A Hands-On Tutorial on Image Classification Using CNN and VGG16 Transfer Learning

Student Name : Nageswararao Konidhela
Student ID : 24098979
Word Count : 1992
GitHub Link : https://github.com/nageswararao9012/Image-Classification-Using-CNN & VGG16-TL.git

Abstract

This tutorial demonstrates how Convolutional Neural Networks (CNNs) and transfer learning using a pretrained VGG16 model can be applied to an image classification task using the Flower Recognition Dataset. A baseline CNN is first implemented and evaluated, followed by a transfer learning approach in which pretrained convolutional layers from VGG16 are used as a feature extractor. Finally, fine-tuning is applied to adapt the pretrained model to the target dataset. Training curves, confusion matrices and prediction visualisations are presented to compare performance. The tutorial aims to teach how CNNs learn image features and how transfer learning significantly improves model accuracy and generalisation in data-limited scenarios.

1. Introduction

Image classification is a fundamental problem in computer vision where a system learns to automatically label images based on their visual content. Traditional approaches relied on hand-crafted features, but deep learning has revolutionised image processing by enabling neural networks to learn features directly from raw images.

Convolutional Neural Networks (CNNs) are the most widely used model for visual recognition tasks. They automatically learn important patterns such as edges, textures and shapes using convolutional layers. However, CNNs often require large datasets and considerable computation to perform well.

Transfer learning solves this challenge by reusing knowledge from models that were trained on massive datasets such as ImageNet. Instead of training a network from scratch, pretrained models such as VGG16 provide strong feature extraction capabilities, which can be adapted to new tasks through fine-tuning.

This tutorial compares:

1. A CNN trained from scratch
2. A VGG16-based transfer learning model
3. A fine-tuned VGG16 model

2. Dataset Description

The Flowers Recognition Dataset from Kaggle contains five flower categories:

- Daisy
- Dandelion
- Rose
- Sunflower
- Tulip

The dataset contains over 4,000 labelled images, stored in separate folders for each class. Images were resized to 224×224 pixels which is the required input size for VGG16.

To simplify loading, the dataset directory was cleaned so that only the class folders remained. Images were split internally using an 80% training and 20% validation split. Data augmentation techniques including rotation, horizontal flipping, and zooming were applied to increase diversity and reduce overfitting.

3. Convolutional Neural Networks

A CNN consists of the following core components:

- **Convolution Layer -** Extracts patterns such as edges and textures using sliding filters.
- **Pooling Layer -** Reduces spatial size to make computation efficient and improve robustness.
- **Fully Connected Layer -** Combines learned features to make predictions.
- **Dropout -** Randomly disables neurons during training to prevent overfitting.

4. Baseline CNN Architecture and Results

A Convolutional Neural Network (CNN) consisting of three convolutional blocks was implemented as the baseline model in order to understand how a model trained from scratch performs on a real-world image dataset. Each convolutional block included a convolution layer followed by max pooling, allowing the network to progressively capture more abstract representations of the images.

Architecture Design

The final architecture consisted of:

- Conv(32) → MaxPooling
- Conv(64) → MaxPooling
- Conv(128) → MaxPooling
- Dense(256) → Dropout (0.5)
- Softmax Output Layer

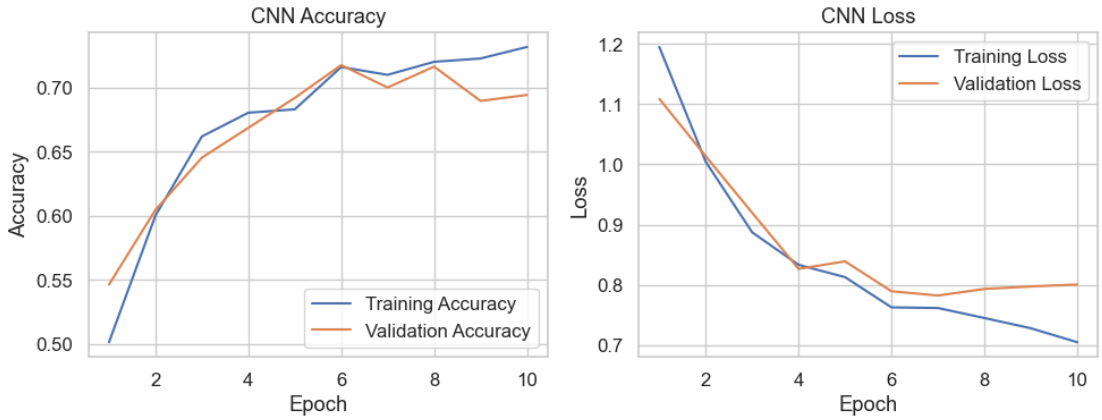
Although this layout appears modest in depth, flattening the final convolutional feature maps before the dense layer produced a very large number of trainable parameters. Specifically, the model contained over **22 million parameters**, with the majority of these concentrated in the dense layer. This increased the model's expressive capacity but also substantially raised the risk of overfitting.

