DMSRESNET: A TINY ROBUST RESNET

Shubham Singh, Xu Zhuo, Inder Khatri
New York University

ABSTRACT

In this work, we study a Distilled MixUp Squeeze Residual Network (ResNet) architecture containing 4.6 million parameters suitable for mobile applications. The model incorporates techniques such as mixUp data augmentation, integration of Squeeze-Excitation blocks, and knowledge distillation from a ResNet50 teacher model. The CIFAR-10 dataset, comprising 60,000 32x32 color images across 10 distinct classes, serves as the benchmark for evaluating the model's performance. Despite being a relatively small model the adapted ResNet architecture achieves a remarkable accuracy rate of 96.14% on the validation data, showcasing its effectiveness in image classification tasks within resource-constrained environments making it suitable for IOT applications. The code for the model is available here.

1 Introduction

ResNet[1] is one of the most widely adopted deep neural network architectures in the field of computer vision. The general trend has been to develop deeper and more complex networks to achieve higher accuracy levels. However, these advancements aimed at improving accuracy do not necessarily result in more efficient models in terms of size and speed.[2] In many real-world applications, such as embedded systems, robotics, self-driving cars, and augmented reality, recognition tasks need to be performed promptly on computationally limited platforms. Consequently, there is an increasing demand for memory-efficient models that offer competitive performance.

Related works, such as MobileNet[3] and WideNet[4], have attempted to address this challenge. In this article, we describe our effort to find a memory-efficient configuration for the ResNET model through several optimizations and modifications in the architecture.

2 Data

The cifar-10[5] dataset was used for training and validation of the model. The labels for the test dataset aren't available.

3 Methodology

Model: The model architecture is based on the works of Thakur et al.[6], and employs different techniques while relying on experimentally obtained best hyperparameters for training a ResNet. The model comprises three residual layers with a configuration of [4, 4, 3] residual blocks, utilizing convolutional kernel sizes of [3, 3, 3] and shortcut kernel sizes of [1, 1, 1]. The number of channels in each layer is set to [64, 128, 256], and batch normalization is applied throughout the network.

Training Setup:

- **Optimizer:** Stochastic Gradient Descent with momentum of 0.9 and weight decay of 0.0005.
- **Data Augmentation:** crop and horizontal flips.
- **Regularization:** Dropout with a drop probability of 0.1 was applied to combat overfitting.
- **Learning Rate and Schedule:** Started with a learning rate of 0.1 and employed a CosineAnnealing LR scheduler.

- **Batch Size and Epochs:** A batch size of 128 and training for up to 200 epochs.

ResNET:
Introduced in 2015 Residual Networks (ResNet) drastically improved the training of deep neural networks by solving the vanishing gradient problem through the introduction of residual connections. These connections are shortcuts that allow gradients to flow through the network by skipping one or more layers, which prevents the gradients from vanishing and allows for significantly deeper networks.

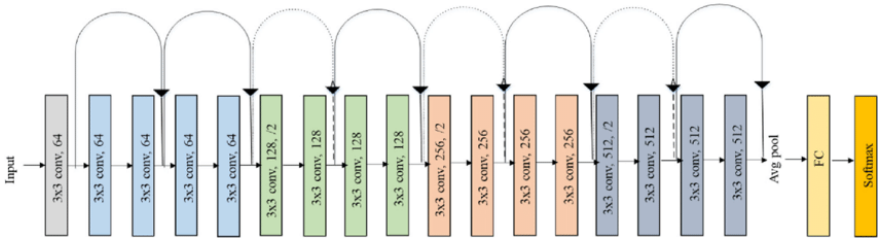

Figure 1: ResNET - 18

ResNet architectures are built using two primary types of blocks: basic blocks and bottleneck blocks. The basic block is usually utilized in ResNet-18 and ResNet-34, consisting of two 3x3 convolutional layers and designed for networks with fewer layers. This block structure simplifies the network design and reduces computational requirements while still benefiting from the residual connections that add inputs directly to outputs, facilitating training and improving performance.

In contrast, the bottleneck block, used in deeper ResNets like ResNet-50, ResNet-101, and ResNet-152, includes three layers (1x1, 3x3, and 1x1 convolutions). The 1x1 layers are responsible for reducing and then increasing (projecting) dimensions, thus managing the network's complexity and computational load more efficiently. These blocks make it feasible to extend the network depth significantly without a substantial increase in resource requirements.

