Deep Learning Image Classification with Custom CNN and Standard Architectures on CIFAR-100

Dhinakar R
Blekinge Institute of Technology - Karlskrona, Sweden - dhra24@student.bth.se

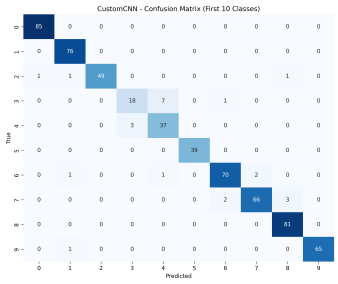

Fig. 1. Custom CNN's confusion matrix

***Abstract*—This paper implements and evaluates multiple deep learning architectures for image classification on the CIFAR-100 dataset. We design a custom convolutional neural network (CNN) with 6 convolutional layers organized in 3 blocks, featuring batch normalization, dropout, and global average pooling. The custom CNN is compared against four state-of-the-art architectures: ResNet-50, VGG-19, DenseNet-121, and EfficientNet-B0. Experimental results show that ResNet-50 achieves the highest accuracy (63.11%), followed by our custom CNN (55.60%), demonstrating that well-designed moderate-sized networks can be competitive with more complex architectures. We analyze training dynamics, model complexity, and filter visualizations to provide insights into the comparative performance of different architectural approaches.**

***Index Terms*—convolutional neural networks, image classification, CIFAR-100, deep learning, ResNet, VGG, DenseNet, EfficientNet**

I. Introduction

Image classification remains a fundamental challenge in computer vision and a testbed for deep learning architectures. While state-of-the-art models continue to evolve in complexity, understanding the effectiveness of different architectural designs provides valuable insights for developing efficient and performant neural networks.

This paper focuses on implementing and evaluating deep learning models for the CIFAR-100 dataset, which contains 100 classes of small (32×32) color images. We design a custom CNN architecture with a moderate number of layers and compare its performance against established models including ResNet-50, VGG-19, DenseNet-121, and EfficientNet-B0.

Our work examines:

- Design principles for effective CNN architectures
- The impact of architectural choices on classification performance
- Training dynamics across different model complexities
- Visual interpretation of learned filters and features

Through this comparison, we aim to understand the trade-offs between model complexity and performance, and to identify which architectural approaches are most suitable for the CIFAR-100 classification task.

II. Custom CNN Architecture

A. Design Philosophy

Our custom CNN architecture follows modern design principles while maintaining reasonable computational requirements. The network consists of three convolutional blocks followed by global average pooling and fully connected layers for classification. Key design considerations include:

- Progressive channel expansion (64→128→256) to learn increasingly complex features
- Dual convolutional layers in each block for feature hierarchies
- Batch normalization after each convolution for training stability
- Dropout layers with strategic rates (0.25 in convolutional blocks, 0.5 before classification)
- Global average pooling to reduce parameters and improve generalization

B. Network Structure

The detailed structure of our custom CNN is as follows:

- **Block 1:**
  - Conv2D: 3→64 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU
  - Conv2D: 64→64 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU
  - MaxPool2D: 2×2, stride=2
  - Dropout: 0.25
- **Block 2:**
  - Conv2D: 64→128 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU
  - Conv2D: 128→128 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU
  - MaxPool2D: 2×2, stride=2
  - Dropout: 0.25
- **Block 3:**
  - Conv2D: 128→256 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU
  - Conv2D: 256→256 channels, 3×3 kernel, padding=1
  - BatchNorm2D + ReLU

  - MaxPool2D: 2×2, stride=2
  - Dropout: 0.25
- **Classification Head:**
  - Global Average Pooling
  - Fully Connected: 256→512 units
  - BatchNorm1D + ReLU
  - Dropout: 0.5
  - Fully Connected: 512→100 units (one per class)

III. Implementation Details

A. Hyperparameters and Training Configuration

The hyperparameters for all models were selected based on empirical testing and common practices in deep learning research:

- **Optimizer:** Adam with learning rate 0.001 and weight decay 1e-4
- **Loss Function:** Cross-Entropy Loss
- **Batch Size:** 128
- **Learning Rate Scheduler:** ReduceLROnPlateau with factor=0.5, patience=5
- **Early Stopping:** 10 epochs for custom CNN, 7 epochs for standard models
- **Maximum Epochs:** 10
- **Random Seed:** 42 for reproducibility

B. Data Preprocessing and Augmentation

The CIFAR-100 dataset was processed using the following techniques:

- **Normalization:** Channel-wise using mean=(0.5071, 0.4867, 0.4408) and std=(0.2675, 0.2565, 0.2761)
- **Training Augmentation:**
  - Random crop: 32×32 with padding=4
  - Random horizontal flip
  - Color jitter: brightness and contrast variations (±20%)
- **Testing Transformation:** Normalization only

C. Standard Model Adaptations

The pre-trained models were adapted for CIFAR-100 as follows:

- **ResNet-50:** Adjusted first conv layer to 3×3 kernel, stride=1, padding=1; removed maxpool; changed FC layer to output 100 classes
- **VGG-19:** Modified final FC layer to output 100 classes
- **DenseNet-121:** Modified classifier to output 100 classes
- **EfficientNet-B0:** Modified classifier to output 100 classes

IV. Results and Analysis

A. Performance Metrics

Table I presents the comprehensive performance metrics for all models.