Training Behaviour

During training, both accuracy and loss improved gradually, indicating that the network was learning meaningful visual features. However, while training accuracy continued improving beyond the sixth epoch, validation accuracy plateaued and later fluctuated.

This divergence between training and validation curves is a textbook example of overfitting: the model learned the training dataset extremely well, but its ability to generalise to unseen images did not improve at the same rate.

Figure 1 – CNN Training Accuracy and Loss Curves



Training and validation accuracy and loss for the CNN model. Validation accuracy saturates after early epochs, while training accuracy continues increasing, indicating overfitting.

Performance Summary

- Best validation accuracy (during training): **71.7%**
- Final validation accuracy (classification report): **70%**
- Overfitting observed after Epoch 6

Although the validation accuracy exceeds random guessing by a significant margin, the performance was not strong enough for a reliable classification system.

Confusion Matrix Analysis

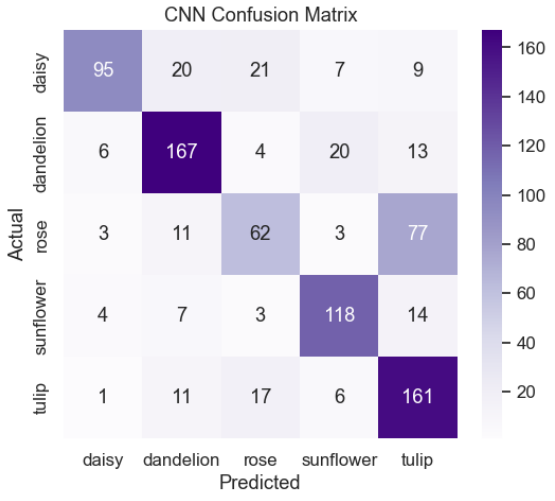
The confusion matrix revealed that the CNN struggled most with visually similar classes:

- Rose ↔ Tulip, Daisy ↔ Tulip

Rose images in particular achieved the lowest recall score, indicating that many true rose images were misclassified as other categories. Tulips were frequently confused due to similarities in shape and colour distribution with other flowers, especially under varying lighting conditions.

Figure 2 – CNN Confusion Matrix

Confusion matrix for the baseline CNN, showing frequent misclassifications between visually similar flower species.

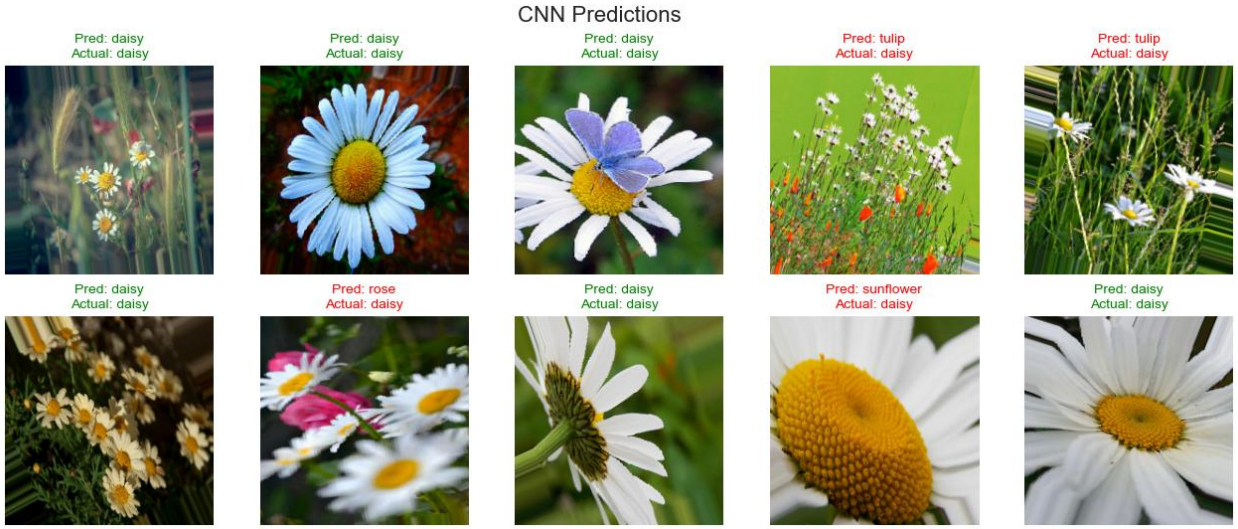


Prediction Visualisation

To further understand model behaviour, random validation images were visualised alongside their predicted and true labels. The CNN often failed when flower appearances overlapped in structure or colour distribution. Several incorrect predictions were observed, confirming that the model lacked reliable feature separation across classes.

