Title: Comparison of 3 different architecture ResNet, VGG16 and CNN model on CIFAR-100 dataset.

GitHub code link:

https://github.com/dhruvjain1999/Cifar_100_dataset_model_comparision.git

1. Abstract

The developed code encompasses the implementation and evaluation of a Convolutional Neural Network (CNN) for image classification on the CIFAR-100 dataset. Additionally, the robustness of the model is assessed through the introduction of **noise**. The study compares the performance of the **CNN model** against well-established architectures, **VGG16** and **ResNet50**. Experimental results showcase the accuracy and robustness of each model, providing insights into their effectiveness for image classification tasks and t-SNE plot.

Key words: VGG16, ResNet, CNN model

2. Introduction

In the rapidly evolving field of computer vision, image classification is a fundamental task with applications ranging from object recognition to autonomous vehicles. This project focuses on implementing a CNN for image classification using the CIFAR-100 dataset, a challenging dataset with 100 classes. To provide a comprehensive comparison, we evaluate CNN against two prominent architectures, VGG16 and ResNet50. Furthermore, we explore the robustness of each model by introducing noise to the test set.

The **CIFAR-100** dataset (Canadian Institute for Advanced Research, 100 classes) is a subset of the Tiny Images dataset and consists of 60000 32x32 color images. The 100 classes in the CIFAR-100 are grouped into 20 super classes. There are 600 images per class. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs). There are 500 training images and 100 testing images per class.

ResNet, short for Residual Network, is a groundbreaking deep learning architecture designed to address the challenge of training very deep neural networks. Introduced by Kaiming He et al. in the paper "Deep Residual Learning for Image Recognition" in 2015, ResNet introduced the concept of residual learning, which involves using shortcut connections to skip one or more layers during training.

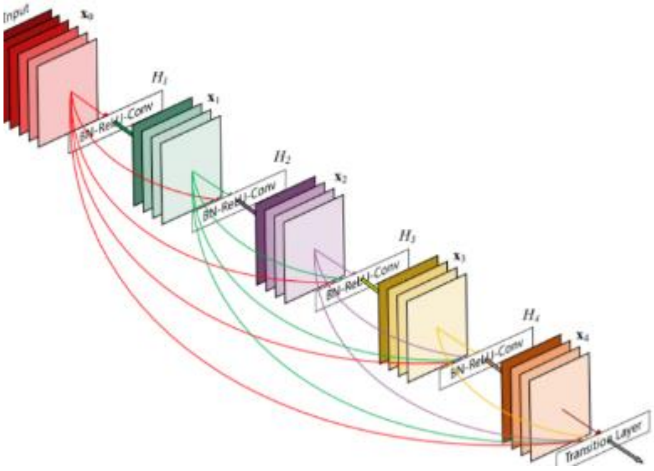


Figure 1: ResNet Architecture

Each of the layers follow the same pattern. They perform 3x3 convolution with a fixed feature map dimension (F) [64, 128, 256, 512] respectively, bypassing the input every 2 convolutions. Furthermore, the width (W) and height (H) dimensions remain constant during the entire layer.

VGG16 A convolutional neural network is also known as a ConvNet, which is a kind of artificial neural network. A convolutional neural network has an input layer, an output layer, and various hidden layers. VGG16 is a type of Convolutional Neural Network that is one of the best computer vision models to date. The creators of this model evaluated the networks and increased the depth using an architecture with very small (3 × 3) convolution filters, which showed a significant improvement on the prior-art configurations. They pushed the depth to 16–19 weight layers making it approx. — 138 trainable parameters.

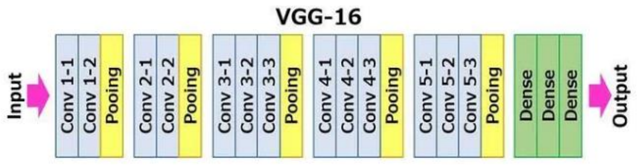


Figure 2 VGG16 architecture

The 16 in VGG16 refers to 16 layers that have weights. In VGG16 there are thirteen convolutional layers, five Max Pooling layers, and three Dense layers which sum up to 21 layers, but it has only sixteen weight layers i.e., learnable parameters layer.

