LaTeX Author Guidelines for CVPR Proceedings

Anonymous CVPR submission

Paper ID *****

Abstract

In this study we evaluate the performance of four deep learning architecture that is CNN(Convolutional Neural Network), ResNet18, AlexNet and MobileNet on the CIFAR-10 image classification dataset. The dataset consists of low resolution images across ten diverse classes which is challenging to accurate classification. Each architecture was customized with regularization techniques like dropout and batch normalization and trained with data augmentation to enhance generalization and also the learning rate scheduling was employed to optimize convergence. Performance metrics including accuracy and loss were recorded for all models on training, validation, and test sets. Results indicate that among the four models CNN performs well based on test accuracy. This study gives insights on strengths and limitations of various architectures for image classification.

1. Introduction

Among various image classification CIFAR-10 dataset is widely used for its simplicity and versatility. Comprising 60,000 color images across ten classes in CIFAR-10 poses unique challenges due to its small image size and diverse content, requiring models to effectively extract and classify visual features with limited pixel resolution. Deep learning architectures particularly Convolutional Neural Networks (CNNs) have demonstrated significant advancements in tackling such tasks. In recent years architectures like ResNet, MobileNet and AlexNet have introduced innovations such as residual learning, depth wise separable convolutions, and parameter efficiency, enhancing model accuracy and generalization. In thsi study we evaluate the performance of four architectures that is CNN, ResNet18, MobileNet and AlexNet on the CIFAR-10 dataset. Each model represents a different approach to image classification, ranging from conventional CNN layers to more sophisticated architectural innovations aimed at improving efficiency and accuracy. The purpose of this study is to compare these architecture in terms of their accuracy and loss and see which model performs well. We employ regularization techniques and also dropout and batch normalization with data augmentation to enhance model robustness. By training, validating, and testing each model on CIFAR-10 we aim to assess their strengths and limitations providing insights into the suitability of each architecture

1.1. Methodology

In this study we implemented and trained four deep learning architectures that is CNN, ResNet18, MobileNet, and AlexNet on the CIFAR-10 dataset to evaluate their effectiveness in image classification. Each model was configured with specific layers, regularization, and data augmentation techniques to optimize performance and generalization. The steps included in the methodology are data preparation, model architectures, training strategy, and evaluation the model performance.

1.1.1 Data Preparation

The CIFAR-10 dataset contains 60,000 32x32 RGB images across ten classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck) with a predefined training set of 50,000 images and a test set of 10,000 images. Each image was preprocessed as follows:

* Normalization: Pixel values were scaled to the [0,1] range by dividing by 255.0 which will help in stabilizing gradient descent.

* One-Hot Encoding: The class labels were converted to one-hot vectors as this approach is essential for categorical cross-entropy loss.

* Train-Validation Split: To allow model tuning and validation during training the training dataset was further split into 80% for training and 20% for validation.

1.1.2 Models

1.1.3 CNN (Convolutional Neural Network) Model:

we designed and trained a CNN for the CIFAR-10 image classification task. The CNN model consists of three main convolution blocks where each incorporating batch normalization and dropout to improve regularization and prevent overfitting. The architecture details are as follows:

a. Model Architecture:

i. First Convolution Block:

* Two convolutional layers with 64 filters and a 3x3 kernel size, using ReLU activation and L2 regularization.

* Batch normalization is applied after each convolutional layer to normalize activations, accelerating convergence.

* Max pooling with a 2x2 pool size reduces spatial dimensions.

* Dropout with a rate of 0.3 is added to reduce overfitting by randomly deactivating neurons.

ii. Second Convolution Block:

* Two convolutional layers with 128 filters and a 3x3 kernel size, using ReLU activation and L2 regularization.

* Batch normalization is applied after each convolutional layer.

* Max pooling with a 2x2 pool size reduces spatial dimensions.

* Dropout with a rate of 0.4 to further reduce overfitting.

iii. Third Convolution Block:

* Two convolutional layers with 256 filters and a 3x3 kernel size, using ReLU activation and L2 regularization.

* Batch normalization after each convolutional layer.

* Max pooling with a 2x2 pool size.

* Dropout with a rate of 0.5 for enhanced regularization.

iv. Fully convolution Layers:

* After flattening, a dense layer with 512 units, ReLU activation, and L2 regularization is added.

