Attention or Convolution — or Both? An Empirical Study in Image Recognition

Likith R, Anurag Kumar, Kundan Kumar
Department of Electrical Engineering, IIT Roorkee
{likith_r, anurag_k, kundan_k}@ee.iitr.ac.in

***Abstract*—This project investigates the comparative performance of attention-based models and convolutional neural networks (CNNs) for image recognition tasks. Specifically, we evaluate Vision Transformers (ViTs), known for their global self-attention mechanism, against established CNN architectures such as AlexNet and ResNet-50. The study aims to understand the trade-offs in terms of accuracy, computational efficiency, and data requirements between these paradigms. Additionally, we explore a hybrid architecture that integrates convolutional layers with transformer blocks to assess whether combining the local inductive biases of CNNs with the global reasoning of transformers yields performance gains. Experimental results highlight the strengths and limitations of each approach across varying dataset sizes and complexity levels.**

I. AlexNet

A. Model Overview

AlexNet, introduced by Krizhevsky et al., revolutionized image classification with its deep convolutional architecture and use of GPUs for training. The architecture consists of five convolutional layers and three fully connected layers, employing ReLU activations, overlapping max pooling, local response normalization, and dropout for regularization.

1) Architecture Details: Given an input image of size $224 \times 224 \times 3$, the network consists of:

- **Conv1:** 96 filters of size 11×11, stride 4 → ReLU → Local Response Normalization (LRN) → MaxPooling (3×3, stride 2)
- **Conv2:** 256 filters of size 5×5, padding 2 → ReLU → LRN → MaxPooling
- **Conv3–5:** 384, 384, and 256 filters (3×3), with ReLU (Conv5 followed by MaxPooling)
- **FC6, FC7:** Fully connected layers with 4096 neurons each, ReLU + Dropout ($p = 0.5$)
- **FC8:** Output layer with Softmax activation over N classes (e.g., 10 for CIFAR-10, 256 for Caltech-256)

2) Key Components and Formulas:

a) ReLU Activation: AlexNet uses the Rectified Linear Unit (ReLU) instead of tanh or sigmoid:

$$f(x) = \max(0, x)$$

b) Local Response Normalization (LRN): Encourages local competition among neuron activations:

$$b^i_{x,y} = \frac{a^i_{x,y}}{\left(k + \alpha \sum\limits_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2\right)^{\beta}}$$

Typical parameters: $k = 2$, $\alpha = 10^{-4}$, $n = 5$, $\beta = 0.75$.

c) Dropout Regularization: Applied to FC6 and FC7 to prevent co-adaptation:

With probability $p = 0.5$, a neuron is set to 0 during training.

d) Loss Function: Cross-entropy with L2 weight regularization:

$$\mathcal{L}(w) = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log f_c(x_i) + \frac{\lambda}{2}\|w\|^2$$

3) Training Configuration:

- **Optimizer:** AdamW (adaptive momentum with decoupled weight decay)
- **Batch Size:** 128
- **Learning Rate:** Controlled via OneCycleLR (max_lr = 0.01)
- **Weight Decay:** $\lambda = 0.01$

a) AdamW Update Rule: For parameter w, AdamW performs the following updates:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1)\nabla\mathcal{L}(w_t) \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)(\nabla\mathcal{L}(w_t))^2 \\ \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t} \\ w_{t+1} &= w_t - \eta\left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} + \lambda w_t\right) \end{aligned}$$

where $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$.

b) OneCycle Learning Rate Schedule: The OneCycleLR policy increases the learning rate to a peak value and then decreases it over training steps using a cosine annealing schedule:

$$\eta_t = \eta_{\text{min}} + \frac{1}{2}(\eta_{\text{max}} - \eta_{\text{min}})\left(1 + \cos\left(\frac{\pi t}{T}\right)\right)$$

where t is the current step and T is the total number of steps.

4) Data Augmentation: To enhance generalization, the following augmentations are applied:

- Random 224×224 crop from 256×256 images
- Horizontal flipping with 50% probability
- RGB intensity jittering using PCA-based color augmentation

5) Summary:

TABLE I: AlexNet Summary

Component	Value
Parameters	∼62 million
Learnable Layers	8 (5 conv + 3 fc)
Activation	ReLU
Regularization	Dropout + LRN
Optimizer	AdamW + OneCycleLR
Original Dataset	ImageNet-1K

B. Experimental Setup

We evaluate AlexNet using two datasets—CIFAR-10 and Caltech-256—via transfer learning from a pretrained ImageNet model. The final classification layer was modified to match the number of classes in each dataset: 10 for CIFAR-10 and 256 for Caltech-256.

