Task A:

DynamicConv2D: A Specialized Convolution Module

We propose DynamicConv2D, a specialized convolution module designed with the primary goal of dynamically adapting to an arbitrary number of input channels while preserving spatial dimensions. This design is motivated by practical observations: in real-world scenarios, spectral channels of input images may be incomplete due to sensor limitations or transmission losses. Compared to traditional convolutional neural networks (CNNs), which utilize fixed-weight kernels and struggle with variable input channel configurations, the DynamicConv2D module adjusts kernel weights dynamically based on the input features, selecting the most suitable kernel subset for the current input.

The implementation of DynamicConv2D involves introducing multiple sets of convolutional kernels in the early layers of the model. The module automatically selects and applies a subset or the entirety of these weights depending on the input channel composition. For example, with a standard three-channel RGB input, the full set of weights is used. However, in cases with only two or one input channels, only the corresponding parts of the kernel weights are activated for feature extraction. This enables the model to remain robust and effective when faced with heterogeneous inputs, such as RG, GB, or even single-channel data like R or G.

However, this adaptive capability comes at a computational cost. Maintaining multiple kernel sets and dynamically selecting them during each forward pass significantly increases the computational burden (FLOPs), especially when the batch size is large or input resolution is high. Additionally, training may converge more slowly due to the increased diversity in input representation the model must learn, necessitating longer training times for stable performance.

Despite these trade-offs, DynamicConv2D presents a clear theoretical advantage. It offers a flexible and adaptive convolution mechanism capable of handling channel variation, making it particularly suited for applications involving multi-sensor inputs, incomplete data, or cross-domain image recognition tasks.

Experimental Results and Analysis

To validate the effectiveness of DynamicConv2D, we conducted experiments on ImageNet-mini, a 50-class subset of the ImageNet dataset. ResNet-34 was used as the baseline model, and its initial convolutional layer was replaced with DynamicConv2D for comparison. The training setup included 50 epochs, a batch size of 32, SGD optimizer with momentum, and CrossEntropy Loss as the loss function. All images underwent standard normalization and data augmentation techniques including

horizontal/vertical flipping, rotation, and random distortion to improve generalization.

As shown in Table 1, although DynamicConv2D incurs significantly higher FLOPs compared to the baseline, it achieves substantial improvements in classification accuracy, precision, recall, and F1-score, demonstrating its practical advantage in handling heterogeneous input channels.

model	Accuracy	Precision	Recall	F1-score	FLOPS
Baseline	20.44%	26.8%	20.44%	18.75%	673,554,432
DynamicConv2D	25.56%	31.06%	25.56%	24.19%	2,623,360,963

Table 1: Performance comparison with RGB input.

As shown in Table 2, although performance decreases as the number of input channels drops, DynamicConv2D maintains a reasonable level of classification accuracy. Particularly in the two-channel (e.g., RG) scenario, the model achieves an accuracy of 23.11% and an F1-score of 21.24%, indicating strong channel tolerance. Even under single-channel input, though the performance drops more significantly, the model still outperforms random guessing and retains basic feature extraction and recognition capabilities.

通道組合	Accuracy	Precision	Recall	F1-score
RGB	25.56%	31.06%	25.56%	24.19%
RG	23.11%	27.84%	23.11%	21.24%
RB	20.67%	25.55%	20.67%	18.77%
GB	20.89%	23.13%	20.89%	18.55%
R	9.78%	14.04%	9.78%	8.10%
G	11.56%	21.01%	11.56%	9.91%
B	8.44%	10.54%	8.44%	7.06%

Table 2: DynamicConv2D performance under different input channel combinations.

Conclusion

Overall, DynamicConv2D delivers substantial improvements in both model performance and input flexibility. Its key advantage lies in the ability to adapt computation dynamically based on different input channel configurations, allowing the model to remain stable even under incomplete data or variable input settings. Experimental results confirm that with full RGB input, DynamicConv2D outperforms the traditional ResNet-34 in accuracy and generalization. Even when channels are partially missing, the model continues to generate reasonable predictions. Although the computational cost is higher, its adaptability to input heterogeneity makes it highly

valuable in practical applications.

Task B:

Design Principle

This model architecture integrates traditional convolutional neural networks (CNNs) with a self-attention mechanism to enhance the model's ability to capture local and global contextual features. While conventional CNNs perform well in image classification, their limited receptive field hinders the ability to capture long-range dependencies. Inspired by the Vision Transformer (ViT), we introduce a Self-Attention Block after the CNN backbone to enable global context modeling.

The design preserves CNN's strengths in local feature extraction while augmenting it with global information integration via self-attention, aiming to balance classification performance and inference efficiency.

Layer Definition

The model is modular, with each ConvBlock treated as an effective layer composed of:

- One Conv2d layer (3x3 kernel, padding=1)
- One BatchNorm2d
- One ReLU activation
- One MaxPool2d (kernel size=2)

Each ConvBlock functions as a composite layer that halves the spatial resolution and enhances abstract feature representation. After four ConvBlocks, a SelfAttentionBlock is appended. This block flattens the feature tensor of shape [B, C, H, W] into a sequence and applies nn.MultiheadAttention. It includes residual connections and LayerNorm, then reshapes the output back for classification.

Experimental Results and Analysis

We trained and tested the model on the Mini-ImageNet dataset, which consists of 50 classes and is a common benchmark for few-shot learning. The model uses CrossEntropyLoss and is optimized using Stochastic Gradient Descent (SGD). A StepLR learning rate scheduler reduces the learning rate by a factor of 0.1 every 30 epochs. Training was conducted over 50 epochs with a batch size of 32.

Table 3 shows that our CNN + Self-Attention model achieved 71.33% accuracy, 73.07% precision, and 70.74% F1-score with only 1.24 million parameters and 2.42 GFLOPs of computational cost. In comparison, ResNet-34 achieved slightly higher accuracy (74.00%) and F1-score (73.54%) but required significantly more resources

(21.8M parameters, 3.67 GFLOPs), indicating that our model offers a more resource-efficient alternative.

model	Accuracy	Precision	Recall	F1-score	FLOPS	Params
ResNet-34	74.00%	74.55%	74.00%	73.54%	3.67 GFLOPs	21.8M
CNN + Self-Attention	71.33%	73.07%	71.33%	70.74%	2.42 GFLOPs	1.24M

Table 3: Performance comparison between ResNet-34 and our model.

Ablation Study

To verify the contribution of the self-attention module, we performed an ablation study by removing the Self-Attention Block. As shown in Table 4, without the attention module, model accuracy dropped to 65.78%, and F1-score fell to 66.72%, clearly demonstrating the effectiveness of incorporating self-attention.

model	Accuracy	Precision	Recall	F1-score
CNN + Self-Attention	71.33%	73.07%	71.33%	70.74%
CNN	65.78%	66.72%	65.78%	64.80%

Table 4: Ablation study: with vs. without Self-Attention.

Conclusion

In summary, the proposed CNN-Attention hybrid architecture demonstrates competitive classification performance while maintaining extremely low parameter count and computational cost. The inclusion of self-attention significantly improves performance, making this lightweight model a strong candidate for deployment in resource-constrained environments.