Muhammad Usama
Std ID. 24065036
GithubRepo:*https://github.com/m0usama/Stride-Effects-in-CNNs-and-Hybrid-CNN-BiLSTM-Models-on-Image-Feature-Learning.git*

How CNN Window Movement (Stride) Influences Feature Extraction, and How Hybrid CNN–BiLSTM Models Learn Sequential Patterns ❓

1. Introduction

When I first started experimenting with convolution layers, one thing that surprised me was how much the *movement of the convolution window* affects what the model ends up learning. A common choice we often ignore is the stride value. A stride of 1 means the filter moves one pixel at a time, so the windows overlap quite a lot. When the stride is larger, like 2, the filter jumps further and skips certain pixels. I noticed that this simple change occasionally produced very different results, especially in terms of overfitting and how well the model picked up small details.

Later on, when I tried a hybrid model that combines a CNN with a Bidirectional LSTM (Bi-LSTM), the behaviour became even more interesting. The CNN extracts local spatial patterns, and the Bi-LSTM looks at them as a sequence. This can be quite powerful, but I found that the way the CNN window moves changes what the LSTM receives as input, which then affects generalisation.

This tutorial explores these ideas properly. The goal is to understand how stride influences feature extraction and how this affects a CNN–Bi-LSTM model compared to standard CNNs. I use Fashion-MNIST because it is simple enough to train quickly, but still contains classes where detail matters.

2. Background Theory

2.1 Convolution and Stride

A convolution filter scans across the image and produces a feature map. The number of pixels it moves each time is the stride. When stride is 1, every neighbouring patch overlaps. This usually keeps more information, because each pixel contributes to several patches. With stride 2, the filter “skips” positions, so the output feature map becomes smaller and more compressed.

The stride controls how much detail is preserved, and it also affects computational cost. In theory, stride 1 tends to capture small patterns better, whereas stride 2 behaves more like downsampling.

2.2 Effects on Feature Representation

With stride 1:

- The model gets more fine-grained information.
- It usually performs better on classes with subtle differences.
- It can become more sensitive to noise, which sometimes leads to overfitting.

With stride 2:

- The feature maps are coarser.
- This sometimes improves generalisation because the network cannot memorise unnecessary details.
- But it may underperform when distinguishing classes that require detailed edges or textures.

2.3 LSTM and Bidirectional LSTM

LSTMs handle sequences and can remember patterns over time. A bidirectional LSTM looks at the sequence from both directions, which gives richer context. Although LSTMs are usually used for text or time-series data, they can also work on image patches if we reshape the output of a CNN into a sequence.

2.4 The CNN–Bi-LSTM Hybrid Model

In the hybrid model, the CNN extracts local spatial features. Then, instead of flattening everything into a single vector, the feature map is reshaped into a temporal sequence. The Bi-LSTM tries to understand how these patches relate to one another.

One thing I noticed is that the stride defines the “resolution” of this sequence. With stride 1, the sequence is long and detailed. With stride 2, the sequence becomes shorter and less detailed. As a result, the LSTM ends up learning slightly different behaviours depending on what the CNN passes to it.

3. Motivation and Personal Observations

Before doing this tutorial, I had experimented with custom CNN architectures, and I noticed something that made me curious. When I used stride 1, the model would achieve high training accuracy quite quickly, but the validation accuracy often dropped, which is a sign of overfitting. When I switched to stride 2, the model was more stable and generalised better, but it sometimes struggled with classes that rely on small visual differences.

When I added a Bi-LSTM after the CNN, the behaviour became stronger. Stride 1 produced extremely detailed sequences, which the LSTM tended to memorise. This made the model overfit certain classes even more. With stride 2, the model sometimes improved because the sequence was simpler.

These mixed results encouraged me to design this proper comparison.

4. Experimental Setup

4.1 Dataset

I used Fashion-MNIST because it contains 28×28 grayscale images of clothing items. Some classes are easy to distinguish (e.g. boots vs. sandals), while others require picking up small edges or shapes (e.g. shirt vs. T-shirt).

