Convolutional Neural Networks: Effect of Filters and Depth

Author: Subhash Chandrupatla

Student id: 24072089

GIT Repository: https://github.com/Subhashchandrupatla/CNN-Effect-of-Filters-and-Depth

1.Introduction

Convolutional Neural Networks (CNNs) are among the most powerful and widely used deep-learning architectures for analysing spatial and visual data. They have transformed fields such as computer vision, medical imaging, facial recognition, autonomous vehicles, and even areas like natural language processing when data is arranged in spatial forms. Their strength comes from the ability to automatically learn patterns from raw data without requiring hand-crafted features.

A CNN's performance heavily depends on architectural choices, especially the **number of filters** used in each convolutional layer and the **depth of the network**, i.e., the total number of layers. Filters control what features the model can learn, while depth determines the complexity of hierarchical representation. Both factors significantly influence the accuracy, generalisation ability, computational cost, and effectiveness of a CNN.

This report provides a detailed and expanded analysis of how filters and depth affect the behaviour of CNNs. It includes discussions on feature extraction, computational load, vanishing gradients, overfitting, and the practices used in popular architectures. The aim is to provide an academically sound, comprehensive understanding suitable for coursework or research writing

2. Filters in CNNs

Filters (or kernels) are small matrices that slide across the input image to detect specific patterns. Each filter detects a single type of pattern. For example:

- A horizontal-edge filter detects horizontal lines.
- A diagonal-edge filter detects diagonal patterns.
- A texture filter may detect repeated patterns like fabrics, bricks, or waves.

When a filter traverses an image, it performs element-wise multiplication and summation (convolution), creating a **feature map**. Every filter produces a separate feature map, meaning the number of filters determines the richness of the extracted information.

2.1 The Role of Filter Count

The number of filters in a layer determines how many features the network can detect at that stage of processing. Increasing the number of filters allows the CNN to:

1. **Capture more diverse patterns**
   with more filters, the model learns a richer variety of edges, textures, and shapes.
2. **Increase representational power**
   Different filters specialise in different orientations, colours, and spatial patterns.
3. **Improve model accuracy**
   with more unique feature detectors, the model becomes better at distinguishing between similar classes.

However, increasing the number of filters also increases:

- The number of parameters
- Memory usage
- Training computation

So, filter count must be balanced.

2.2 Filter Size and Its Effect

Common filter sizes include **3×3**, **5×5**, and **7×7**. Smaller filters:

- Capture fine details
- Reduce parameters
- Allow stacking multiple layers (with ReLU) that together simulate larger receptive fields

Larger filters extract broader patterns but are computationally expensive. Modern architectures like VGG and ResNet heavily favour 3×3 filters because they provide efficiency and flexibility.

2.3 Too Few vs Too Many Filters

- **Too few filters**
  the model fails to capture necessary patterns → under fitting.
- **Too many filters**
  the model memorises the dataset → overfitting.
  Training becomes slow, and the model requires more memory.

A balanced design is essential

3. Depth of CNNs

Depth refers to the number of layers in the network. Each layer learns increasingly abstract features. For example:

- **Layer 1** learns edges.
- **Layer 3–5** learns textures.
- **Layer 10–20** learns object parts.
- **Layer 30+** learns full objects (faces, animals, vehicles).

3.1 Benefits of Increasing Depth

1. **Hierarchical Feature Learning**
   Deep networks can learn multi-level structures in images, starting from simple edges to high-level concepts.
2. **Better Accuracy**
   More layers often mean more learning capacity, which helps in solving complex tasks.
3. **Improved Generalisation**
   Deep networks understand subtle variations in image structure, improving performance on unseen data.
4. **Ability to model complex tasks**
   Tasks like object detection, segmentation, and large-scale classification benefit from deep architectures.

3.2 Problems with Very Deep Networks

Increasing depth introduces several challenges:

1. **Vanishing and exploding gradients**
   As backpropagation flows through many layers, gradients shrink or grow uncontrollably.
   This makes training unstable.
2. **Higher training cost**
   More layers require exponentially more computation.
3. **Risk of overfitting**
   Deep networks may memorise training data if regularisation is weak.
4. **Hyperparameter tuning becomes difficult**
   The more layers added, the more complex optimisation becomes.

Architectures like ResNet introduced innovations such as **skip connections** to combat these issues.

4. The Combined Effect of Filters and Depth

Filters and depth complement each other. Increasing only one of them usually does not provide optimal performance.

4.1 Deep But Narrow Networks

A deep network with very few filters cannot capture enough variety in features. Even if it has many layers, each layer may extract almost identical patterns, limiting the model's effectiveness.

Such networks tend to:

- Have low accuracy
- Fail to generalise
- Show weak feature diversity

4.2 Shallow But Wide Networks

Shallow networks with many filters can detect diverse patterns early, but without depth, they fail to combine these patterns into higher-level concepts.