3. Related work

Prior research has extensively explored deep learning models for image classification, with architectures like VGG16 and ResNet50 demonstrating remarkable performance on various datasets. Studies have investigated the robustness of neural networks, emphasizing the importance of models that can generalize well to noisy or distorted inputs. My work builds upon these foundations, contributing to the understanding of CNN performance and robustness in the context of the CIFAR-100 dataset.

4. Approach

Here approach involves the implementation of a CNN architecture tailored for image classification. The model is trained on the CIFAR-100 dataset, with training and validation results monitored for performance trends. To assess robustness, a novel image is introduced with controlled noise, and model evaluations are conducted. Technical correctness is maintained through adherence to best practices in deep learning, utilizing appropriate preprocessing techniques and model architectures.

4.1. CNN architecture

Convolutional Layer 1 (conv2d_27):

Input shape: (None, 32, 32, 3) (batch size, height, width, channels)

Output shape: (None, 30, 30, 32) (batch size, height, width, number of filters)

Number of parameters: 896

Explanation: The convolutional layer has 32 filters of size 3x3, resulting in an output volume of 30x30 with 32 channels.

Max Pooling Layer 1 (max_pooling2d_27):

Input shape: (None, 30, 30, 32)

Output shape: (None, 15, 15, 32)

Explanation: Max pooling with a pool size of 2x2 reduces the spatial dimensions by half.

Convolutional Layer 2 (conv2d_28):

Input shape: (None, 15, 15, 32)

Output shape: (None, 13, 13, 64)

Number of parameters: 18,496

And so on for other layers can be found in table1. Total no of parameters is 250276.

After running the model and obtaining the evaluation metrics, the primary objective is to assess whether the model is overfitting or underfitting. To achieve this, I introduced a **noise term** to the testing dataset. By incorporating noise into the model, I evaluate its performance. The level of noise provides valuable insights into the model's behavior; a high performance under noise indicates effective generalization. Conversely, if the performance degrades when noise is introduced, it suggests the need to adjust hyperparameters for improved accuracy.

Furthermore, I am incorporating a **novel image** to assess the model's performance when presented with input that was not part of the training or testing datasets. This step helps evaluate the model's ability to generalize to unseen data and provides insights into its robustness and real-world applicability.

5. Experimental results

The "Test Accuracy %" values represent the accuracy achieved by three different models—VGG16, ResNet, and a generic CNN—on a test dataset. Here's the interpretation.

The VGG16 model achieved an accuracy of approximately 26.29% on the test dataset. This percentage indicates the proportion of correctly predicted instances out of the total test samples.

The ResNet model achieved a higher accuracy compared to VGG16, reaching approximately 34.48%. It suggests that the ResNet model performed better in terms of correct predictions on the same test dataset.

The generic CNN model achieved the highest accuracy among the three models, with a test accuracy of approximately 42.06%. This indicates a relatively better performance in terms of correct predictions on the test dataset compared to VGG16 and ResNet.

Table 1 Test Accuracy

	Test Accuracy %
VGG16 Model	26.29
ResNet Model	34.48
CNN Model	42.06

In summary, based on the test accuracy percentages, the generic CNN model appears to be the most accurate among the three models, followed by the ResNet model, and then the VGG16 model. However, it's essential to consider other factors such as model complexity, computational resources, and the specific requirements of the task before making a final decision on model selection.

Table 2 Comparison of all 3 models

	Precision %	Recall %	F1-score %
ResNet Model	37	34	33
VGG-16 Model	28	26	24
CNN Model	44	42	41

5.1.Noise checking

The ResNet model demonstrated a reasonable level of accuracy, approximately 34.6%, when tested on data that includes noise. This suggests that one can still add more parameters to tune the model as it underfits.

The VGG-16 model exhibited a lower accuracy of around 26.5% on the noisy test dataset. This indicates that the model's performance might be more affected by the presence of noise compared to the ResNet model.

The generic CNN model achieved a high accuracy of approximately 41.69% on the noisy test dataset. This suggests good performance in the presence of noise, making it a relatively robust model under such conditions.

The ranking in terms of performance on noisy data is as follows: CNN Model (41.69%) > ResNet Model (34.6%) > VGG-16 Model (26.5%). The CNN model appears to be the most robust in handling noise, while the ResNet model performs reasonably well, and the VGG-16 model shows comparatively lower performance in the presence of noise.