All images were resized to 224 × 224 to meet AlexNet's input requirements. Training was conducted in a GPU-accelerated environment using mixed precision. We trained for up to 10 epochs on CIFAR-10 and 15 epochs on Caltech-256.

C. Data Preprocessing

- **Augmentation:** AutoAugment policies specific to CIFAR-10 were employed to enhance generalization.
- **Normalization:** Mean and standard deviation normalization was applied to align with AlexNet's pretrained configuration.
- **Batch Size:** 256 images per batch.
- **Optimizers and Schedulers:** AdamW optimizer with a OneCycle learning rate scheduler.

D. Transfer Learning and Fine-tuning

The pretrained AlexNet model was frozen except for the final classifier layer, which was replaced with a new fully connected layer tailored for the respective datasets. This strategy allowed faster convergence and reduced the risk of overfitting.

E. Training and Evaluation Metrics

We used CrossEntropyLoss as the loss function. Evaluation was based on:

- Test Accuracy
- Confusion Matrix
- Per-class Accuracy
- Classification Report (Precision, Recall, F1-Score)

F. Results on CIFAR-10

Training AlexNet on CIFAR-10 for 10 epochs yielded satisfactory generalization with the following results:

- **Test Accuracy:** *81.71%*

The confusion matrix in Fig. **??** shows high accuracy for most classes, although some confusion remains between visually similar categories. Table II presents detailed classification metrics.

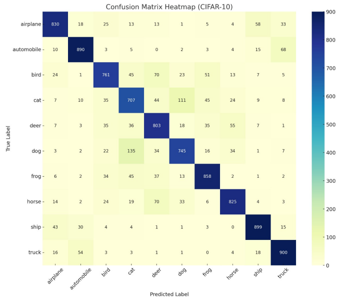

Fig. 1: confusion matrix for cifar 10

TABLE II: Classification Report for CIFAR-10

Class	Precision	Recall	F1-Score	Support
0	0.82	0.83	0.83	1000
1	0.88	0.88	0.88	1000
2	0.83	0.73	0.78	1000
3	0.73	0.65	0.69	1000
4	0.77	0.78	0.77	1000
5	0.76	0.78	0.77	1000
6	0.82	0.86	0.84	1000
7	0.81	0.86	0.84	1000
8	0.86	0.90	0.88	1000
9	0.87	0.90	0.88	1000
Overall	**0.82**	**0.82**	**0.82**	**10000**

G. Results on Caltech-256

Caltech-256 was trained for 15 epochs. Due to its higher class diversity and dataset complexity, the overall performance was lower than CIFAR-10.

- **Test Accuracy:** *65%*
- **Observations:** Misclassifications were common among visually or semantically similar categories.

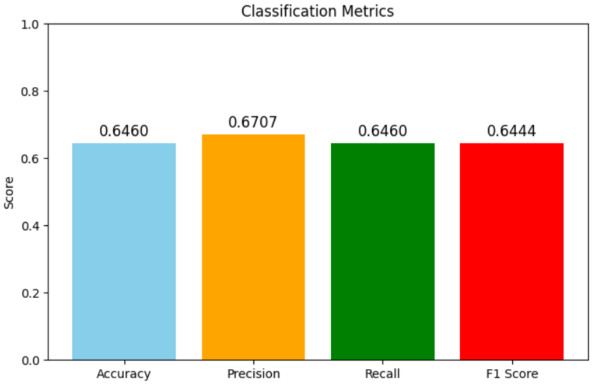

Fig. 2: Metric chart for Caltech-256

H. Comparative Analysis

TABLE III: Performance Comparison between CIFAR-10 and Caltech-256

Metric	CIFAR-10	Caltech-256
Test Accuracy	81.71%	65%

I. Discussion

AlexNet performs effectively on medium-scale datasets like CIFAR-10 but struggles with the larger class imbalance and semantic diversity of Caltech-256 when only the final layer is fine-tuned. Enhancements such as fine-tuning deeper layers or adopting more efficient architectures like ResNet or EfficientNet could significantly improve performance.

