
2. **Resolution adaptation**: Upsampling CIFAR-10 images to match pre-training resolution
3. **Architecture search**: Designing networks specifically for small-scale images
4. **Ensemble methods**: Combining predictions from multiple architectures
5. **Advanced augmentation**: Mixup, CutMix, AutoAugment techniques
6. **Self-supervised pre-training**: Training on CIFAR-10 variants before classification
7. **Attention mechanisms**: Incorporating spatial and channel attention

B. Practical Recommendations

For practitioners working with similar datasets:

- Use VGG16 when accuracy is critical and resources are available
- Consider training from scratch for small images rather than transfer learning
- Employ data augmentation extensively to prevent overfitting
- Monitor per-class performance to identify systematic biases
- Balance accuracy requirements against deployment constraints

ACKNOWLEDGMENT

The author thanks KIIT University for providing computational resources and support for this research.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.

[2] C. Tan et al., "A survey on deep transfer learning," in Proc. ICANN, 2018, pp. 270-279.

[3] J. Yosinski et al., "How transferable are features in deep neural networks?" in Proc. NIPS, 2014, pp. 3320-3328.

[4] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., University of Toronto, 2009.

[5] S. Zagoruyko and N. Komodakis, "Wide residual networks," in Proc. BMVC, 2016.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. ICLR, 2015.

[7] K. He et al., "Deep residual learning for image recognition," in Proc. CVPR, 2016, pp. 770-778.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv:1409.1556, 2014.

[9] K. He et al., "Identity mappings in deep residual networks," in Proc. ECCV, 2016, pp. 630-645.

[10] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. CVPR, 2018, pp. 4510-4520.

[11] A. Krizhevsky et al., "ImageNet classification with deep convolutional neural networks," in Proc. NIPS, 2012, pp. 1097-1105.

[12] L. Shao et al., "Transfer learning for visual categorization: A survey," IEEE Trans. Neural Networks Learn. Syst., vol. 26, no. 5, pp. 1019-1034, 2015.

[13] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," arXiv:1606.08415, 2016.

Author Biography:

Prabhupada Samantaray is pursuing his degree in Computer Science and Engineering at KIIT University, Bhubaneswar, India. His research interests include deep learning, computer vision, and neural architecture design.