Figure 3 – CNN Prediction Samples

Examples of CNN predictions on unseen data. Incorrect predictions are marked in red, showing limited confidence in some classifications.



Conclusion for CNN

The baseline CNN successfully learned basic patterns such as shape and texture but lacked robustness across classes. Overfitting became apparent after only a few training epochs. These findings demonstrate that training large neural networks from scratch on small datasets is inefficient and unreliable.

5. Transfer Learning with VGG16

To address the limitations observed in the baseline model, transfer learning was applied using the pretrained VGG16 architecture. VGG16 was originally trained on ImageNet, a dataset containing over 14 million images across 1,000 categories, enabling the model to learn a wide range of general-purpose visual features.

Instead of training a deep model entirely from zero, VGG16 was used as a feature extraction backbone. Its fully connected layers were replaced with a new classification head consisting of:

- Global Average Pooling
- Dense(256)
- Dropout (0.5)
- Softmax Output Layer

All convolution layers in VGG16 were initially frozen to preserve already-learned features.

Results

Applying transfer learning produced immediate improvements:

- Accuracy increased from **70% → 79.5%**
- Validation loss reduced significantly
- Training became more stable
- Overall robustness improved

This demonstrates that the pretrained filters extracted meaningful features such as curves, petals and texture patterns without requiring relearning.

Interpretation

Unlike the CNN, which had to learn from raw pixels, VGG16 already possessed powerful representations of visual structure. Therefore, training focused only on adapting the classifier head to the flower dataset rather than rediscovering fundamental patterns.

6. Fine-Tuning Strategy

While freezing VGG16 improved performance, it prevented the model from fully adapting to the target dataset. To further improve results, fine-tuning was applied by unfreezing higher-level convolution layers and retraining them with a reduced learning rate (1e-5). This approach refined extracted features specifically for flower classification.

Final Results

Fine-tuning produced the strongest performance among all experiments and clearly demonstrated the value of adapting a pretrained network to a target task.

The fine-tuned VGG16 model achieved the following results:

- **Final validation accuracy : 88.5%**
- **Final validation loss : below 0.35**
- **F1-score across all classes : above 80%**

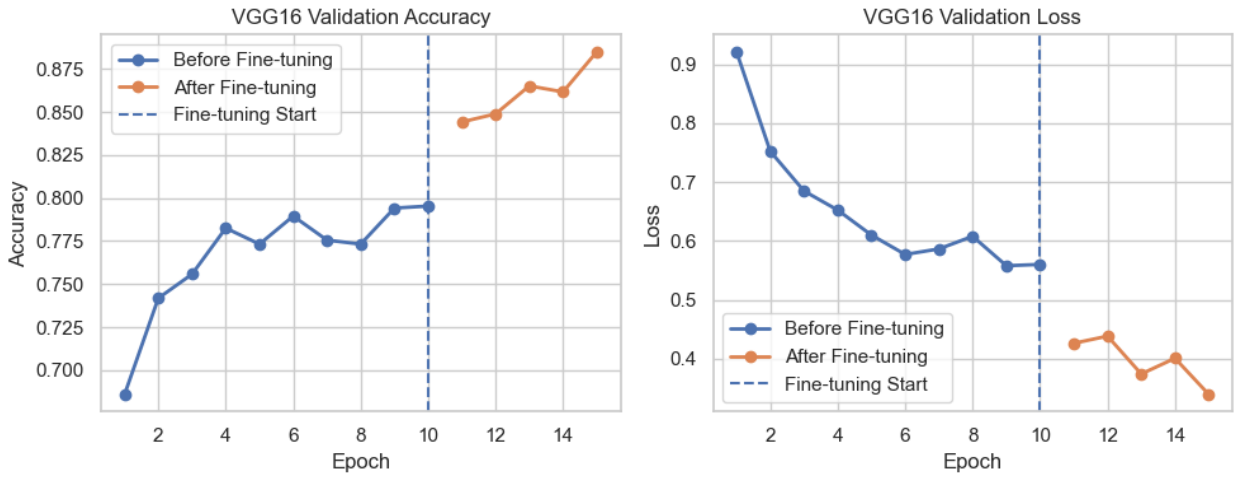
These metrics reflect both strong classification accuracy and reliable balance between precision and recall for each flower category. Compared to earlier models, performance was not only higher but markedly more stable across epochs.