* Batch normalization and a dropout rate of 0.5 are applied.

* The final layer is a softmax layer with 10 units, corresponding to the 10 classes in CIFAR-10

b. Model compilation and training:

i. Compilation: The model was compiled with the Adam optimizer categorical cross-entropy loss and accuracy as the evaluation metric. A reduced learning rate of 0.0001 was chosen to facilitate gradual convergence.

ii. Data Augmentation: To increase the diversity of training images data augmentation was applied using the following transformations:

* Rotation (up to 15 degrees)

* Horizontal shifts (up to 10% of the image width)

*Vertical shifts (up to 10% of the image height)

* Horizontal flips.

iii. Callbacks:

* Early Stopping: Monitors validation loss with a patience of 8 epochs to stop training if performance stagnates and preventing overfitting.

* Learning Rate Reduction on Plateau: Reduces the learning rate by a factor of 0.5 if the validation loss plateaus for 4 epochs, with a minimum learning rate of 1e-6.

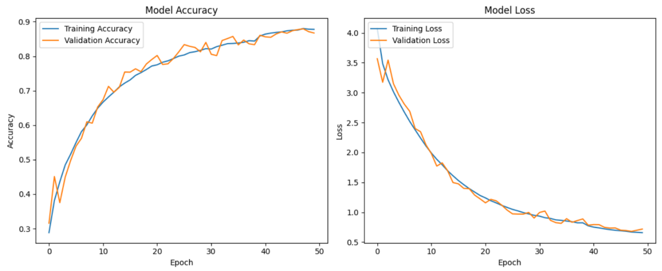

Figure 1. CNN Model.

c. Evaluation: After training the model was evaluated on the test set achieving a test accuracy of approximately 87.39% and a test loss of 0.6944. The learning curves (shown in the figure 1) indicate stable convergence with minimal overfitting, as the validation and training accuracy/loss remain closely aligned.

1.1.4 ResNet18 Model:

we implement ResNet18 architecture for CIFAR-10 image classification. As ResNet18 is known for using residual connections to facilitate the training of deep networks also overcoming the vanishing gradient problem. This architecture enables effective feature extraction through multiple layers without suffering from degradation in performance due to increased depth. The architecture details are as follows:

a. Model Architecture:

i. Initial Convolutional layer:

The input to the network which consists of 32x32 RGB images is first passed through a convolutional layer with 64 filters a kernel size of 3x3 and a stride of 1. This layer is followed by batch normalization and a ReLU activation function which helps maintain stable gradients and accelerates training.

ii. First residual Block:

Two convolutional layers with 64 filters and 3x3 kernels each followed by batch normalization and ReLU activation. A shortcut connection which directly connects the input of the block to its output. Dropout with a rate of 0.5 is applied after the block to reduce overfitting.

iii. Second Residual Block:

For each layer in this block the filter sizes increase progressively (128, 256 and 512) with stride 2 in the first layer of each sub-block to reduce spatial dimensions. Each layer has batch normalization, ReLU activation, and a shortcut connection. Dropout 0.5 is applied to improve regularization.

iv. Final Layers:

* A global average pooling layer reduces the spatial dimensions which results in a compact representation.

* A fully connected dense layer with softmax activation provides classification output across the 10 CIFAR-10 classes.

b. Model Compilation and Training:

i. Compilation: The model was compiled with the RMSprop optimizer categorical cross-entropy loss and accuracy as the metric. The learning rate was initially set to 0.0001 for gradual convergence.

ii Learning Rate Scheduling: A learning rate scheduler was implemented to reduce the learning rate by a factor of 0.1 after 30 epochs, helping the model fine-tune towards convergence in the later stages.

iii Data Augmentation: To enhance generalization, data augmentation was applied using:

* Rotation up to 10 degrees

* Width and height shifts up to 5% of the image size

*Horizontal flips.

iv Early Stopping: Early stopping was set to monitor validation loss with a patience of 5 epochs to prevent overfitting.