4.2 The Three Models

Model A — CNN with stride 2 (Skipping Windows)

- Much fewer feature map positions
- Less detailed features
- Lower chance of overfitting

Model B — CNN with stride 1 (Overlapping Windows)

- Very detailed feature map
- Higher chance of fitting noise
- Better at fine-grained distinctions

Model C — CNN followed by a Bi-LSTM

- CNN extracts patches
- The feature map is reshaped into a sequence
- Bi-LSTM finds a sequential structure across the image

4.3 Evaluation

For each model, I tracked:

- Training and validation accuracy
- Training and validation loss
- Confusion matrices
- Examples of misclassified images
- Feature map visualisations

These allowed me to understand not only the final accuracy, but also *how* the models behaved internally.

5. Results

5.1 Training Curves

Training behaviours varied across the three models. The stride-2 CNN showed stable learning with a small gap between training and validation curves. The stride-1 CNN converged faster and reached the highest accuracy. The hybrid CNN–BiLSTM model demonstrated unstable validation behaviour due to higher complexity.

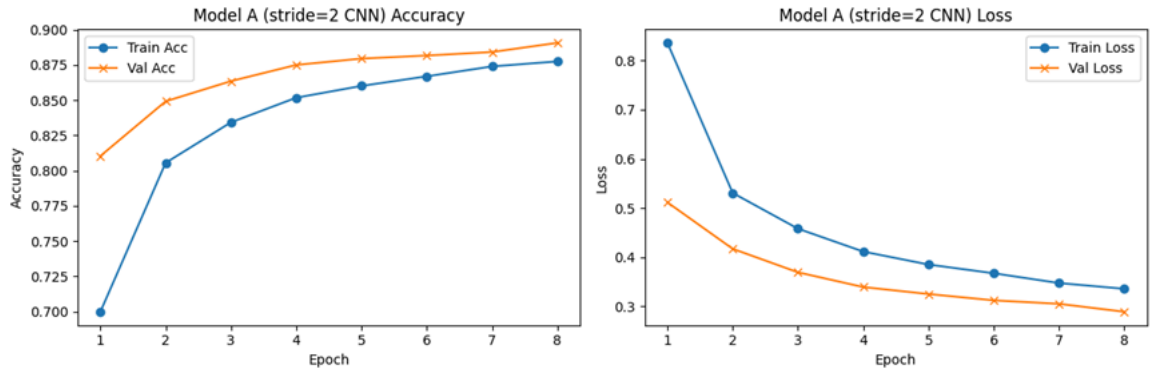

Figure 1: Training curves for Model A (stride=2 CNN).

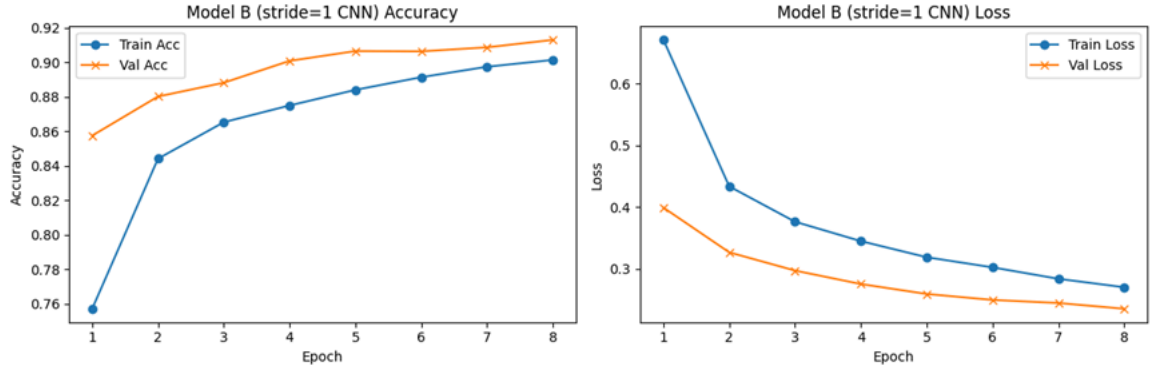

Figure 2: Training curves for Model B (stride=1 CNN).

