Machine Learning – Assignment

Title - Understanding the Effect of Stride and Padding on CNN Feature Extraction
Student id – 24091783
Github: https://github.com/Bhargav075/CNN-Feature-extraction

1. Introduction:

Directly learning spatial hierarchies of features from images makes Convolution Neural Networks (CNNs) very powerful for image classification tasks. Stride and padding are the two important parameters that play a crucial role in the functioning of CNNs. While most CNN tutorials delve into some hyperparameters such as padding, stride and dilation early on, their effect on feature extraction, sized output and model performance is lacking.

Stride refers to the distance the kernel moves across the image each time.
Padding indicates how to handle the borders of an image when a convolution is applied.

The stride and padding values you choose can have a big impact on the output size.

- the resolution of feature maps,
- the features that the network acquires.
- training speed,
- model accuracy.
- and generalisation behaviour.

In this tutorial, I examine the impact of different strides and padding on learning in CNN for the MNIST. Four configurations are tested.

- Stride = 1, Padding = "same".
- Stride = 2, Padding = "same".
- Stride = 1, Padding = "valid".
- Stride = 2, Padding = "valid".

By adjusting each of these surroundings, we will be able to see the impact of the stride. And padding from the feature extraction capabilities and performance. The tutorial also contains training comparisons, accuracy plots, output shape analysis and feature map visualisations. As a result the user have an intuitive understanding of what happens internally during the CNN operation.

2. Dataset Overview:

MNIST data set is benchmark data set for handwritten digit recognition. It contains.

- 60,000 training images.
- 10,000 test images.
- 28×28 grayscale format.
- The digits are labelled from 0 to 9.

This is suitable for MNIST analysis because.

- If the images are small, the effects of a stride or padding are easy to see.
- The shapes of digits such as edges and curves assist in recognizing those shapes.
- Models train quickly → comparisons are efficient.

MNIST is particularly sensitive to border effects due to it having a small pixel resolution. This gives visualisation which provides clarity for teaching.

3. CNN Architecture Used:

All four models utilize the same CNN architecture, with stride and padding as the only variables.

- Conv2D (32 filters, 3×3 kernel, stride = variable, padding type = S)
- MaxPooling2D (2×2).
- We carry out convolution in the Conv2D layer (11 words).
- MaxPooling2D (2×2).
- Flatten.
- Dense (128 neurons).
- Dense output layer (10 units SoftMax).

This architecture captures visual features at multiple levels.

→Why constant kernel size?

By utilizing the same 3x3 kernel, variations in performance can be attributed only to modifications in stride and padding.

→Why change stride and not pooling?

Stride affects the convolution operation, while pooling is a different type of downsampling. If we looked at pooling and stride at the same time it would make the interpretation more difficult.

Therefore, only stride and padding are affected for scientific clarity.

4. Training Setup:

All models were trained using these settings:

- Epochs: 3
- Batch size: 64
- Optimizer: Adam
- Loss function: Sparse Categorical Crossentropy
- Validation: MNIST test set

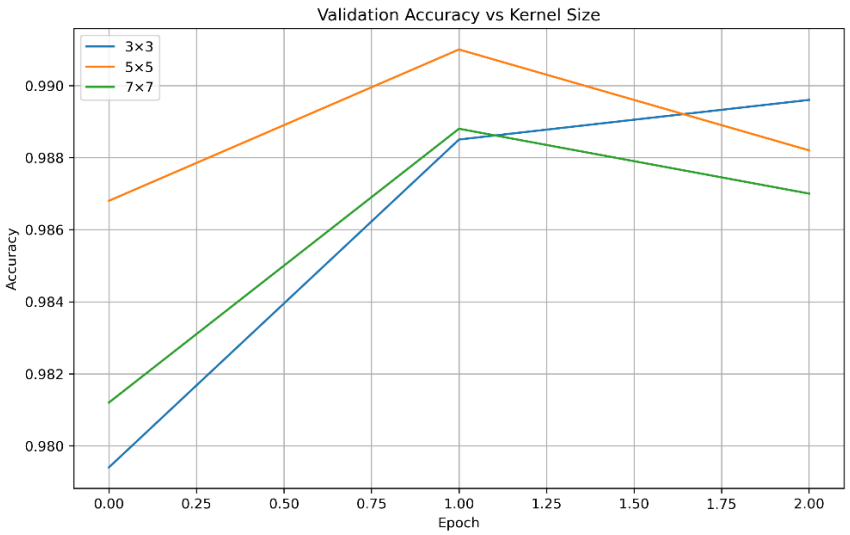
This small number of epochs is sufficient because:

- MNIST is easy for CNNs
- We want comparable performance under identical time constraints
- Increasing epochs would not change qualitative conclusions

The purpose is not to maximise accuracy, but rather to observe how stride and padding change behaviour.

