Conference Paper Title*

Ching-Heng Cheng
Institute of Data Science,
Department of Statistics,
National Cheng Kung University
Tainan, Taiwan
henry918888@gmail.com

***Abstract*—This report covers the first assignment of the Deep Learning course, which consists of two tasks. The first task involves designing a method capable of handling inputs with arbitrary channels—a challenging problem that requires the model to achieve strong generalizability. The second task is to develop a shallow network with no more than four layers, aiming to achieve at least 90% of the performance of ResNet34.**

***Index Terms*—preprocessing, shallow network, generalization, ResNet**

I. Tasks A

This section talks about my method to deal with arbitrary channels input.

A. Pre-processing method

Instead of designing a specialized convolutional module, I adopt image preprocessing techniques to transform variable-channel inputs into standard 3-channel data.

Empirically, deep learning models learn complex non-linear relationships from input data. However, this data often shares a consistent structural pattern. When the combination of input channels differs, the model may suffer from structural misalignment, which in turn affects its performance. As a result, when dealing with inputs of varying channel configurations, models pretrained on natural images (e.g., RGB) may fail to generalize. Beyond degraded performance, such models are also inherently unable to process arbitrary-channel inputs.

DY-CNN [1] addresses this challenge by generating dynamic weights to adaptively integrate information across different channels. While this design increases computational costs (e.g., FLOPs and parameter count), it significantly reduces the need to train and store separate models for each channel configuration. Similarly, the Perceiver architecture [4] is capable of handling inputs with arbitrary numbers of channels. However, it lacks strong spatial inductive biases, and often underperforms on small-scale datasets.

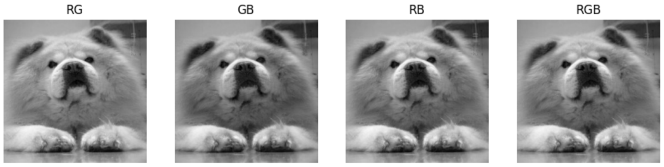

Fig. 1. Different combinations of channels. The visualizations are generated by averaging along channels to produce grayscale images.

Thus, based on the observation of similar hues in grayscale images across different combinations of R, G, and B channels (as shown in Fig.1), I convert the input images into single-channel images by averaging across the channels.

However, converting to grayscale may lead to the loss of certain features. To compensate for this, traditional image processing techniques can be helpful in enhancing contrast and revealing more details. As shown in Fig.2, I apply Contrast Limited Adaptive Histogram Equalization (CLAHE) [9] and gamma correction [7] to enhance contrast, and use a Laplacian filter [8] to extract texture details.

Finally, I concatenate these three processed channels to form a 3-channel image as the input for training. The full preprocessing pipeline is illustrated in Fig.3.

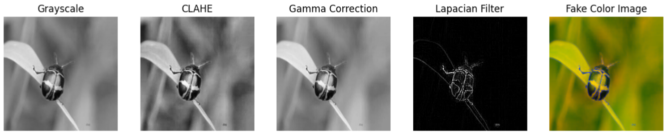

Fig. 2. Demonstration of different conventional image processing results.

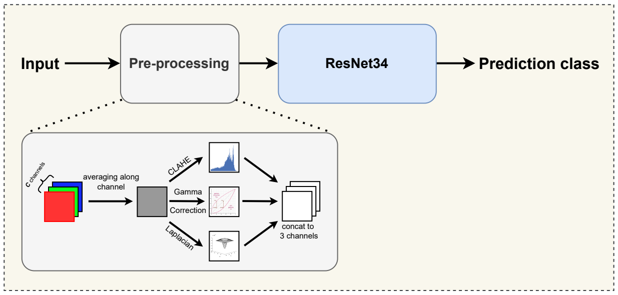

Fig. 3. The full pipeline of my method to deal with various channels input.

Table I presents the experimental results using different combinations of input channels on a model pretrained solely on RGB data. The results demonstrate that my method outperforms the baseline (i.e., pure RGB input). Moreover, even when the input consists of different channel combinations, the performance does not drop significantly, indicating good generalizability of the proposed approach.

Furthermore, since the method involves only preprocessing without modifying the model architecture, the computational