Table 3 Noise Test Accuracy

Model	**Noise Test Accuracy %**
ResNet Model	34.6
VGG-16 Model	26.5
CNN Model	41.69

5.2.Novel image

Novel image typically refers to an image that the model has not encountered during its training phase. It's an image that the model has not seen before and is not part of the dataset used for training or testing. When discussing the "Top Predicted Classes" for each model, it provides insights into the model's predictions and confidence scores for different classes. Let's break down the provided information:

Table 4 Novel Image prediction

Model	**Top Predicted Classes for Resnet Model:**
CNN Model	**1. apple: 82.93%** 2. pear: 6.26% 3. wardrobe: 2.96% 4. plate: 1.97% 5. bowl: 1.47%
Vgg16 Model	**1. clock: 100.00%** 2. telephone: 0.00% 3. can: 0.00% 4. couch: 0.00% 5. crocodile: 0.00%
Resnet Model	**1. poppy: 1.46%** 2. rose: 1.31% 3. tulip: 1.30% 4. sweet pepper: 1.15% 5. aquarium fish: 1.15%

The CNN model expresses strong confidence in predicting the class "apple" for the novel image. The ResNet and VGG16 model appears uncertain about the class of the novel image, providing low confidence scores for multiple classes.

6. Conclusion

In conclusion, our study sheds light on the efficacy of a custom CNN model for image classification on the CIFAR-100 dataset. The comparison with VGG16 and ResNet highlights trade-offs in model complexity and accuracy.

The CNN model exhibited the highest test accuracy, followed by the ResNet model and then the VGG16 model. The CNN model also demonstrated good performance in the presence of noise, making it relatively robust. The CNN model showed the highest accuracy under noisy conditions, suggesting resilience to noise. The ResNet model demonstrated reasonable robustness, while the VGG16 model exhibited a lower accuracy in the presence of noise. The CNN model provided specific and confident predictions for a novel image, particularly for the class "apple." The VGG16 model showed a highly confident prediction for the class "clock". The ResNet model exhibited uncertainty with lower confidence scores for multiple classes for the novel image.

t-SNE plot

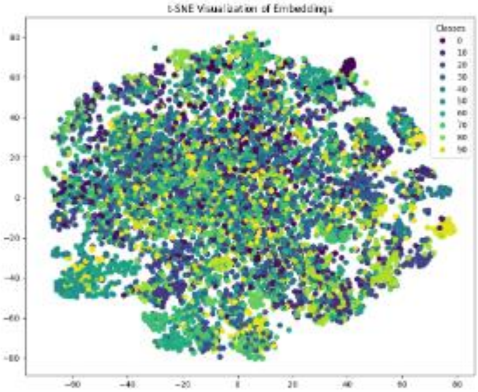


Figure 3 ResNet t-SNE plot

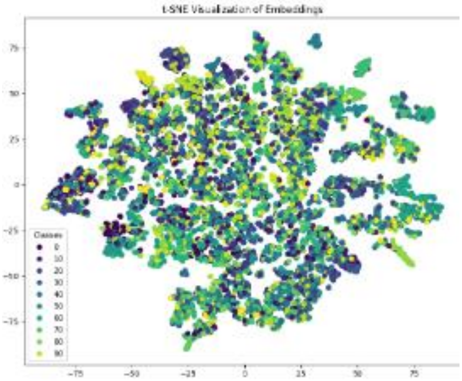


Figure 4 VGG16 t-SNE plot

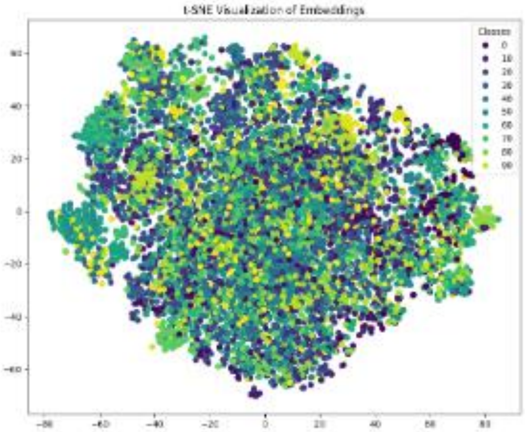


Figure 5 CNN t-SNE plot

7. Future Work

An exploration of ensemble methods could enhance overall model performance by combining predictions from multiple models, providing a robust and accurate solution. Further investigation into transfer learning, especially fine-tuning pre-trained models on a domain-specific dataset, may leverage existing knowledge to improve task-specific performance. Iterative development, user feedback incorporation, and dataset expansion are essential for refining models to meet evolving requirements and ensuring broader applicability.