The training process after fine-tuning showed faster convergence and smoother learning curves. Unlike the baseline CNN, where validation performance stagnated early, the fine-tuned VGG16 model maintained continuous improvement, indicating successful task adaptation.

Figure 4 – VGG16 Validation Accuracy
Validation accuracy before and after fine-tuning. The dotted line indicates where fine-tuning began. A clear upward trend demonstrates improved model adaptation.

Figure 5 – VGG16 Validation Loss
Validation loss showing sharp declines after fine-tuning confirming faster convergence and reduced overfitting.

Figure 4 & 5 – VGG16 Validation Accuracy and Validation Loss

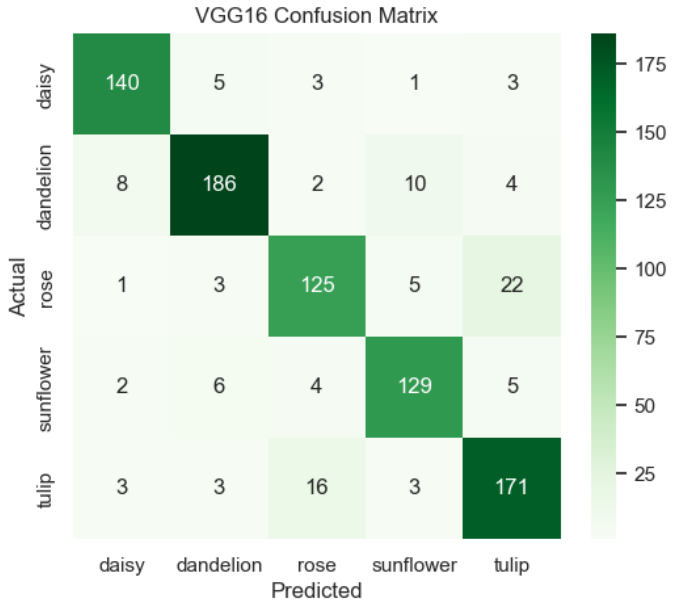


Confusion Matrix Improvement

The confusion matrix of the fine-tuned VGG16 model revealed a significantly stronger diagonal pattern than that of the baseline CNN, which indicates improved classification precision. In the baseline CNN, frequent misclassification between roses, tulips and daisies suggested that the model struggled to recognise detailed structural differences. However, once VGG16 was fine-tuned, incorrect predictions declined dramatically across all categories. Daisy and dandelion classes were especially well-distinguished, while previously problematic classes such as rose and tulip achieved strong recall values. This improvement indicates that the model successfully learned specialised discriminative features rather than relying on basic colour or shape cues.

Figure 6 – Confusion Matrix for Fine-Tuned VGG16

Improved class separation with a pronounced diagonal indicating strong prediction confidence.