These networks:

- Cannot detect abstract structures
- Are less effective for complex datasets
- Perform worse on large-scale tasks

4.3 The Ideal Strategy: Increase Both Gradually

Most successful CNNs increase both filter counts and depth in a structured way:

- Early layers: 16–64 filters
- Middle layers: 128–256 filters
- Deep layers: 512–1024 filters

This mirrors how visual understanding becomes deeper and richer as layers progress

5. Results and Analysis

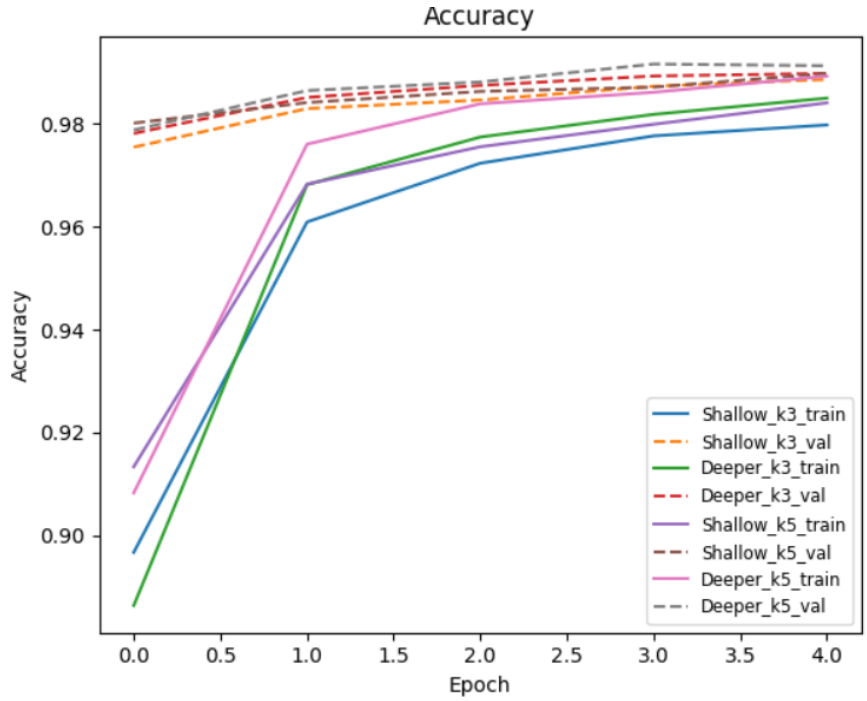
5.1 Accuracy Comparison

The final test accuracy achieved by each model was:

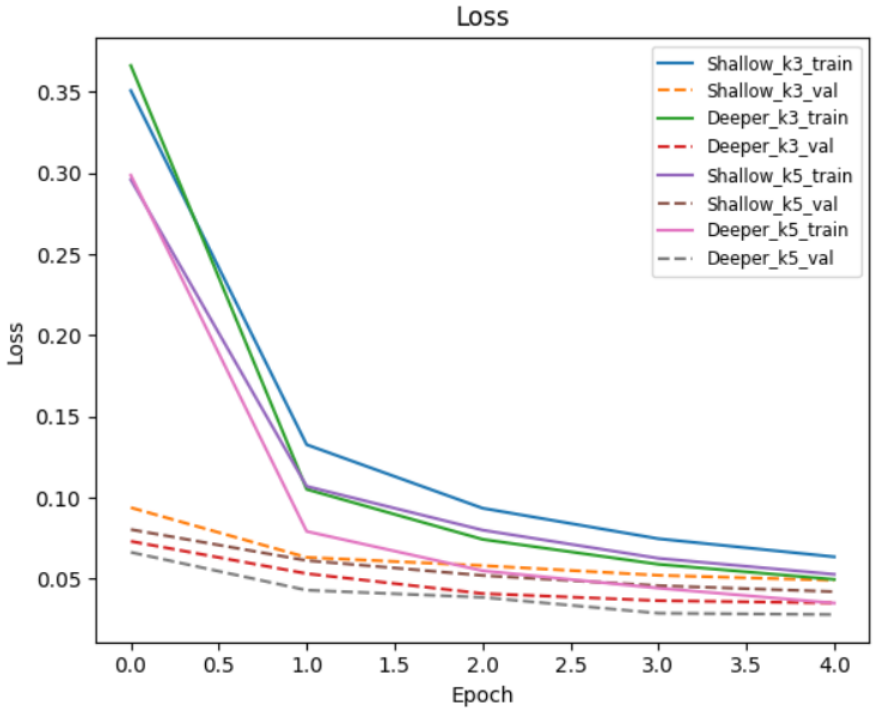
- **Shallow_k3:** 98.44%
- **Deeper_k3:** 99.09%
- **Shallow_k5:** 98.77%
- **Deeper_k5:** 99.04%

These values reveal several important insights.