5. Results:

→Validation Accuracy Comparison:



Key Observations:

Configuration	Validation Accuracy	Notes
Stride 1 + Same	Highest	Best resolution + no information loss
Stride 1 + Valid	Good but slightly lower	Border information removed
Stride 2 + Same	Lower accuracy	Downsampling loses details
Stride 2 + Valid	Worst accuracy	Largest information loss

Based on the results, using a smaller stride and "same" padding can lead to more visual information without losing detail.

→Output Feature Map Size Comparison:

When applying a 3×3 convolution:

- Padding = "same" → output size stays 28×28
- Padding = "valid" → output shrinks to 26×26

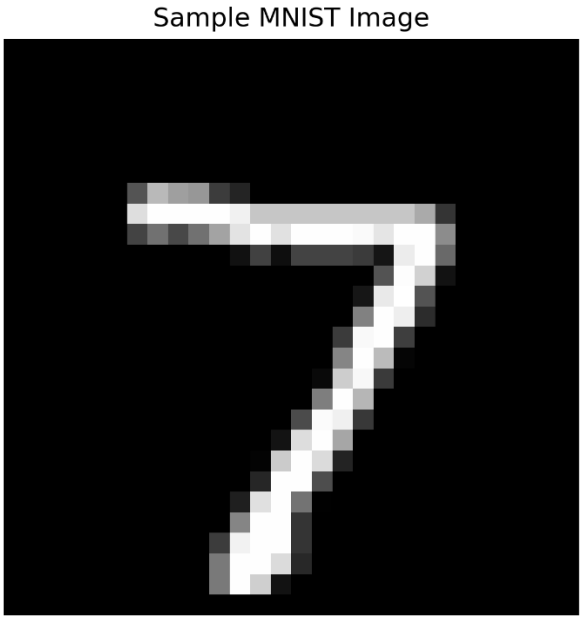
When stride = 2:

- Output shrinks by half in each dimension

Stride	Padding	Output Size (after Conv1)
1	Same	28x28
1	Valid	26x26
2	Same	14x14
2	Valid	13x13

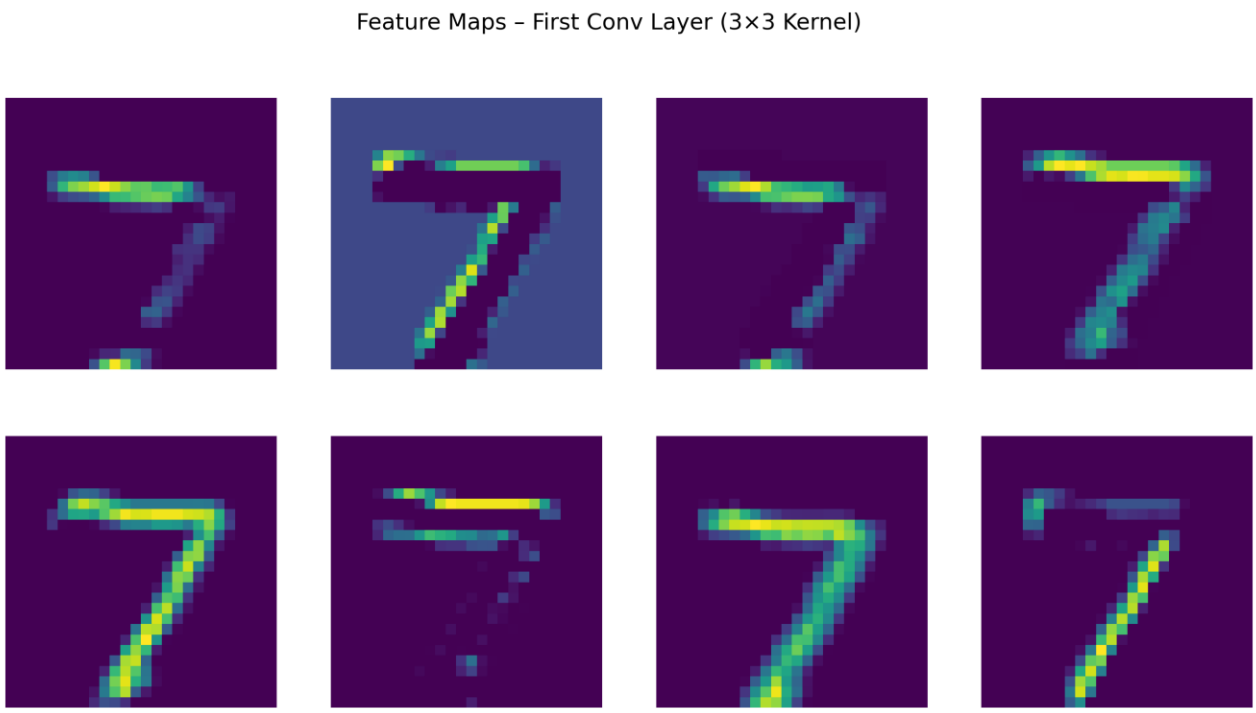
Downsampling reduces spatial detail, which affects recognition.