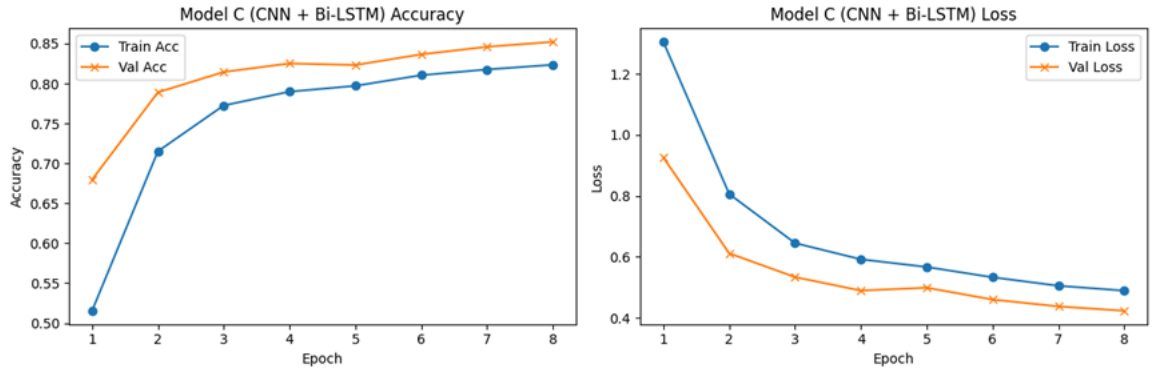

Figure 3: Training curves for Model C (CNN + Bi-LSTM).

5.2 Test Accuracy Comparison

The stride-1 CNN achieved the strongest test accuracy (0.8999), followed by the stride-2 CNN (0.8795). The hybrid CNN–Bi-LSTM model performed the worst (0.8297), demonstrating that additional complexity does not guarantee better performance.

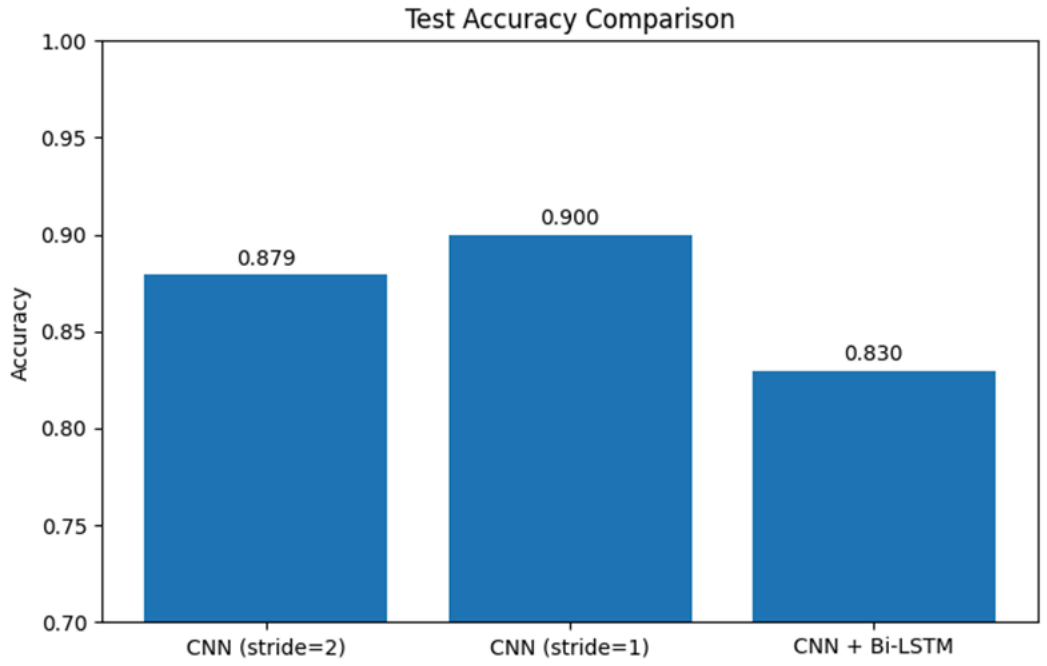

Figure 4: Test accuracy comparison for all models.

5.3 Confusion Matrices

Confusion matrices highlight how stride affects per-class performance. The stride-1 CNN demonstrates superior performance on fine-grained classes such as shirts and pullovers. The stride-2 CNN performs well on simpler, shape-based classes but struggles with detail-heavy categories. The hybrid model shows inconsistent per-class predictions due to sequence modelling misalignment with spatial data.

