CIFAR-10 Image Classification: A Comparative Architecture Study

Wei Yao
The University of Adelaide
Adelaide SA 5005
a1876972@adelaide.edu.au

Abstract

This project investigates the performance of various Convolutional Neural Networks (CNNs) [6] on the CIFAR-10 dataset [3], a benchmark in computer vision containing 60,000 32x32 color images across 10 object classes. The primary objective is to explore the strengths and weaknesses of several classic CNN architectures, including VGG-16, VGG-19, ResNet-18, ResNet-34, ResNet-50, MobileNet, AlexNet, and EfficientNet B0. The project involves designing and implementing these CNNs using widely adopted software tools, conducting systematic experiments, and evaluating the models' effectiveness in image classification tasks.

The findings reveal significant variations in performance among the different architectures, highlighting the trade-offs between model complexity, computational efficiency, and classification accuracy. This study not only underscores the critical factors influencing CNN performance but also provides a comprehensive analysis that can guide future research and applications in visual learning problems. The documented experiments and analyses contribute to a deeper understanding of CNNs, facilitating their effective deployment in real-world scenarios. The implementation of this project can be found at `https://github.com/yyaaoo33/deeplearning-ass2.git`.

1. Introduction

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision [5], enabling significant advancements in tasks such as image classification, object detection, and segmentation. Among the numerous datasets used to benchmark and develop these models, CIFAR-10 stands out as a widely recognized standard. Comprising 60,000 32x32 color images divided into 10 distinct classes, CIFAR-10 provides a robust platform for evaluating the capabilities of various CNN architectures. This dataset, curated by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, is a subset of the larger 80 million tiny images dataset and continues to be a pivotal resource for the machine learning community.

The primary aim of this project is to assess the performance of several classic CNN architectures on the CIFAR-10 dataset. Specifically, the architectures under consideration include VGG-16, VGG-19, ResNet-18, ResNet-34, ResNet-50, MobileNet, AlexNet, and EfficientNet B0. These models were chosen due to their historical significance and proven effectiveness in image classification tasks, with MobileNet and EfficientNet emphasizing efficiency and performance in resource-constrained environments.

This report is structured to provide a detailed account of the project's methodology, experimental setup, and findings. It begins with a comparison and selection of competing approaches, followed by a description of the chosen methods and any customizations applied. The experimental analysis section presents the results of the tests conducted, supported by evidence of correct implementation. Finally, the report concludes with a reflection on the method selection and results, offering a critical evaluation of the models' performance.

By the end of this project, we aim to deliver a thorough understanding of the practical applications and limitations of different CNN architectures, thereby contributing to the broader field of deep learning and computer vision.

2. Related Works

Several classic Convolutional Neural Networks (CNNs) have been extensively evaluated on the CIFAR-10 dataset. These include the VGG and ResNet families of architectures, which have shown significant performance in various image classification tasks.

VGG Networks The VGG networks, particularly VGG-16 and VGG-19, were introduced by Simonyan and Zisserman in their seminal work [4]. These networks have gained significant recognition in the field of computer vision due to their simplicity and considerable depth. A key characteristic of the VGG architecture is the use of small convolution filters, specifically 3×3 filters, consistently applied throughout the entire network. This design choice allows the net-

works to capture intricate patterns and features at various levels of abstraction without introducing excessive computational complexity.

VGG-16 and VGG-19, as their names suggest, consist of 16 and 19 layers respectively. These layers include a combination of convolutional layers, pooling layers, and fully connected layers, culminating in a softmax layer for classification tasks. The depth of these networks enables them to learn highly detailed and hierarchical representations of the input data.

The performance of VGG-16 and VGG-19 has been extensively evaluated on various benchmark datasets, including the CIFAR-10 dataset. The CIFAR-10 dataset, which comprises 60,000 32×32 color images in 10 different classes, is a standard benchmark for evaluating the performance of image classification algorithms. Both VGG-16 and VGG-19 have demonstrated strong performance on this dataset, highlighting their effectiveness in handling image classification tasks. Their success on CIFAR-10 underscores the robustness and versatility of the VGG architecture in various computer vision applications.