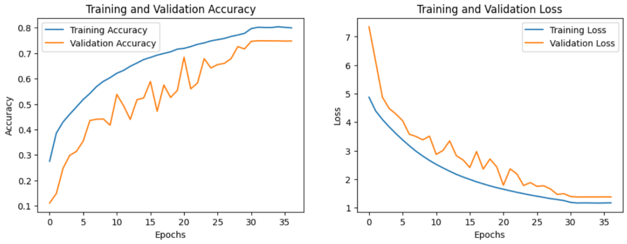

Figure 2. ResNet18 Model.

c. Evaluating: After training the model was evaluated on the test set achieving a test accuracy of approximately 85.49% and test loss of 0.4954. The learning curves (shown in the figure 2) show that the model converges effectively with a minor gap between training and validation accuracy that indicats limited overfitting.

1.1.5 AlexNet Model

we implemented a modified AlexNet architecture to classify images from the CIFAR-10 dataset to fit for 32x32 input size. This architecture includes several convolutional layers for feature extraction also followed by fully connected layers for classification. The model adopt regularization techniques and data augmentation to improve performance.

a. Model Architecture:

i. Convolution Layers:

* Layer 1: The input layer receives images of size 32x32x3. This layer applies a convolution with 96 filters, 3x3 kernel, stride 1, and ReLU activation and followed by batch normalization to normalize the activations and accelerate convergence. A max-pooling layer with a 2x2 pool size and a stride of 2 is applied to downsample the feature maps.

* Layer 2: The second layer consists of 256 filters with 3x3 kernel, stride 1, and ReLU activation. Batch normalization is used again to stabilize training and followed by a max-pooling layer with a 2x2 pool size and a stride of 2.

* Layer 3: This layer uses 384 filters with 3x3 kernel and ReLU activation also with batch normalization. As of the previous layers no pooling is applied retaining the spatial dimensions for deeper feature extraction.

* Layer 4: The fourth layer contains 384 filters with 3x3 kernel and ReLU activation and followed by batch normalization. This layer continues to expand feature complexity capturing more complicate patterns in the images.

Layer 5: The fifth layer has 256 filters 3x3 kernel and ReLU activation along with batch normalization and a max-pooling layer with a 2x2 pool size and stride 2. This pooling reduces the spatial dimensions and preparing the features for the fully connected layers.

ii. Fully Connected Layers:

* Flattening: The output from the final convolutional layer is flattened to form a 1D vector.

* Dense Layers: Two dense layers with 4096 neurons each and ReLU activation are applied in sequence. Dropout with a rate of 0.3 is applied after each dense layer to reduce overfitting by randomly dropping units during training.

* Output Layer: The final dense layer has 10 neurons and the corresponding to the 10 CIFAR-10 classes with a softmax activation for multi-class classification

b. Model Compilation and Training:

i. Compilation: The model was compiled using the Adam optimizer with an initial learning rate of 1e-5 to ensure gradual convergence. The categorical cross-entropy loss function was used for multi-class classification and accuracy was selected as the evaluation metric.

ii. Learning Rate Scheduling: A learning rate scheduler was implemented to reduce the learning rate by half every 10 epochs to ensuring the model learns quickly at first and then fine-tunes its weights as training progresses.

iii Data Augmentation: The augmentation was implemented using Keras's ImageDataGenerator To further enhance the models robustness and prevent overfitting data augmentation was applied using the following techniques:

* Width and Height Shifts: Images were randomly shifted horizontally and vertically by up to 10% of the image size for simulating variations in object positioning.

* Horizontal Flipping: Random horizontal flips were applied to introduce variations in orientation.

c. Evaluation: After training the model was evaluated on the test set, achieving a test accuracy of approximately 75.88% and a test loss of 0.7053. The learning curves (shown in the figure 3) show that the model was able to generalize reasonably well on the CIFAR-10 dataset also it

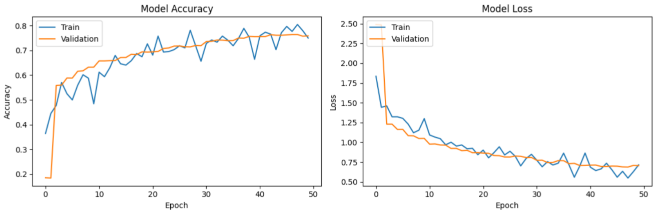

Figure 3. AlexNet Model.

did not perform as effectively as deeper architectures like ResNet18.