J. Placeholders for Results

- Insert error vs. epoch graph for both datasets here.
- Attach classification report tables (Caltech-256).
- Include visual sample outputs (correct vs. misclassified).

II. Vision Transformers (ViT)

Vision Transformers (ViTs), introduced in the seminal paper *An Image is Worth 16x16 Words*, represent a paradigm shift in computer vision. By adopting the transformer architecture—previously dominant in natural language processing—ViTs eliminate the need for convolutional layers in image classification. Instead, they divide input images into fixed-size patches (e.g., 16×16), flatten them into sequences, and treat these as tokens, much like words in NLP models. These embeddings, enriched with positional information, are processed through a standard transformer encoder. With sufficient data and compute resources, ViTs have achieved state-of-the-art performance on benchmarks like ImageNet, challenging the long-standing dominance of CNNs.

A. Model Architecture

The ViT architecture replaces traditional convolutional layers with transformer blocks operating on image patches. An input image $x \in \mathbb{R}^{H \times W \times C}$ is divided into $N = \frac{HW}{P^2}$ non-overlapping patches of size $P \times P$. Each patch is flattened and linearly projected using a learnable matrix $E \in \mathbb{R}^{(P^2 \cdot C) \times D}$, producing embeddings $z_0^p \in \mathbb{R}^{N \times D}$. A learnable classification token $z_0^0 \in \mathbb{R}^D$ is prepended to the sequence. Positional embeddings $E_{\text{pos}} \in \mathbb{R}^{(N+1) \times D}$ are then added:

$$z_0 = [z_0^0; z_0^1; \ldots; z_0^N] + E_{\text{pos}}$$

This sequence is passed through L transformer encoder layers. Each layer includes multi-head self-attention (MSA) and a feed-forward multi-layer perceptron (MLP), combined with residual connections and layer normalization:

$$z'_\ell = \text{MSA}(\text{LN}(z_{\ell-1})) + z_{\ell-1}, \quad z_\ell = \text{MLP}(\text{LN}(z'_\ell)) + z'_\ell$$

After the final layer, the output of the [CLS] token z_L^0 is passed to a classification head for final prediction.

B. Inductive Bias

Unlike CNNs, ViTs do not encode spatial locality or translation invariance by design. This lack of inductive bias increases data requirements but allows ViTs to learn global dependencies and spatial patterns purely from data. While this results in flexibility, it also demands careful pretraining and regularization strategies.

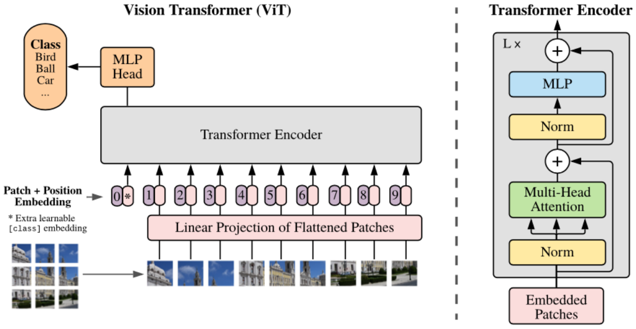

Fig. 3: Vision Transformer (ViT) architecture: Images are split into patches, embedded, and processed through transformer encoder blocks.

C. Our Implementation

We implemented the ViT-Small architecture (22M parameters) with a 16×16 patch size. The model was pretrained in two stages—first on the ImageNet-21K dataset (21,000 classes), then fine-tuned on ImageNet-1K (1,000 classes). This multi-stage pretraining allowed the model to capture both broad and fine-grained visual representations. Input images were resized to 224×224×3 and converted into tensor format for GPU processing. To adapt the model to the Caltech-256 dataset, we adopted a transfer learning strategy: the pretrained transformer backbone was frozen, and only a new linear classification head was trained to map features to 257 classes (256 categories + background).

This reduced computational cost and training time, while avoiding overfitting due to the smaller target dataset. Notably, we deliberately avoided data augmentation and regularization techniques to test the natural generalization ability of the models across different distributions.

D. Training and Testing Summary

- **Loss Function:** Cross-entropy loss
- **Optimizer:** AdamW with weight decay
- **Epochs:** 5 (extendable to 10)
- **Accelerator:** NVIDIA P100 GPU (16GB memory)
- **Framework:** PyTorch
- **Objective:** Study generalization and performance trade-offs between ViTs and CNNs

E. Evaluation Results

To assess model performance, we use confusion matrices generated from the test predictions. These visualizations highlight classification accuracy and error distribution across all classes.