3.1 Convolutional Block (BasicBlock):

The Convolutional Block, also known as BasicBlock, serves as the fundamental building block of the ResNet architecture. It comprises two convolutional layers, each followed by batch normalization to stabilize training. The first convolutional layer employs a kernel size of 3x3 and a specified stride, while the second convolutional layer maintains the input size. Additionally, a shortcut connection allows the network to bypass layers, aiding in the flow of gradients during training. This block's design prioritizes computational efficiency while facilitating effective feature extraction.

3.2 Squeeze-and-Excitation (SE) Block:

The Squeeze-and-Excitation Block [7] enhances feature recalibration within the network by adaptively weighting channel-wise feature responses. It consists of two convolutional layers followed by a global average pooling operation to capture channel-wise statistics. The first convolutional layer reduces the number of channels, enabling computational efficiency, while the subsequent activation function introduces non-linearity. The final convolutional layer restores the original channel dimensionality, and a sigmoid activation function scales the features, emphasizing informative channels. This block fosters the network's ability to focus on salient features, thereby improving classification performance.

3.3 Residual Layers:

The Residual Layers encapsulate multiple instances of the Convolutional Block, forming the core structure of the ResNet architecture. Each Residual Layer comprises a sequence of BasicBlocks, with the number of blocks determined by the specified architecture configuration. The stride parameter controls the downsampling of feature maps within each layer, facilitating the extraction of features at multiple scales. The inclusion of skip connections mitigates the vanishing gradient problem, enabling effective gradient flow during training. This hierarchical arrangement of Residual Layers enables the network to learn increasingly abstract representations of the input data, leading to improved classification performance.

Mixup Training:
Mix-up augmentation, introduced by Zhang et al. in 2017 [8], is an innovative technique designed to improve the

generalization of image classification models by training on linear combinations of image pairs and their labels. The corresponding labels are mixed in the same manner, promoting the model to learn more robust features that are not tied to specific training examples.
The primary benefits of using Mix-up include reducing the risk of overfitting by smoothing the label space and enhancing model robustness against noisy data. It also stabilizes the training process, leading to more reliable convergence. Implementing Mix-up in training convolutional neural networks for image recognition not only boosts accuracy but also aids the model in achieving better generalization across varied datasets. This method's efficacy underscores its utility in complex visual tasks, making it a vital tool in advanced deep-learning applications.

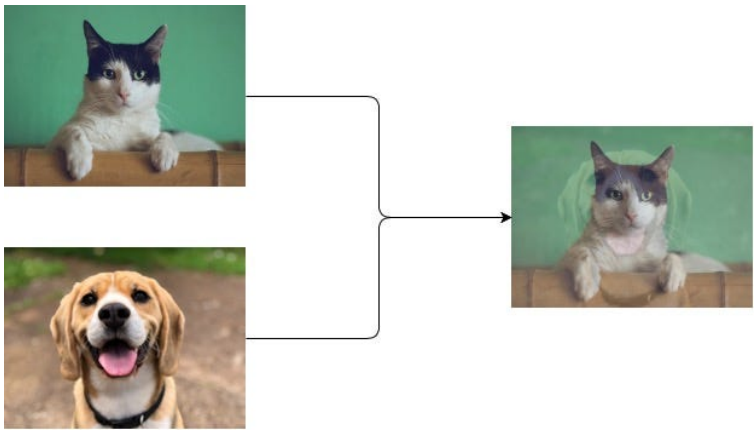

Figure 2: Mixup Training

Knowledge Distillation:
Knowledge distillation [9] is a technique used for transferring knowledge from a large, complex teacher model to a smaller, more efficient one known as a student model. It involves training the student model to mimic the soft output (class probabilities) of the teacher model rather than the hard targets (actual class labels). The soft probabilities contain richer information about the input space as they reflect the confidence of the teacher model across all classes.