8. References

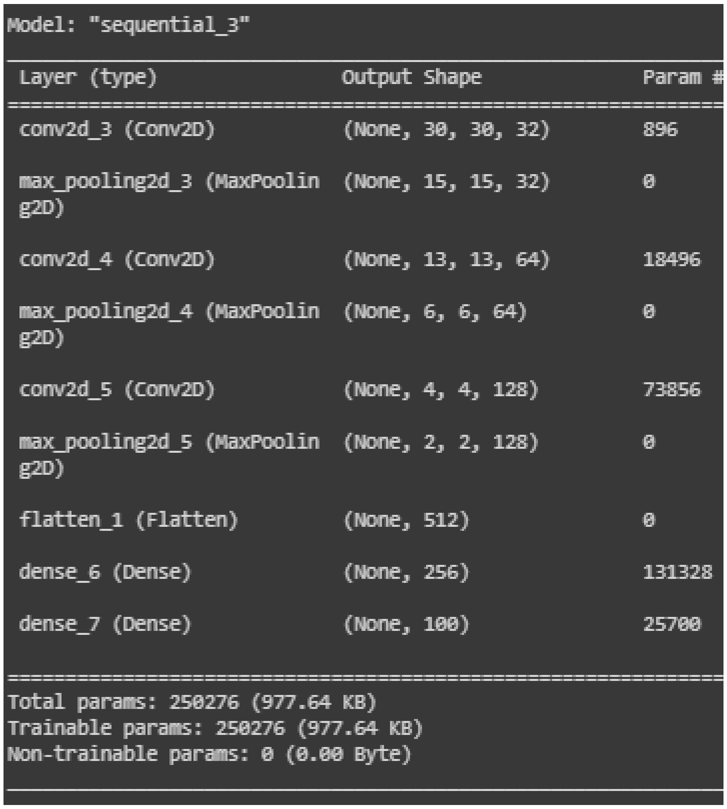
[1] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition (Version 6). arXiv. https://doi.org/10.48550/ARXIV.1409.1556

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166, 1994.

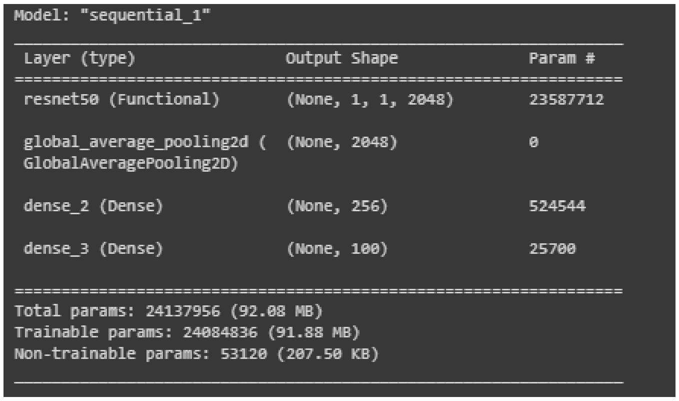
[3] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1512.03385