III. ResNet-50 (CNN)

ResNet-50, introduced in the groundbreaking paper *Deep Residual Learning for Image Recognition* by He et al. (2015), marked a milestone in deep learning by enabling the successful training of ultra-deep convolutional neural networks. It solved the degradation problem—where

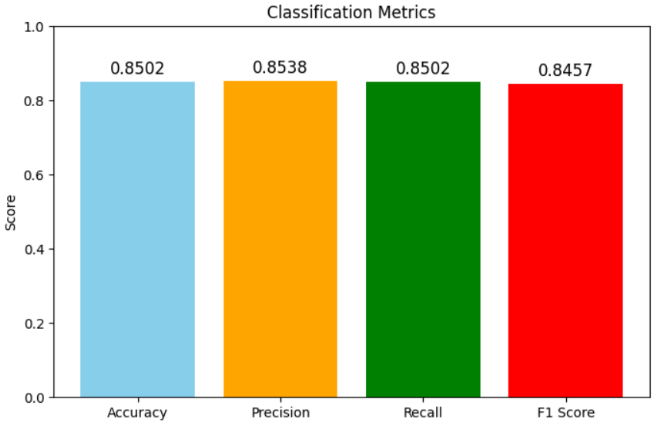

Fig. 4: Metric chart for Vision Transformer (Vit) pretrained on the ImageNet1k dataset.

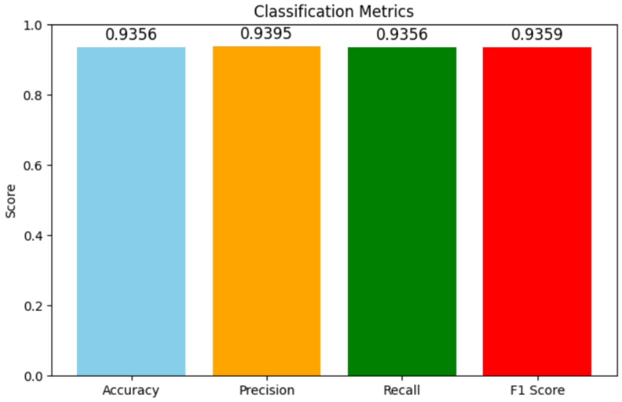

Fig. 5: Metric chart on the ImageNet21k dataset.

accuracy degrades as depth increases—by incorporating residual (or skip) connections that allow gradients to flow backward without attenuation.

With its 50 layers and approximately 25.6 million parameters, ResNet-50 strikes a balance between depth, performance, and computational cost, making it a preferred architecture for many visual recognition benchmarks. The model has served as a foundational backbone for numerous downstream tasks and continues to be a competitive baseline in contrast to more recent models like Vision Transformers (ViTs).

A. Model Architecture

ResNet-50 is composed of a stem layer followed by four sequential residual stages. The building block of each stage is a bottleneck residual unit designed for efficiency and depth. Let $x \in \mathbb{R}^{H \times W \times C}$ be the input tensor. The residual unit computes a transformation $\mathcal{F}(x)$ and adds it to the original input:

$$\mathcal{F}(x) = W_3 \cdot \sigma(\mathrm{BN}(W_2 \cdot \sigma(\mathrm{BN}(W_1 \cdot x))))$$

$$y = \mathcal{F}(x) + x \quad \text{(if dimensions match)}$$

If x and $\mathcal{F}(x)$ differ in shape, a linear projection W_s is applied to align dimensions:

$$y = \mathcal{F}(x) + W_s \cdot x$$

The overall architecture comprises:

- Initial 7×7 convolution + batch norm + ReLU (stride 2)
- 3×3 max pooling (stride 2)
- Four stages with $\{3, 4, 6, 3\}$ bottleneck blocks
- Global average pooling
- Fully connected layer for classification

Each bottleneck block uses a 1×1–3×3–1×1 convolution stack to reduce, process, and restore channel dimensions, optimizing both accuracy and efficiency.

B. Inductive Bias

Unlike Vision Transformers (ViTs), CNNs like ResNet-50 possess strong built-in inductive biases:

- **Local connectivity:** Convolutions focus on local features, enabling efficient edge and texture detection.
- **Translation invariance:** Weight sharing across spatial locations captures patterns irrespective of position.
- **Hierarchical representation:** Early layers capture low-level features (edges, blobs), while deeper layers learn complex shapes.