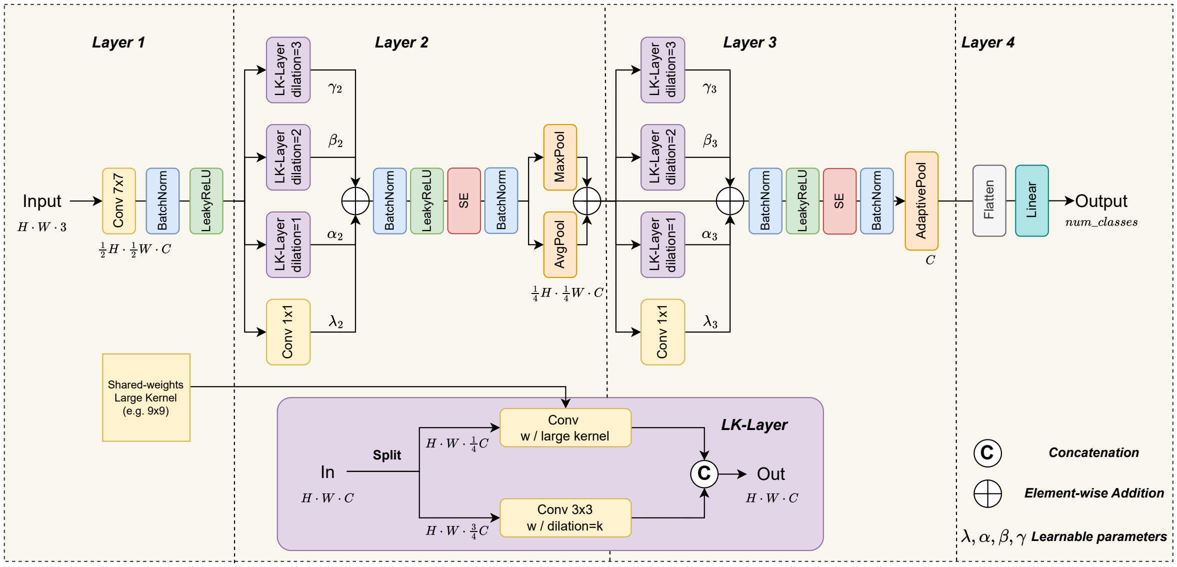

Fig. 4. My proposed architecture. Shallow network with a weight-shared large kernel.

cost does not increase. This indicates that the FLOPs and the number of parameters remain the same as the baseline.

	RGB	RG	GB	RB	R	G	B
My Method	0.7044	0.6689	0.6822	0.6889	0.6089	0.6978	0.6533
Baseline	0.6844	-	-	-	-	-	-

TABLE I
Accuracy comparison under different channel combinations. Baseline is a ResNet34 [2] model without our preprocessing method.

II. Task B

This section presents the architecture of my designed model, which uses only four layers and achieves competitive performance of a naive ResNet34.

A. Shallow Network with a Weight-Shared Large Kernel

The standard ResNet architecture uses residual connections to stabilize gradient propagation and enable deeper networks. However, increasing depth also raises parameter count and computational cost, making it less suitable for edge deployment.

Additionally, its simple additive feature fusion limits deep feature reuse, and uniform channel processing leads to suboptimal use of channel-wise information.

Several works have been proposed to address these limitations. For instance, ResNeXt [11] improves upon ResNet by introducing cardinality—the number of parallel paths within a block—through grouped convolutions, achieving higher accuracy with better parameter efficiency. Other architectures have also explored alternatives such as dynamic convolutions [1] and attention mechanisms [10] to enhance feature representation and efficiency.

Nevertheless, many of these methods still result in relatively high parameter counts and FLOPs, which remain unsuitable for lightweight deployment scenarios.

Motivated by the need for a more compact yet effective design, and building upon the ability of Weight-Shared Large Kernels [6] to capture global features efficiently, I propose a shallow network architecture, illustrate on Fig.4, consisting of only four layers. Despite its simplicity, the model achieves performance comparable to ResNet34, offering a promising trade-off between accuracy and computational efficiency.

Firstly, the input image is passed through a 7×7 convolution with a stride of 2 for downsampling.

The proposed **LK-kernel** module then splits the feature maps into two parts: $\frac{3}{4}C$ and $\frac{1}{4}C$, where C denotes the number of input feature channels. The $\frac{1}{4}C$ sub-feature is processed by a weight-shared large kernel to capture global information, while the remaining $\frac{3}{4}C$ sub-feature undergoes a local 3×3 convolution with varying dilation rates [12].

After concatenating the two sub-features, a set of learnable weights is used to compute a weighted sum of the features from different dilation branches. Furthermore, SE [3] module is adopted to dynamically adjust the channel weights.

The structures of Layer3 and Layer2 are conceptually the same, differing only in their downsampling strategy. Finally, the flattened features are passed through a linear layer to produce the final prediction.

Despite our efforts, the shallow network falls short of ResNet34's 90 % accuracy target, reaching only around 80%. However, it demands far fewer resources—requiring just 2% of ResNet34's parameters and significantly fewer FLOPs. As Table II shows, our shallow model nonetheless delivers performance comparable to ResNet34. The main drawback is its larger GPU–memory footprint during training, which stems from the use of large kernels and parallel convolutions in each layer.

TABLE II
COMPARISON OF MODEL PERFORMANCE, COMPUTATIONAL COST, AND PARAMETER SIZE.

Model	Accuracy (%)	FLOPs (G)	Params (M)
ResNet34	68.44	4.804	21.310
Ours (4-layer)	55.33	2.146	0.225

- Implementation Settings All images were resized to 256 × 256 and loaded in batches of 32 for training, validation, and testing. All models were trained for 20 epochs using the Adam [5] optimizer, with an initial learning rate of 0.001 annealed to 0.000001 via a cosine scheduler.

REFERENCES

[1] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11027–11036, 2020.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[3] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

[4] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention, 2021.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[6] Chenghao Li, Chaoning Zhang, Boheng Zeng, Yi Lu, Pengbo Shi, Qingzi Chen, Jirui Liu, Lingyun Zhu, Yang Yang, and Heng Tao Shen. Interpreting and improving attention from the perspective of large kernel convolution, 2024.

[7] Arun N Netravali. *Digital pictures: representation, compression, and standards*. Springer, 2013.

[8] Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. Local laplacian filters: edge-aware image processing with a laplacian pyramid. *Commun. ACM*, 58(3):81–91, February 2015.

[9] S.M. Pizer, R.E. Johnston, J.P. Ericksen, B.C. Yankaskas, and K.E. Muller. Contrast-limited adaptive histogram equalization: speed and effectiveness. In *[1990] Proceedings of the First Conference on Visualization in Biomedical Computing*, pages 337–345, 1990.

[10] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module, 2018.

[11] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2017.

[12] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions, 2016.