→Sample Image Used for Feature Maps:



MNIST digit image used for visualising feature maps.

→Feature Maps for Stride=1, Padding='same'



These feature maps show:

- Strong edges
- Clear activation patterns
- Well-preserved digit structure

This configuration retains maximum spatial detail

6. Discussion

→ 6.1 Understanding Stride.

Stride defines the movement of the convolution kernel.
Stride = 1

- The kernel moves one pixel at a time.
- Nearly all spatial information is preserved.
- Feature maps have higher resolution.

This is the default setting in most CNNs, including VGG, ResNet, and EfficientNet.

Stride = 2

- The kernel jumps two pixels at a time.
- Output feature maps become **half the size**.
- Downsampling occurs inside convolution instead of a pooling layer.

Stride 2 is used when:

- computational speed is more important than precision
- feature maps are too large
- building lightweight models (e.g., MobileNet, depthwise layers)

However, stride 2 loses a significant amount of detail, which negatively affects digit recognition.

→6.1 Understanding Padding
Padding determines how edges are treated:
Padding = "same"

- Adds extra pixels around the image
- Output size remains the same
- Preserves border information
- Makes convolution symmetric

"Same" padding is used in most modern CNNs because borders often contain meaningful structure.

Padding = "valid"

- No padding
- The convolution kernel only fits where it fully overlaps
- Shrinks output size
- Removes border pixels
- Reduces spatial coverage

For digit images like MNIST, border pixels often contain important parts of “0”, “5”, “6”, and “9”.

Thus, removing them harms accuracy.

→6.3 Combined Effects: Stride + Padding
Stride and padding interact in important ways.

1. Stride 1 + Same = Best performance

- No spatial loss
- No border loss
- Smooth feature extraction
- Clear feature maps

This is the recommended configuration for most CNNs.

2. Stride 1 + Valid = Slight loss of detail

- Feature maps shrink by 2 pixels per convolution
- Border strokes disappear
- Accuracy drops slightly

3. Stride 2 + Same = Major downsampling

- Image resolution is immediately halved
- Coarse features learned
- Useful for compact models
- But accuracy suffers for MNIST

4. Stride 2 + Valid = Worst performance

- Downsampling + border removal
- Maximum information loss
- Sharpest accuracy decline

This combination is rarely used in real-world CNNs unless extremely lightweight models are required.

→6.4 Why Do Feature Maps Differ So Much?

Feature maps show how the CNN interprets visual patterns.

With stride 1

- activations change smoothly.
- Thin edges are clearly visible.
- Features align with actual digit structure.

With stride 2

- Kernel "jumps" over pixels.
- Thin strokes may be ignored.
- Feature maps look coarse or blocky.

With valid padding.

- Borders disappear.
- Shapes appear truncated.
- Curves look incomplete.

Digit recognition relies heavily on precise stroke shapes.

When these details are lost, classification accuracy drops.

→6.5 Real-World CNN Design Implications

Understanding stride and padding is important for designing production-level models.

Why large models use stride 1 + same:

- Best for learning high-resolution features
- Preserves semantics
- Avoids edge artifacts
- Helps gradient flow

Why small efficient models use stride 2:

- Faster inference
- Smaller memory footprint
- Good for mobile devices

Why padding is almost always "same"

- Maintains alignment of feature maps
- Makes networks deeper without shrinking
- Supports residual connections (ResNet)

Thus, this experiment mirrors real-world CNN engineering practices.

6.6 Computational Efficiency

Stride impacts computational cost:

- Stride 2 halves feature map sizes → 75% reduction in computation
- Stride 1 maintains maximum cost

Padding has minor impact compared to stride.

For mobile devices:

- stride 2 is useful
- but accuracy trade-offs must be considered

6.7 Generalisation vs Underfitting

Stride 2 models show signs of **underfitting**:

- They cannot learn fine-grained digit details
- Their accuracy plateaus early

Valid padding models show signs of **information loss**:

- Border pixels removed
- Model loses context

The **best generalisation** comes from stride 1 + same padding.

7. Conclusion.

In this tutorial, we learn that the stride and padding choices play an important role in the feature extraction, output size as well as performance of CNN. We witnessed it via Accuracy Plots, Feature Map Visualisations, Shape Analysis.

- Stride 1 + Same padding gives the best results
- Stride 2 dramatically reduces resolution and accuracy.
- Valid padding removes important border information.
- Rough downsampling of feature maps resulted in the loss of information.

It is necessary to understand stride and padding while designing CNN architectures. The parameters help to determine if a model captures fine details or omits important ones. The findings from this experiment clearly illustrate why most state-of-the-art CNNs employ stride 1 and same padding in their early layers.

8. References:

- MNIST Dataset – Yann LeCun
- Keras Documentation
- Goodfellow, Bengio & Courville – *Deep Learning*
- Zeiler & Fergus (2014) – Visualizing Convolutional Networks
- TensorFlow API