These priors reduce data requirements and enhance generalization on small to medium-sized datasets. However, they may also limit expressiveness in capturing long-range spatial dependencies, where ViTs excel.

C. Our Implementation

We used the standard ResNet-50 model pretrained on the ImageNet-1K dataset. To evaluate its transferability and robustness, we fine-tuned it on two datasets of contrasting scale and complexity:

- **CIFAR-10:** A small, low-resolution dataset with 10 coarse object categories. Achieved **89% accuracy**.
- **Caltech-256:** A more diverse dataset with 256 categories and greater intra-class variation. Achieved **86% accuracy**.

a) Design choices and methodology::

- Input images were resized to $224\times224\times3$ and normalized using dataset-specific means and standard deviations.
- The pretrained convolutional backbone was initially frozen to retain general features.
- A lightweight fully connected head was trained to adapt to new label spaces (10 and 257 classes respectively).
- After partial convergence, full fine-tuning was performed with a reduced learning rate for better adaptation.
- No data augmentation was used to isolate the model's intrinsic generalization capacity.

This setup minimizes computational overhead and overfitting while showcasing ResNet's robustness to domain shifts.

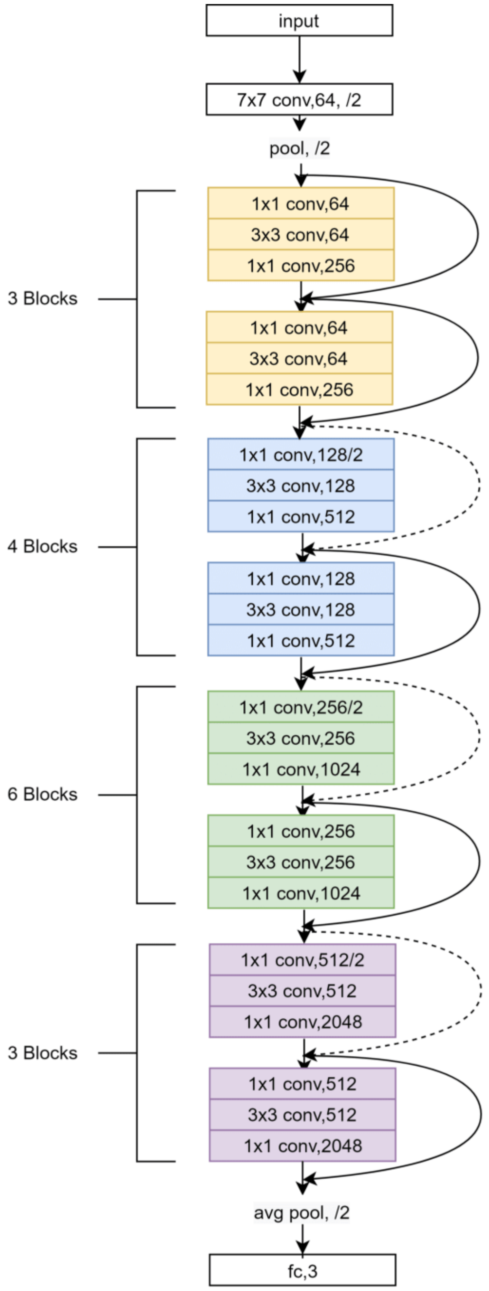

Fig. 6: ResNet-50 architecture: input flows through a series of residual bottleneck blocks with identity or projection shortcuts.

D. Training and Testing Summary

- **Loss Function:** Cross-entropy loss
- **Optimizer:** AdamW (decoupled weight decay)
- **Learning rate schedule:** StepLR with warm-up
- **Epochs:** 10 (CIFAR-10 and Caltech-256)
- **Accelerator:** NVIDIA P100 GPU (16GB)
- **Framework:** PyTorch
- **Objective:** Evaluate the trade-off between architectural inductive bias and data-driven learning in CNNs vs. ViTs

E. Evaluation Results

We evaluated the model using class-wise confusion matrices, highlighting where ResNet excels and where it struggles:

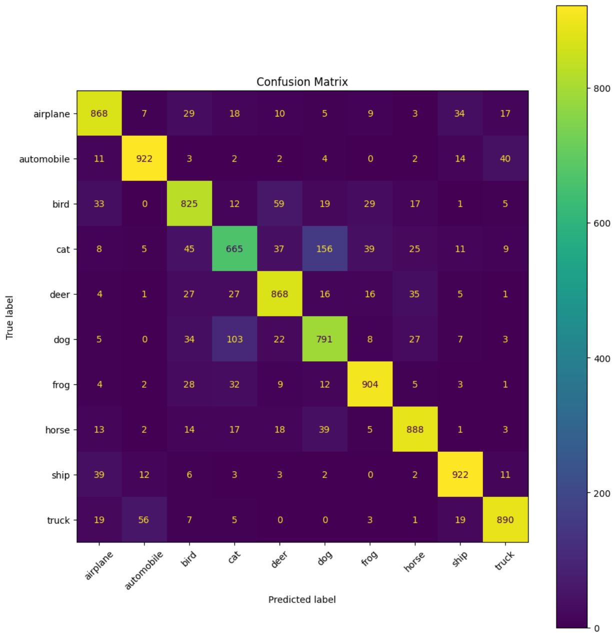

Fig. 7: Confusion Matrix for ResNet-50 on CIFAR-10. High diagonal dominance reflects strong per-class accuracy.

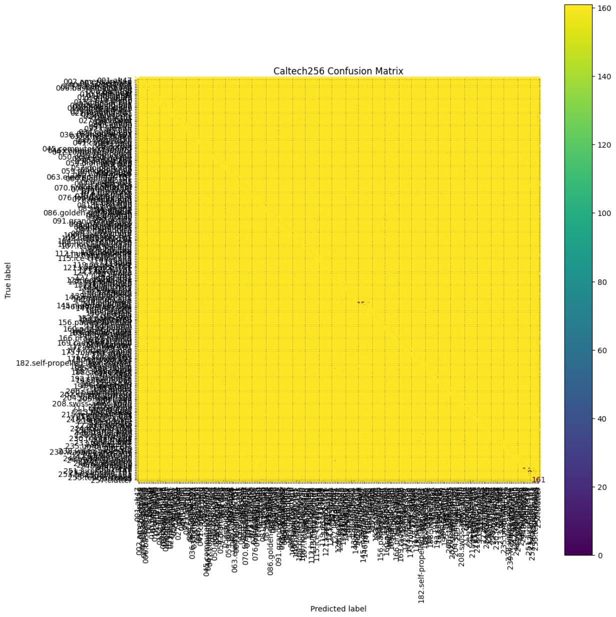

Fig. 8: Confusion Matrix for ResNet-50 on Caltech-256. Minor confusion appears among visually similar categories.

F. Closing Insight

ResNet-50 continues to demonstrate competitive performance across a spectrum of tasks, validating the enduring relevance

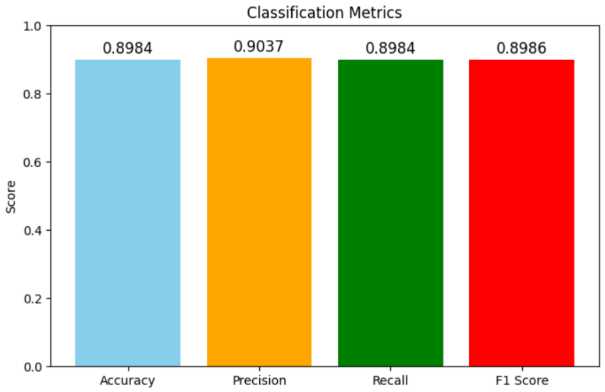

Fig. 9: metric chart for resnet50 pretrained on imagenet21k dataset.

of convolutional architectures. While ViTs represent a new frontier by learning long-range dependencies without strong priors, ResNet's structured design offers remarkable efficiency and generalization—especially in scenarios where data is limited or compute is constrained. The interplay between bias and flexibility remains a central theme in choosing between CNNs and transformers for vision tasks.

IV. CoAtNet

A. *Introduction to CoAtNet*

In the realm of deep learning for computer vision, two dominant paradigms have emerged: convolutional neural networks (CNNs), known for their strong inductive biases and performance on smaller datasets, and vision transformers, which leverage self-attention to model long-range dependencies and scale effectively with large data. Each has unique strengths and weaknesses—CNNs generalize efficiently but may lack global context modeling, while transformers are flexible but data-hungry.