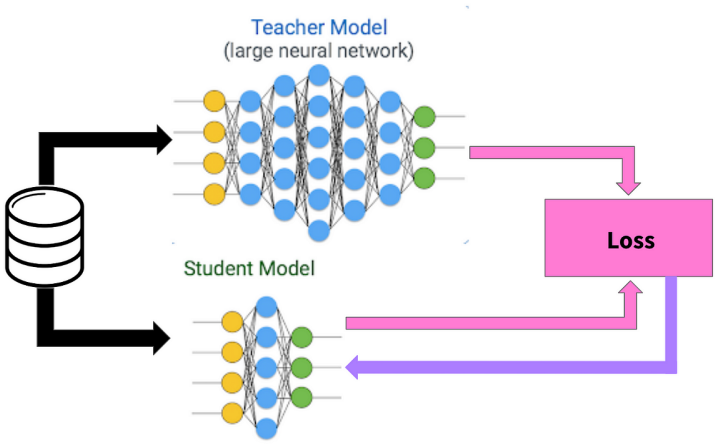

Figure 3: Knowledge Distillation

The primary equation for knowledge distillation is:

$$L = (1 - \alpha) \times H(y, \sigma(z/T)) + \alpha \times H(y, \sigma(z)) \tag{1}$$

where H is the cross-entropy, y is the true label, z is the student's logits, σ denotes the softmax function, T is the temperature scaling the logits, and α balances the two terms. Higher T values produce softer probabilities, facilitating

better student learning. This technique not only enhances model efficiency but also retains performance, making it pivotal for deploying deep learning models in resource-constrained environments.

$$\text{CrossEntropyLoss}(\text{pred}, y) = -\sum_{i} y_i \log(\text{pred}_i) \tag{2}$$

$$\begin{aligned} \text{mixupLoss} = \lambda \cdot \text{CrossEntropyLoss}(\text{pred}, y_a) \\ + (1 - \lambda) \cdot \text{CrossEntropyLoss}(\text{pred}, y_b) \end{aligned} \tag{3}$$

4 Results and Discussion

The performance of the proposed Distilled MixUp Squeeze Residual Network (DMSResNet) architecture , which has 4,697,742 parameters, was evaluated on the CIFAR-10 dataset, considering various configurations and optimization techniques. Table 1 summarizes the comparative results of different models, showcasing their train accuracy, validation accuracy, and test accuracy. The baseline ResNet model achieved a train accuracy of 100%, a validation accuracy

Model Name	Train Acc.	Validation Acc.	Test Acc.
ResNET + SE	99.99%	95.51%	85.90%
ResNET + SE + MixUp	52.43%	96.56%	**86.91%**
KDistillation DMSResNET	52.90%	96.14%	**86.96%**

Table 1: Model comparison

of 81.79%, and a test accuracy of 63.4%. The integration of Squeeze-and-Excitation (SE) blocks into the ResNet architecture led to significant improvements, with the ResNet model augmented with SE blocks achieving a train accuracy of 99.99%, a validation accuracy of 95.51%, and a test accuracy of 85.9%. Further enhancements were achieved by incorporating MixUp data augmentation into the model training process. The ResNet model augmented with both SE blocks and MixUp achieved a train accuracy of 52.43%, a validation accuracy of 96.56%, and a test accuracy of 86.91%. Moreover, the application of knowledge distillation (KD) techniques, where the model learns from a larger teacher model, resulted in comparable performance with MixUp augmentation. The KD-enabled DMSResNet achieved a train accuracy of 52.90%, a validation accuracy of 96.14%, and a test accuracy of 86.96%.

The experimental results demonstrate the efficacy of the proposed DMSResNet architecture in achieving high accuracy rates on the CIFAR-10 dataset while adhering to stringent parameter constraints. The integration of SE blocks, MixUp augmentation, and knowledge distillation techniques collectively contribute to the model's improved performance, making it suitable for resource-constrained environments where memory-efficient models are essential.

References

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

[2] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[3] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

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

[5] Alex Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009.

[6] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. 55(12), 2023.

[7] Aditya Thakur, Harish Chauhan, and Nikunj Gupta. Efficient resnets: Residual network design. *arXiv preprint arXiv:2306.12100*, 2023.

[8] Fuzhao Xue, Ziji Shi, Futao Wei, Yuxuan Lou, Yong Liu, and Yang You. Go wider instead of deeper, 2021.

[9] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.