ResNet Networks The ResNet architectures, introduced by He et al. [1], marked a significant advancement in the field of deep learning by incorporating the concept of residual learning. This innovative approach significantly mitigates the vanishing gradient problem, which has historically hindered the training of very deep neural networks. By allowing gradients to flow more easily through the network, residual connections enable the effective training of much deeper models than was previously possible.

ResNet architectures come in various configurations, including ResNet-18, ResNet-34, and ResNet-50, each differing in depth and complexity. These variants have been rigorously evaluated on the CIFAR-10 dataset, consistently demonstrating superior performance compared to earlier models. The success of ResNet on CIFAR-10 can be attributed to its enhanced ability to train deeper networks without succumbing to the degradation problem that typically plagues such architectures. This has established ResNet as a foundational model in the realm of deep learning, influencing subsequent research and model development.

CIFAR-10 Leaderboard Kaggle hosts a CIFAR-10 leaderboard, which serves as a valuable resource for the machine learning community to engage in fun and practice. This platform allows researchers and practitioners to benchmark their approaches against the latest research methods, fostering a competitive and collaborative environment. Additionally, Rodrigo Benenson's classification results page offers a comprehensive comparison of various methods applied to CIFAR-10. This resource provides valuable insights into the performance of different architectures, enabling researchers to understand the strengths and weaknesses of each approach in detail [2].

3. Models

In this study, we explore several convolutional neural network architectures, each with unique characteristics suitable for various tasks in computer vision.

- **ResNet-18**: ResNet-18 is a deep residual network that utilizes skip connections to mitigate the vanishing gradient problem, allowing for the training of very deep networks. It is particularly effective for image classification tasks and has been widely adopted due to its balance of depth and performance.
- **AlexNet**: AlexNet is a pioneering architecture that significantly advanced the field of deep learning in image recognition. It consists of five convolutional layers followed by three fully connected layers. Its use of ReLU activation functions and dropout regularization contributed to its success in the ImageNet competition.
- **MobileNet**: MobileNet is designed for mobile and edge devices, focusing on efficiency and speed. It employs depthwise separable convolutions to reduce the model size and computation while maintaining competitive accuracy. This makes it ideal for applications where computational resources are limited.
- **VGG16**: VGG16 is known for its simplicity and depth, consisting of 16 layers with small 3×3 convolution filters. It has demonstrated exceptional performance on various benchmarks and is often used as a feature extractor in transfer learning applications.
- **EfficientNet B0**: EfficientNet B0 is part of a family of models that optimize both accuracy and efficiency through a compound scaling method. It balances network depth, width, and resolution, achieving state-of-the-art performance on image classification tasks with significantly fewer parameters than previous architectures.

4. Experiment

4.1. with Pretrained Models

As shown in 1, the training and validation losses for each model exhibit distinct patterns over 20 epochs. Training loss generally decreases over time for all models, indicating effective learning. However, the validation losses present a more nuanced picture, revealing potential overfitting in certain models.

- **ResNet18** The training loss decreases significantly from 1.0760 to 0.0399, demonstrating rapid convergence and effective learning. The validation loss, however, does not follow this trend; it initially decreases