1.1.6 MobileNet Model

we utilized a pre-trained MobileNet model to classify images from the CIFAR-10 dataset. MobileNet is known for its efficiency and lightweight design and achieved through depthwise separable convolutions that reduce computational complexity. By leveraging a pre-trained model with additional fine-tuning this architecture provides a balance between accuracy and efficiency.

a. Model Architecture:

i. Pre-trained MobileNet as Base Model:

* We used MobileNet with pre-trained weights on ImageNet as the base model. The top layers of MobileNet were removed to allow for a custom classification head specific to CIFAR-10.

* The base model was set to accept 32x32 RGB images, matching the CIFAR-10 image dimensions.

ii. Fine-tuning of Base Model:

* The last 10 layers of the MobileNet base model were unfrozen to enable fine-tuning and allowing the network to adapt to CIFAR-10 features while retaining the learned weights from ImageNet. This strategy leverages the general visual features from ImageNet and adapts them to CIFAR-10.

iii Custom Classification Head:

* A Global Average Pooling layer was added to aggregate spatial information from the base models output feature maps and resulting in a compact representation.

* A fully connected layer with 1024 neurons and ReLU activation was added with followed by a Dropout layer 0.5 for regularization and reducing overfitting.

* The Dense layer with 10 neurons and softmax activation was used to produce class probabilities for the 10 CIFAR-10 classes.

b. Model Compilation and Training:

i. Compilation: The model was compiled with the Adam optimizer and learning rate of 1e-4 and modest rate that allows gradual fine-tuning. Categorical cross-entropy loss was used for multi-class classification with accuracy as the evaluation metric.

ii. Data Augmentation:

* Rotation: Images were randomly rotated by up to 15 degrees.

* Width and Height Shifts: Horizontal and vertical shifts up to 10% of the image size were applied.

* Horizontal Flip: Random horizontal flips added orientation variance to the dataset.

c. Training Strategy: The model was trained for 50 epochs using a batch size of 64. By fine-tuning only the last few layers of MobileNet the model can adapt to the CIFAR-10 dataset with a reduced risk of overfitting. The training and validation accuracy/loss were monitored and training progress was tracked to ensure stable convergence.

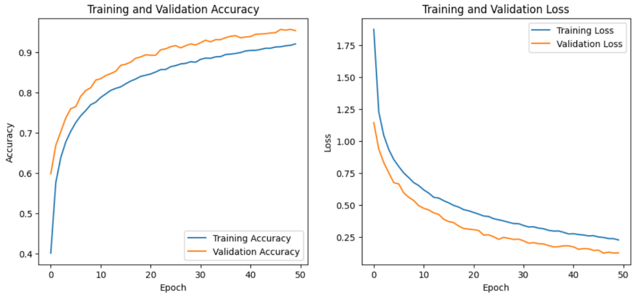

Figure 4. MobileNet Model.

d. Evaluation: After training the model was evaluated on the test set, achieving a test accuracy of approximately 85.49% and a test loss of 0.4954. The learning curves (shown in the figure 4) show that MobileNet even when adapted with fine-tuning will performs effectively on small-scale datasets like CIFAR-10.

1.2. Best Model Analysis

Model	Test Accuracy	Test Loss
CNN Model	0.87	0.69
ResNet50 Model	0.85	0.49
Modified AlexNet	0.76	0.71
MobileNet	0.85	0.49

Table 1. Model Comparison Table

The CNN Model achieved the highest test accuracy of 0.87 making it the best-performing model based on accuracy. Although the ResNet18 and MobileNet models had lower test losses (0.49) and CNN model (0.69) they did not surpass the CNN model in accuracy. This indicates that the CNN model balances accuracy and generalization effectively on the CIFAR-10 dataset making it the most suitable choice among the evaluated architectures for this task.

References

[1] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324. https://ieeexplore.ieee.org/document/726791

[2] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778). https://ieeexplore.ieee.org/document/7780459

[3] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (Vol. 25, pp. 1097–1105).

[4] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. https://arxiv.org/abs/1704.04861

[5] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), 1-48.

1.3. Link To Github

https://github.com/Anusha0113/Deep-Learning-Fundamentals