ResNet-50 achieved the highest performance across all metrics, followed by our custom CNN. Notably, our custom architecture outperformed both DenseNet-121 and EfficientNet-B0, despite having fewer parameters. VGG-19 showed poor performance, likely due to optimization difficulties given the training constraints.

TABLE I
Model Performance Comparison

Model	Acc.	Prec.	Rec.	F1
CustomCNN	0.556	0.564	0.556	0.552
ResNet-50	0.631	0.653	0.631	0.627
VGG-19	0.024	0.004	0.024	0.006
DenseNet-121	0.529	0.530	0.529	0.520
EfficientNet-B0	0.334	0.333	0.334	0.321

B. Training Dynamics

The custom CNN showed healthy learning curves with both training and validation metrics improving steadily throughout training. Interestingly, validation metrics consistently exceeded training metrics, indicating:

- Effective regularization through dropout and batch normalization
- The impact of data augmentation making the training set more challenging than the validation set
- No signs of overfitting, even after multiple epochs

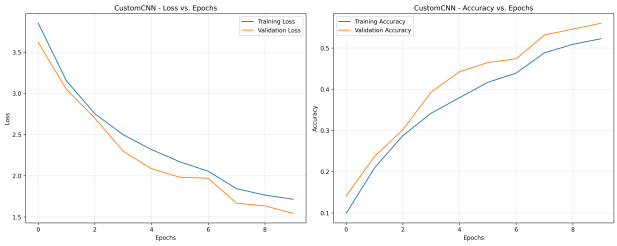

Fig. 2. Training and validation loss/accuracy for the custom CNN. The validation metrics consistently exceed training metrics, indicating effective regularization and the impact of data augmentation.

ResNet-50 showed similarly healthy learning dynamics but reached higher performance levels, likely due to its skip connection architecture that facilitates gradient flow in deeper networks.

C. Filter Visualization

Analyzing the first-layer convolutional filters before and after training provides insight into how the network learns to process visual information. Fig. 3 shows these filters for our custom CNN.

Fig. 3. First layer convolutional filters before training (top) and after training (bottom). The filters evolved from random initializations to structured patterns sensitive to edges, textures, and color transitions.

The filters evolved from random initializations to structured patterns that detect:

- Edge features at various orientations
- Color transition boundaries
- Textural patterns relevant for classification

This progression demonstrates the network's ability to learn fundamental visual features that serve as building blocks for higher-level representations in deeper layers.

D. Architectural Comparison

The performance ranking of the architectures (ResNet-50 ¿ CustomCNN ¿ DenseNet-121 ¿ EfficientNet-B0 ¿ VGG-19) offers several insights:

- **Skip connections matter:** ResNet-50's superior performance highlights the importance of residual connections in facilitating gradient flow and enabling effective training of deeper networks.
- **Depth vs. design:** Our custom CNN outperformed more complex architectures through careful design choices, suggesting that architectural efficiency can sometimes compensate for reduced depth.
- **Training challenges:** VGG-19's poor performance underscores the difficulties in training very deep networks without skip connections, especially with limited training epochs.
- **Parameter efficiency:** EfficientNet-B0, despite being designed for efficiency, underperformed on this specific task, possibly due to its mobile-oriented design choices that may not be optimal for the CIFAR-100 dataset.

V. Conclusion

This paper presented a comparative analysis of deep learning architectures for CIFAR-100 image classification, highlighting the effectiveness of both established models and a custom-designed CNN. Our key findings include:

- ResNet-50 achieved the highest accuracy (63.11%), demonstrating the effectiveness of skip connections for this classification task.
- Our custom CNN (55.60% accuracy) outperformed more complex architectures like DenseNet-121 and EfficientNet-B0, showcasing the importance of thoughtful design over sheer model size.
- Effective regularization through dropout and batch normalization, combined with data augmentation, prevented overfitting across all successfully trained models.
- Visualization of convolutional filters revealed the evolution from random initializations to structured feature detectors capturing edges, textures, and color patterns.

The performance gap between our custom CNN and ResNet-50 (7.51%) suggests that while architectural innovations like skip connections provide significant advantages, well-designed moderate-sized networks can still achieve competitive results.

Future work could explore hybrid architectures that incorporate skip connections into smaller networks, potentially achieving a better balance between model complexity and performance for tasks like CIFAR-100 classification.

Acknowledgment

The author would like to thank the course instructors for their guidance and feedback during this implementation.

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, 2016, pp. 770-778.

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

[3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700-4708.

[4] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in International Conference on Machine Learning, 2019, pp. 6105-6114.

[5] A. Krizhevsky, "Learning multiple layers of features from tiny images," Technical Report, University of Toronto, 2009.