9. Appendix

Table 5: CNN architecture

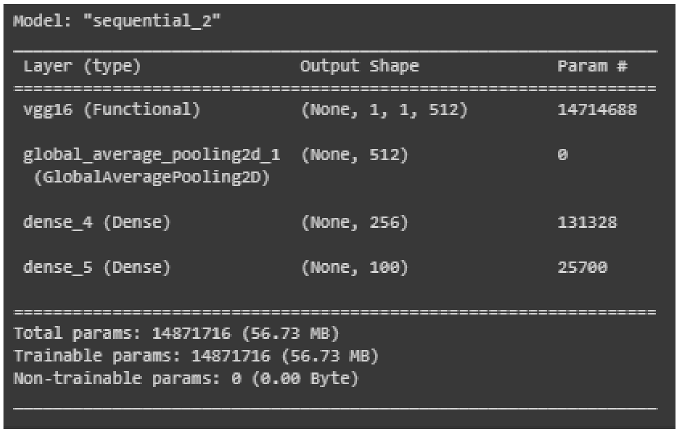


```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 30, 30, 32)        896

 max_pooling2d_3 (MaxPoolin  (None, 15, 15, 32)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_4 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 conv2d_5 (Conv2D)           (None, 4, 4, 128)         73856

 max_pooling2d_5 (MaxPoolin  (None, 2, 2, 128)         0
 g2D)

 flatten_1 (Flatten)         (None, 512)               0

 dense_6 (Dense)             (None, 256)               131328

 dense_7 (Dense)             (None, 100)               25700

=================================================================
Total params: 250276 (977.64 KB)
Trainable params: 250276 (977.64 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```



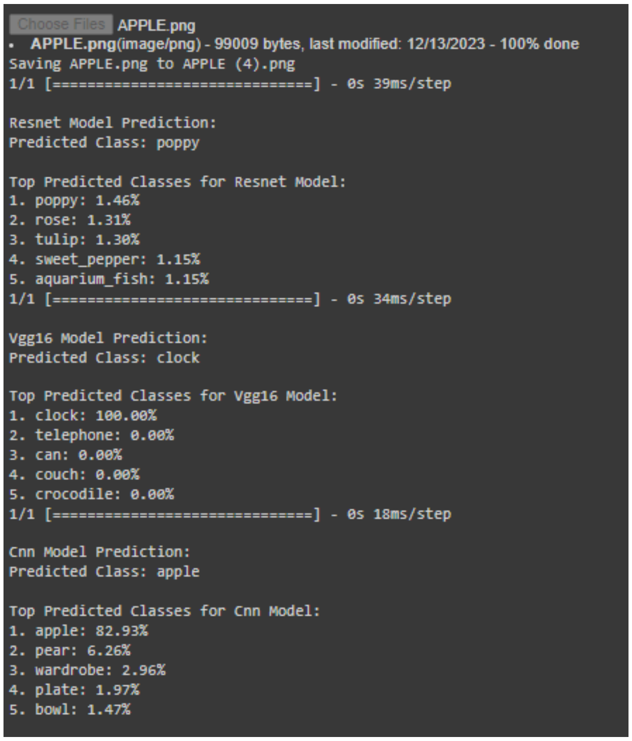
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 1, 1, 2048)        23587712

 global_average_pooling2d (  (None, 2048)              0
 GlobalAveragePooling2D)

 dense_2 (Dense)             (None, 256)               524544

 dense_3 (Dense)             (None, 100)               25700

=================================================================
Total params: 24137956 (92.08 MB)
Trainable params: 24084836 (91.88 MB)
Non-trainable params: 53120 (207.50 KB)
_________________________________________________________________
```

Figure 6 resnet50 architecture



```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 1, 1, 512)         14714688

 global_average_pooling2d_1  (None, 512)               0
  (GlobalAveragePooling2D)

 dense_4 (Dense)             (None, 256)               131328

 dense_5 (Dense)             (None, 100)               25700

=================================================================
Total params: 14871716 (56.73 MB)
Trainable params: 14871716 (56.73 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 7 VGG16 architecture



```
APPLE.png
• APPLE.png(image/png) - 99009 bytes, last modified: 12/13/2023 - 100% done
Saving APPLE.png to APPLE (4).png
1/1 [==============================] - 0s 39ms/step

Resnet Model Prediction:
Predicted Class: poppy

Top Predicted Classes for Resnet Model:
1. poppy: 1.46%
2. rose: 1.31%
3. tulip: 1.30%
4. sweet_pepper: 1.15%
5. aquarium_fish: 1.15%
1/1 [==============================] - 0s 34ms/step

Vgg16 Model Prediction:
Predicted Class: clock

Top Predicted Classes for Vgg16 Model:
1. clock: 100.00%
2. telephone: 0.00%
3. can: 0.00%
4. couch: 0.00%
5. crocodile: 0.00%
1/1 [==============================] - 0s 18ms/step

Cnn Model Prediction:
Predicted Class: apple

Top Predicted Classes for Cnn Model:
1. apple: 82.93%
2. pear: 6.26%
3. wardrobe: 2.96%
4. plate: 1.97%
5. bowl: 1.47%
```

Figure 8 Novel image prediction