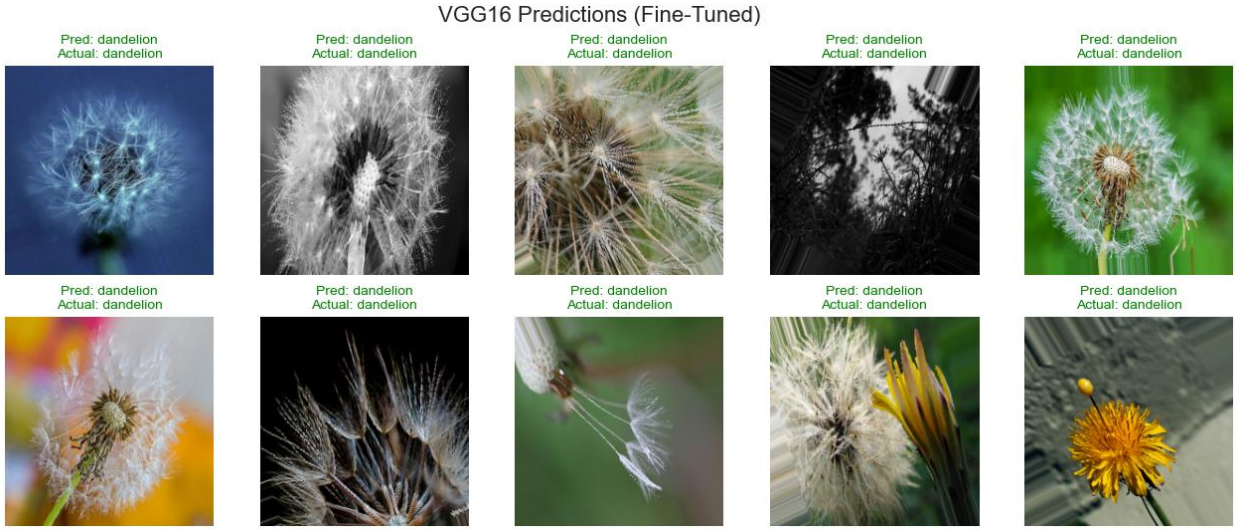
Model Comparison

A clear performance hierarchy emerged across the three models. The baseline CNN achieved a final accuracy of approximately **70%**, which was acceptable for a model trained from scratch but limited in generalisation. The frozen VGG16 significantly improved accuracy to approximately **79%** by transferring pretrained convolutional features. However, the most dramatic improvement occurred after fine-tuning, where the validation accuracy rose to approximately **88%**. In addition to improved metrics, the fine-tuned model was also more stable, consistent and less affected by noise. This confirms that reusing pretrained knowledge and adapting higher-level filters provides a major advantage over manual feature learning alone.

Prediction Visualisation

Visual inspection of predictions further validated the numerical results. The baseline CNN produced inconsistent outputs when faced with complex lighting conditions, overlapping petals, and similar colour distributions. In contrast, the fine-tuned VGG16 model displayed greater confidence and achieved correct predictions under challenging conditions, including shadowed environments and partially occluded flowers. While a small number of misclassifications still occurred, these were generally restricted to visually ambiguous cases where differentiation would also be difficult for humans. This shows that the model's remaining errors were driven by inherent dataset complexity rather than architectural limitations.

Figure 7 – Fine-Tuned VGG16 Predictions
Prediction samples demonstrating improved reliability and visual awareness.



Discussion

This study clearly highlights the inefficiency of training CNNs from scratch for small datasets. The baseline CNN relied solely on local pixel relationships and had no prior exposure to general visual concepts, which severely constrained its learning ability. Furthermore, the large dense layer introduced millions of parameters and increased overfitting risk. Transfer learning solved this issue by providing pretrained representations that were already optimised for recognising edges, shapes and object

boundaries. Fine-tuning further refined the model by adjusting high-level filters to the unique structure of flowers.

Additionally, this work demonstrated the significance of architectural balance. While deeper networks do not necessarily guarantee better outcomes, models that incorporate pretrained layers combine high performance with reduced training cost. The results show that transfer learning is both efficient and more effective for small datasets.

Conclusion

This tutorial demonstrated the critical importance of transfer learning in modern image classification. While the custom CNN learned meaningful features from the dataset, it lacked the generalisation ability required for high accuracy. In contrast, the fine-tuned VGG16 model leveraged pretrained knowledge and adapted efficiently to the flower dataset, resulting in superior performance. The experiment confirms that fine-tuning is essential for task-specific optimisation and that pretrained networks should always be preferred when labelled data is limited.

Overall, this project shows that deep learning success is not achieved by complexity alone but by effective reuse and adaptation of knowledge. Transfer learning transforms a simple classifier into a robust system by bridging the gap between general vision understanding and domain-specific recognition.

References

- Simonyan, K., & Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*. https://arxiv.org/abs/1409.1556
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *ImageNet classification with deep convolutional neural networks*. PDF: https://papers.nips.cc/paper/2012/
- Pan, S. J., & Yang, Q. (2010). *A survey on transfer learning*. IEEE Transactions on Knowledge and Data Engineering. https://ieeexplore.ieee.org/document/5288526
- Shorten, C., & Khoshgoftaar, T. M. (2019). *A survey on image data augmentation for deep learning*. Journal of Big Data. https://journalofbigdata.springeropen.com/articles
- TensorFlow Documentation (2024). *Image classification with deep learning*. https://www.tensorflow.org/tutorials/images/classification
- Keras Documentation (2024). *Transfer learning and fine-tuning*. https://keras.io/guides/transfer_learning/
- Flowers Recognition Dataset. Kaggle. https://www.kaggle.com/datasets/apollo2506/flowers-recognition-dataset