CoAtNet (Convolution and Attention Network) is a hybrid architecture designed to unify the strengths of both paradigms. It introduces a staged framework that begins with convolutional layers for stable local feature extraction and gradually transitions to attention mechanisms for capturing global representations. This design enables CoAtNet to scale across various data regimes, offering a balanced trade-off between generalization and capacity.

B. *Architecture*

CoAtNet follows a five-stage hierarchical design, denoted S0 through S4, where spatial resolution decreases and feature dimensionality increases with depth. The key innovation is the seamless integration of convolutional blocks in early stages with transformer-based attention blocks in later stages.

1) *Stage-wise Structure:*

- **S0 (Stem)**: Input preprocessing using a standard 3×3 convolution.
- **S1–S2**: High-resolution processing using MBConv blocks—mobile inverted bottleneck convolutions with depthwise separable operations and SE modules.
- **S3–S4**: Transition to Transformer blocks featuring relative self-attention and feedforward networks, ideal for global context modeling.

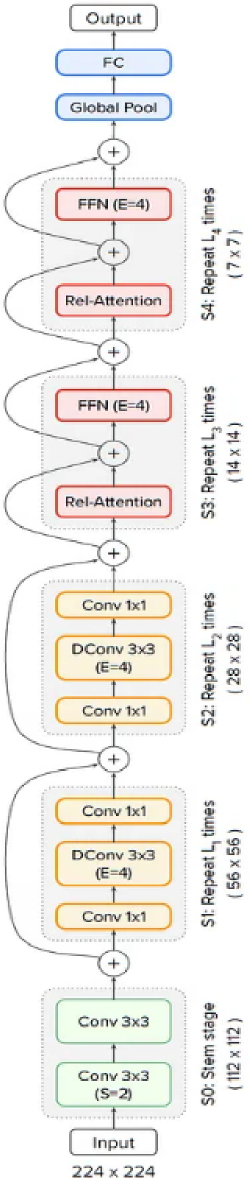
Output
FC
Global Pool
FFN (E=4)
Rel-Attention
S4: Repeat L_4 times (7 x 7)
FFN (E=4)
Rel-Attention
S3: Repeat L_3 times (14 x 14)
Conv 1x1
DConv 3x3 (E=4)
Conv 1x1
S2: Repeat L_2 times (28 x 28)
Conv 1x1
DConv 3x3 (E=4)
Conv 1x1
S1: Repeat L_1 times (56 x 56)
Conv 3x3
Conv 3x3 (S=2)
S0: Stem stage (112 x 112)
Input
224 x 224

Fig. 10: CoAtNet architecture: A five-stage hybrid pipeline combining convolution (MBConv) and attention (Transformer) blocks.

MBConv blocks are efficient for local processing, while transformer blocks expand the model's receptive field. The progressive downsampling ensures that attention layers operate on reduced spatial dimensions, mitigating the quadratic complexity of self-attention.

2) Block Transition: From Conv to Attention:

- **S1–S2**: Employ MBConv for local inductive bias and parameter efficiency.
- **S3–S4**: Shift to self-attention for enhanced global feature learning.

This gradual transition enables CoAtNet to generalize like a CNN on small datasets while scaling like a transformer on larger tasks.

C. Inductive Bias

CoAtNet's hybrid design explicitly encodes spatial priors through convolution in early layers, providing translation invariance and efficient weight sharing. This strong inductive bias supports generalization when training data is scarce. Conversely, self-attention layers introduced later in the network allow dynamic, input-dependent global interactions. By combining these elements, CoAtNet offers a flexible mechanism that adapts to data availability and task complexity, generalizing well across a range of vision applications.

D. How We Implemented It

We implemented CoAtNet-1, pretrained on ImageNet-1K, and fine-tuned it on the Caltech-256 dataset using transfer learning. Mixed-precision (fp16) training was employed on an NVIDIA Tesla P100 GPU to enhance speed and memory efficiency.

To maintain computational feasibility, CoAtNet uses stride-2 convolutions during early MBConv stages (S0–S2) for spatial downsampling. By the time attention layers are applied in S3 and S4, the feature maps have reduced dimensions (e.g., 14×14 or 7×7), making attention computations tractable.