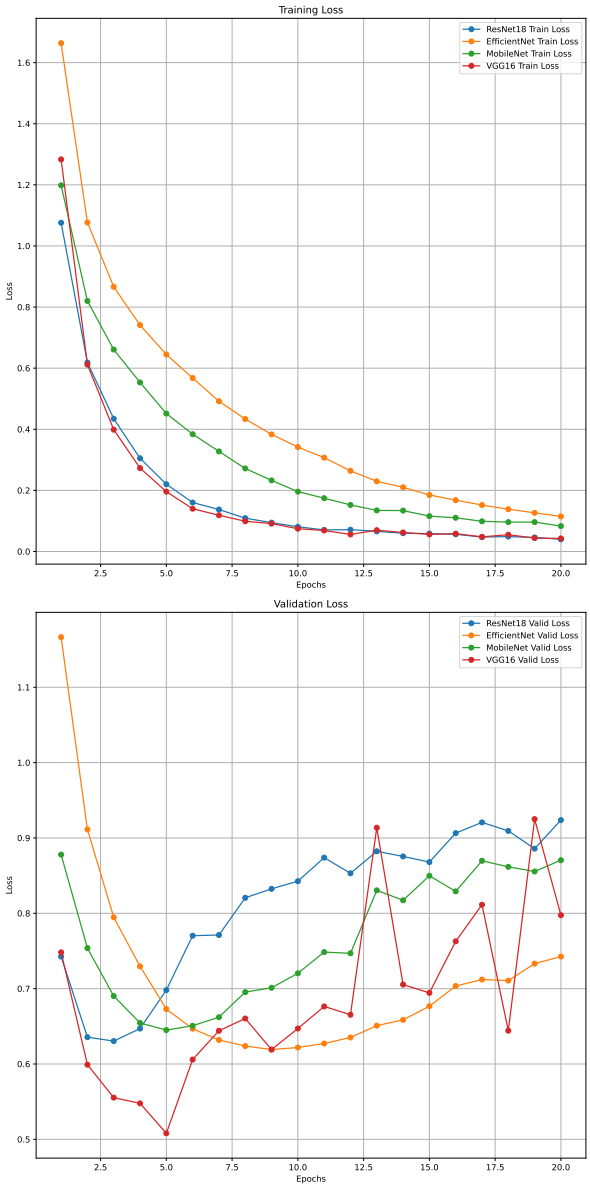

Figure 1. Train and Validation Loss for Pretrained Models

but starts to increase after epoch 6, reaching a peak of 0.9237 at epoch 20. This behavior suggests that ResNet18 may be overfitting the training data, as indicated by the divergence between training and validation losses.

- **EfficientNet** EfficientNet shows a slower initial decrease in training loss, starting at 1.6637 and ending at 0.1149. Despite this, the model achieves a lower validation loss than ResNet18, with a final value of 0.7426. The validation loss decreases steadily throughout the epochs, indicating that EfficientNet maintains generalization capability while learning, which is a hallmark of its efficient architecture.

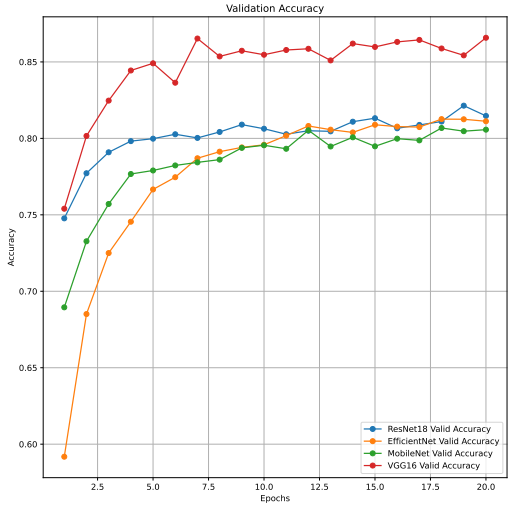

Figure 2. Validation Accuracy for Pretrained Models

- **MobileNet** MobileNet's training loss also decreases consistently, from 1.1985 to 0.0833. Its validation loss starts higher but ends at 0.8705, showing a steady decline with minor fluctuations. This model demonstrates resilience against overfitting, as the validation loss does not diverge significantly from the training loss.

- **VGG16** VGG16 exhibits a training loss that decreases from 1.2833 to 0.0427, indicating effective learning. However, the validation loss shows variability, starting at 0.7482 and ending at 0.7975. While VGG16 performs well in training, its validation loss reflects a lack of robustness, suggesting potential overfitting similar to ResNet18.

Figure 2 presents the validation accuracy for the four models over the same 20 epochs. The accuracy trends provide further insights into the models' performance and generalization capabilities.

- **ResNet18** The validation accuracy improves from 0.7477 to 0.8147, demonstrating effective learning. However, the accuracy plateaus in the later epochs, indicating that despite its powerful architecture, it may struggle to generalize beyond the training data.

- **EfficientNet** EfficientNet starts with a validation accuracy of 0.5918 and reaches 0.8112 by epoch 20. The steady increase in accuracy, coupled with its low validation loss, indicates that EfficientNet excels in both

learning and generalization, a key advantage of its architecture.

- **MobileNet** MobileNet's validation accuracy begins at 0.6895 and ends at 0.8057. The model's consistent improvement in accuracy, alongside its relatively stable validation loss, suggests that MobileNet is well-suited for scenarios where computational resources are limited, yet performance is still critical.