Accuracy



Loss



6. Case Study: Popular CNN Architectures

Examining real architectures shows how filters and depth influence design.

6.1 LeNet-5 (Shallow, Simple)

- Designed for MNIST digit recognition
- Very few filters
- Only a few layers

Strength: Fast and lightweight
Weakness: Not suitable for complex tasks

LeNet proves that limited depth and filters work only for simple datasets.

6.2 VGG-16 / VGG-19 (Deep and Wide)

VGG introduced the concept of using:

- Many small 3×3 filters
- 16–19 layers
- A systematic doubling of filters at each stage

This approach achieved excellent accuracy but required heavy computation.

VGG shows how deep networks with increasing filters extract powerful hierarchical representations.

6.3 ResNet (Very Deep, With Skip Connections)

ResNet solved the vanishing gradient problem by adding residual connections:

- Extremely deep (50, 101, 152 layers)
- Efficient training despite depth
- High accuracy in large datasets

ResNet demonstrates how very deep networks can be stabilised through architectural innovation.

7. Computational and Memory Considerations

Increasing depth or filters significantly impacts hardware requirements.

7.1 Training Time

More layers and filters → more floating-point operations.
Training becomes slower and requires GPUs or TPUs.

7.2 Memory Usage

Memory stores:

- Weights
- Feature maps
- Gradients

Larger networks often require:

- Smaller batch sizes
- Gradient checkpointing
- Model compression

7.3 Inference Speed

Real-time systems like mobile apps or embedded devices cannot run extremely deep or wide CNNs. Lightweight architectures such as MobileNet, SqueezeNet, or ShuffleNet reduce filter count and depth to match hardware constraints.

This highlights that increasing filters and depth is not always practical for deployment environments.

8. Impact on Model Accuracy and Generalisation

8.1 When Filters and Depth Increase Accuracy

- Datasets with complex high-resolution images
- Tasks requiring fine-grained pattern detection
- Environments with large and diverse training samples

The model learns subtle details and relationships between patterns.

8.2 When Increasing Filters/Depth Does NOT Improve Accuracy

1. **Small datasets**
   Too much capacity leads to overfitting.
2. **Simple tasks**
   Extra filters and layers do not add useful representational power.
3. **Unbalanced architectures**
   If depth and filter count do not match, learning becomes inefficient.
4. **Insufficient hardware**
   Reduced batch sizes slow training and destabilise gradients.

8.3 Diminishing Returns

Beyond a certain point, adding filters or depth yields marginal improvements but drastically increases computation. This has been observed in models like:

- ResNet-152 vs ResNet-101
- VGG-19 vs VGG-16

This illustrates that performance saturates after a certain architectural complexity.

9. Regularisation for Deep and Wide Networks

To prevent overfitting in large networks, several regularisation techniques are essential:

1. **Dropout**
   Randomly drops neurons, reducing memorisation.
2. **Batch Normalisation**
   Stabilises training and helps deeper networks converge.
3. **Data Augmentation**
   Synthetic image variations improve generalisation.
4. **Early Stopping**
   Prevents excessive training.
5. **Skip Connections**
   Allow very deep models to learn effectively.
6. **Weight Decay / L2 Regularisation**
   Prevents weights from growing too large.

Regularisation enables larger networks to perform well without sacrificing generalisation.

10. Conclusion

Filters and depth are fundamental components that determine the performance and learning ability of Convolutional Neural Networks. Filters define how many types of patterns the network can recognise, while depth determines how many hierarchical levels of representation it can learn. Increasing either dimension increases learning capacity, but also computational cost and risk of overfitting.

The best-performing CNNs balance these elements carefully—gradually increasing filter counts through the layers while maintaining appropriate depth. Architectures like VGG, ResNet, and MobileNet demonstrate how thoughtful design enables networks to achieve high accuracy, efficiency, and stability.

Ultimately, understanding the effect of filters and depth allows practitioners to design CNNs tailored to their applications, balancing performance with computational constraints. Effective CNN design is not just about making networks larger, but making them smarter, structured, and optimally balanced

11. References

1.LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, *521*(7553), 436–444

2.Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

3. **Dumoulin, V., & Visin, F. (2016).** A guide to convolution arithmetic for deep learning. arXiv preprint arXiv*:1603.07285.*

4. **Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).**ImageNet classification with deep convolutional neural networks.Advances in Neural Information Processing Systems *(*NIPS*)*, 25, 1097–1105.(Introduces AlexNet—demonstrates the impact of filter count and depth).

5. **Keras Documentation (2024).** Convolutional layers, training utilities, and visualisation tools.