E. Training and Testing Summary

- **Loss Function:** Cross-entropy
- **Optimizer:** AdamW with weight decay
- **Epochs:** 10
- **Accelerator:** NVIDIA Tesla P100 GPU (16GB)
- **Framework:** PyTorch
- **Batch Size:** 256
- **Goal:** Achieve efficient training-performance trade-off across CNN, ViT, and hybrid models

F. Evaluation Results

We evaluated model performance using confusion matrices, providing insight into class-wise prediction accuracy and error distribution.

V. Conclusion

A. Comparing Convolution and Attention

In our study comparing convolutional and attention-based architectures, we evaluated ResNet50 and Vision Transformer (ViT), both pretrained on the ImageNet-1K dataset and fine-tuned on the Caltech-256 dataset. The results showed that ResNet50 achieved a top-1 accuracy of **86%**, slightly outperforming ViT, which reached **85%**. This performance gap highlights a difference in the generalizability of the two models. ResNet50, being a convolutional neural network, benefits from strong inductive biases such as locality and translation invariance, which help it generalize better on relatively small datasets like ImageNet-1K or Caltech-256. On the other hand, ViT has a weaker inductive bias and relies more on learning spatial patterns directly from data. While this makes it powerful on large-scale datasets, it may struggle to generalize as effectively on smaller datasets without extensive data or augmentation. The reduced bias in ViT offers flexibility but also makes it more dependent on the scale and diversity of training data, which explains the slight drop in accuracy compared to ResNet50 in this experiment, Contrastingly ViT pretrained on imagenet21k , fine-tuned on caltech256 outperformed resnet50 trained on the same with top-1 accuracy of **93%** and **89%** respectively, this improvement is was a consequence of the large data it was trained on, ViT captured better long range features and obtained a better fit than the Resnet50.

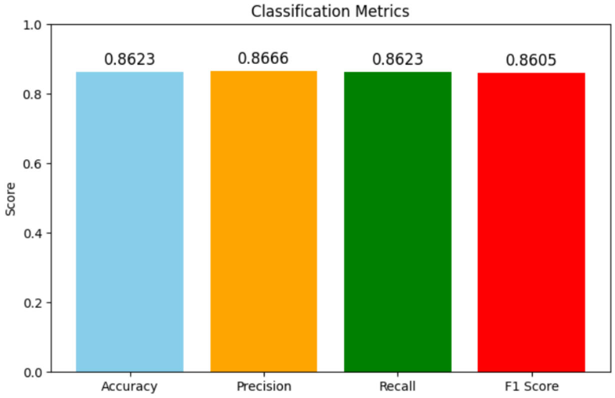

Fig. 11: metric chart for CoAtNet model pretrained on imagenet1k .

B. Need for a Hybrid

To bridge the gap in performance between convolutional and attention-based models, we explored CoAtNet—a hybrid architecture that combines convolutional layers with Transformer blocks. CoAtNet integrates the strong inductive biases of convolutional networks, such as locality and translation equivariance, with the global receptive field and dynamic attention mechanism of Transformers. This fusion allows the model to capture both fine-grained local patterns and long-range dependencies, enabling it to generalize well across varying dataset sizes. As a result, CoAtNet demonstrated improved generalizability on both ImageNet-1K and Caltech-256 datasets, achieving a top-1 accuracy of approximately **86%**, effectively matching or slightly exceeding the performance of pure convolutional or transformer-based models individually.

TABLE IV: Comparison of Top-1 Accuracy on ImageNet-1K

Model	Top-1 Accuracy (%)
ResNet50	86.6
CoAtNet	86.0
ViT	85.0

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[3] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[4] X. Ding, X. Zhang, J. Han, and G. Ding, "CoAtNet: Marrying convolution and attention for all data sizes," *arXiv preprint arXiv:2106.04803*, 2021.

[5] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019.

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[7] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[8] G. E. Hinton *et al.*, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[9] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?," in *Proc. IEEE ICCV*, 2009.

[10] A. Krizhevsky, "Learning multiple layers of features from tiny images," Technical Report, University of Toronto, 2009.

[11] J. Deng *et al.*, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE CVPR*, 2009, pp. 248–255.

[12] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, 2007.

[13] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning augmentation strategies from data," in *Proc. IEEE CVPR*, 2019, pp. 113–123.

[14] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," *arXiv preprint arXiv:1404.5997*, 2014.

[15] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[16] P. Micikevicius *et al.*, "Mixed precision training," in *International Conference on Learning Representations (ICLR)*, 2018.

[17] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. ICML*, 2019, pp. 6105–6114.