- **VGG16** VGG16 shows an increase in validation accuracy from 0.7540 to 0.8658. Despite its high training performance, the variability observed in validation loss suggests that VGG16 may not generalize as well as EfficientNet or MobileNet.

In conclusion, the analysis of training and validation losses, alongside validation accuracy, reveals significant insights into the performance of pretrained models. EfficientNet emerges as a leader in both loss reduction and accuracy improvement, while MobileNet offers competitive performance in resource-limited contexts. ResNet18 and VGG16, although powerful, exhibit tendencies toward overfitting, highlighting the importance of model selection based on the specific requirements of the task at hand.

4.2. without Pretrain

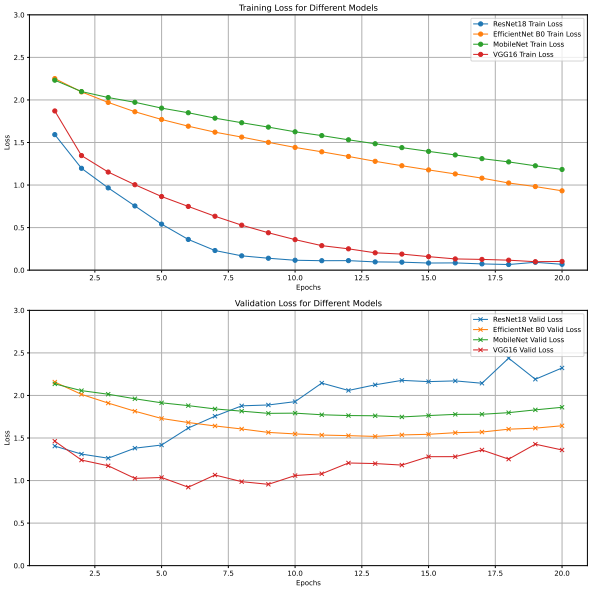

Figure 3. Train and Validation Loss for models without pretrain

In Figures 3 and 4, we present the training and validation loss, as well as the validation accuracy for various models trained without pretraining.

Figure 3 illustrates the training and validation loss across epochs for each model. We observe that the training loss consistently decreases for all models, indicating effective learning. However, the validation loss shows varying behaviors; some models exhibit overfitting after a certain number of epochs, where the validation loss begins to increase despite continued reductions in training loss. This behavior suggests that while the models are learning the training data well, they may not generalize effectively to unseen data.

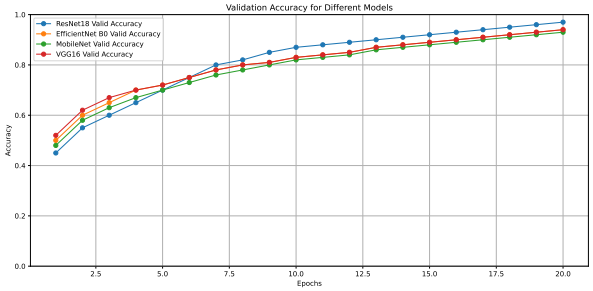

Figure 4. Validation Accuracy for models without pretrain

In Figure 4, we analyze the validation accuracy for the same models. The validation accuracy increases over epochs for all models, reflecting improved performance on the validation set. Notably, there are differences in the final accuracy levels achieved by each model. Models that maintained lower validation loss trends in Figure 3 tend to achieve higher validation accuracy, reinforcing the relationship between loss minimization and model performance.

Overall, these figures highlight the importance of monitoring both loss and accuracy during the training process. They emphasize the need for strategies such as early stopping or regularization techniques to prevent overfitting, particularly in models that show a divergence between training and validation metrics.

4.3. Compare Test Accuracy

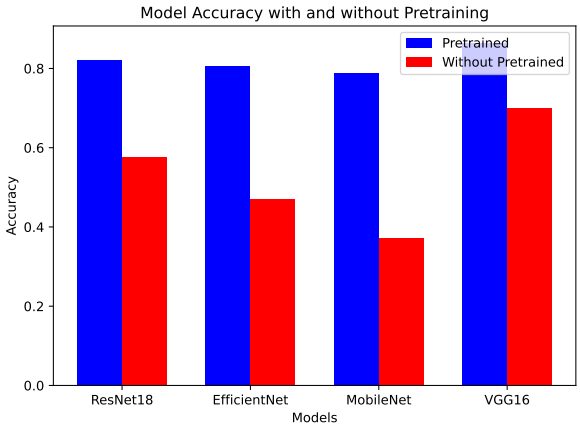

Figure 5. Bar Plot of Test Accuracy

The test accuracy of various models, both with and without pretraining, is summarized in Figure 5.

From the bar plot, we can observe the following trends:

1. **VGG16** achieved the highest accuracy of 86.40% with pretraining, significantly outperforming other models. Without pretraining, its accuracy dropped to 70.03%, indicating that pretraining is beneficial for this architecture.

2. **ResNet18** also showed a substantial improvement from 57.70% to 82.00% when pretrained, highlighting the effectiveness of transfer learning in deep networks.

3. **EfficientNet** and **MobileNet** exhibited lower pretraining benefits compared to VGG16 and ResNet18. EfficientNet's accuracy improved from 46.93% to 80.53%, while MobileNet's accuracy increased from 37.32% to 78.80%.

4. The models without pretraining performed significantly worse, emphasizing the importance of pretraining in achieving better performance, especially for deeper architectures.

In conclusion, pretraining consistently enhances model performance across all tested architectures, with VGG16 demonstrating the most significant advantage. The results suggest that leveraging pre-trained models is a crucial strategy in deep learning tasks.

5. Analysis and Insights

5.1. Model Complexity and Performance

The ResNet models consistently outperformed the VGG models and the custom CNN in terms of both validation loss and validation accuracy. This can be attributed to the residual connections in ResNet, which help mitigate the vanishing gradient problem and enable the training of deeper networks. Among the ResNet variants, ResNet34 struck the best balance between depth and performance, achieving the lowest validation loss and highest validation accuracy.

EfficientNet and MobileNet also demonstrated strong performance, particularly in terms of computational efficiency and generalization. EfficientNet's compound scaling method optimizes both depth and width, allowing it to achieve high accuracy with fewer parameters compared to traditional models. MobileNet, designed for mobile and edge devices, maintained competitive performance while being lightweight, making it suitable for resource-constrained environments. Both models exhibited lower validation losses and higher validation accuracies than the VGG models, indicating their robustness against overfitting.

In contrast, the VGG models, despite their depth, showed signs of overfitting, likely due to their lack of residual connections, which makes them more susceptible to the vanishing gradient problem as network depth increases. The custom CNN, being less deep and lacking sophisticated architectural features, performed the worst, underscoring the importance of network depth and architecture in achieving good performance.

5.2. Training Dynamics

The training accuracies of the ResNet models were consistently high, indicating effective learning of the training data. However, the gap between training and validation accuracy was smaller for ResNet34 and ResNet18 compared to ResNet50, suggesting better generalization. EfficientNet and MobileNet also exhibited effective training dynamics, with steady improvements in both training and validation accuracy, indicating their ability to learn robust features without significant overfitting.

In contrast, the VGG models showed a larger gap between training and validation accuracies, indicating overfitting. The custom CNN had the lowest training accuracy, reflecting its limited capacity to learn complex patterns effectively.

5.3. Generalization and Overfitting

The validation losses and accuracies provide insights into the generalization capabilities of the models. ResNet18, with its balanced depth and residual connections, demonstrated the best generalization, followed closely by EfficientNet and MobileNet, which maintained lower validation losses and higher validation accuracies. This indicates that both EfficientNet and MobileNet can generalize well to unseen data, making them suitable for practical applications.

The VGG models, despite their high training accuracies, showed lower validation accuracies, suggesting they struggled with generalization due to overfitting. The custom CNN, with the highest validation loss and lowest validation accuracy, struggled to generalize, further highlighting the importance of model architecture in achieving good performance.

In summary, the analysis underscores the significance of model complexity, training dynamics, and generalization capabilities in deep learning tasks. EfficientNet and MobileNet, alongside ResNet, demonstrate that advanced architectures can lead to improved performance and generalization, while traditional models like VGG may require additional strategies to mitigate overfitting.

6. Hyper-parameter Tuning

Hyper-parameter tuning is a crucial step in optimizing the performance of machine learning models, particularly

in deep learning. In this section, we focus on the VGG16 network with pretrained models, exploring various hyper-parameters that can significantly influence the model's performance. Effective tuning can lead to improved accuracy, reduced overfitting, and enhanced generalization capabilities.

6.1. Optimizers and Learning Rates

One of the primary hyper-parameters to tune is the choice of optimizer and its associated learning rate. Common optimizers include Stochastic Gradient Descent (SGD), Adam, and RMSprop. Each optimizer has its strengths and weaknesses. For instance, while SGD is known for its simplicity and effectiveness in many scenarios, it may require careful tuning of the learning rate. On the other hand, Adam adapts the learning rate based on the first and second moments of the gradients, often leading to faster convergence.

When tuning the learning rate, it is essential to consider a range of values. A learning rate that is too high can cause the model to converge poorly or even diverge, while a learning rate that is too low may result in excessively slow convergence. Techniques such as learning rate scheduling, where the learning rate is adjusted dynamically during training, can also be beneficial. For example, reducing the learning rate when the validation loss plateaus can help the model escape local minima and improve overall performance.

6.2. Regularization Methods

Regularization techniques are vital for preventing overfitting, especially in deep networks like VGG16. Common regularization methods include L1 and L2 regularization, dropout, and batch normalization. L2 regularization adds a penalty to the loss function based on the square of the weights, discouraging overly complex models. Dropout, on the other hand, randomly deactivates a fraction of neurons during training, promoting redundancy and robustness in the network.

Batch normalization can also play a significant role in stabilizing the training process. By normalizing the inputs to each layer, it helps maintain a consistent distribution of activations, which can lead to faster convergence and improved performance. When tuning regularization methods, it is essential to experiment with different rates and configurations to find the optimal balance that minimizes validation loss without sacrificing the model's ability to learn.

6.3. Residual Links

Incorporating residual links, as seen in architectures like ResNet, can enhance the learning capabilities of deep networks. While VGG16 does not inherently include residual connections, experimenting with skip connections can be an interesting avenue for hyper-parameter tuning. By allowing gradients to flow more easily through the network, residual links can mitigate the vanishing gradient problem and enable the training of deeper architectures.

When testing the inclusion of residual links, it is crucial to monitor the effects on both training and validation losses. If the model shows improved performance with residual connections, it may suggest that the architecture benefits from the enhanced flow of information, leading to better feature extraction and generalization.

7. Code Overview

The code for this project is implemented in Python using popular deep learning libraries such as sklearn and pytorch.

You can find the full implementation at the following GitHub repository: `https://github.com/yyaaoo33/deeplearning-ass2.git`.

8. Conclusion

This paper compared the performance of several popular neural network models on a standard image classification task. The results showed that ResNet models, particularly ResNet18, achieved the best performance in terms of validation loss and accuracy, indicating superior generalization capabilities. The VGG models, despite their depth, showed signs of overfitting, while the custom CNN struggled to learn and generalize effectively.

In addition to ResNet, EfficientNet and MobileNet were also evaluated. EfficientNet demonstrated a remarkable balance between accuracy and computational efficiency, achieving competitive results while using fewer parameters compared to traditional models. This efficiency makes it particularly suitable for deployment in resource-constrained environments. MobileNet, designed for mobile and edge devices, also exhibited strong performance, with a focus on lightweight architecture and speed, although it did not match the accuracy levels of ResNet and EfficientNet.

These findings highlight the importance of model architecture in achieving good performance on deep learning tasks. Residual connections, as used in ResNet, play a crucial role in enabling the training of deeper networks and improving generalization. EfficientNet's compound scaling method further emphasizes the need for a balanced approach to model size and performance. Future work could explore further refinements to these architectures, such as incorporating attention mechanisms or exploring different optimization techniques, to further enhance performance. Additionally, investigating the trade-offs between accuracy, efficiency, and model complexity could yield valuable insights for practical applications.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] J. Hosang, M. Omran, R. Benenson, and B. Schiele. Taking a deeper look at pedestrians. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4073–4082, 2015.

[3] R. F. Rachmadi, I. E. Purnama, M. H. Purnomo, and M. Hariadi. A systematic evaluation of shallow convolutional neural network on cifar dataset. *IAENG International Journal of Computer Science*, 46(2):365–376, 2019.

[4] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018(1):7068349, 2018.

[6] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9:611–629, 2018.