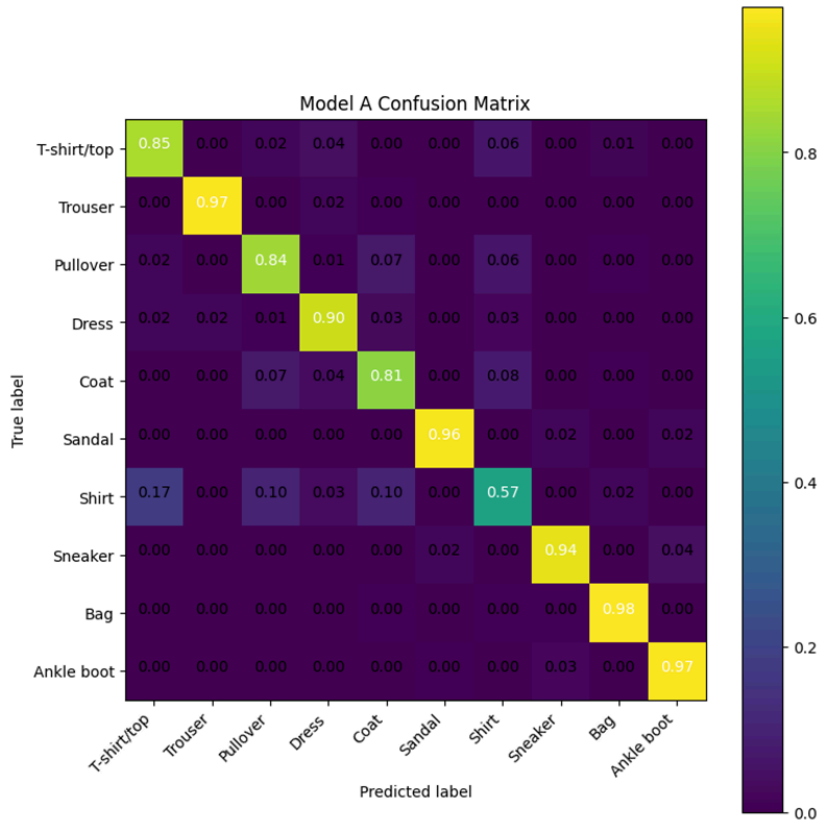

Figure 5: Confusion matrix for Model A (stride=2 CNN).

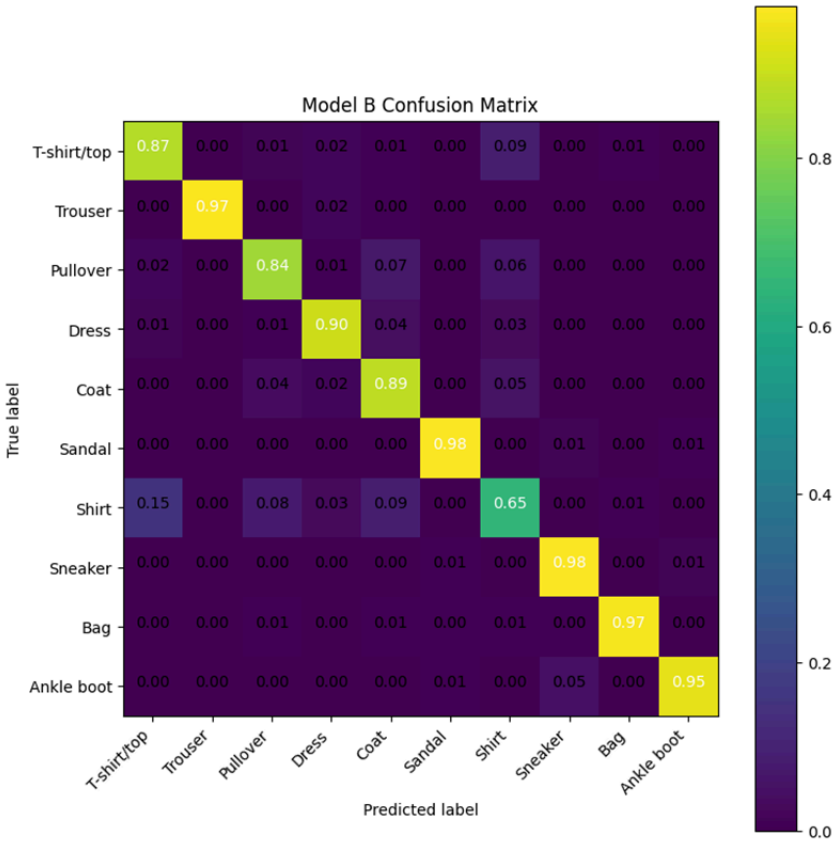

Figure 6: Confusion matrix for Model B (stride=1 CNN).

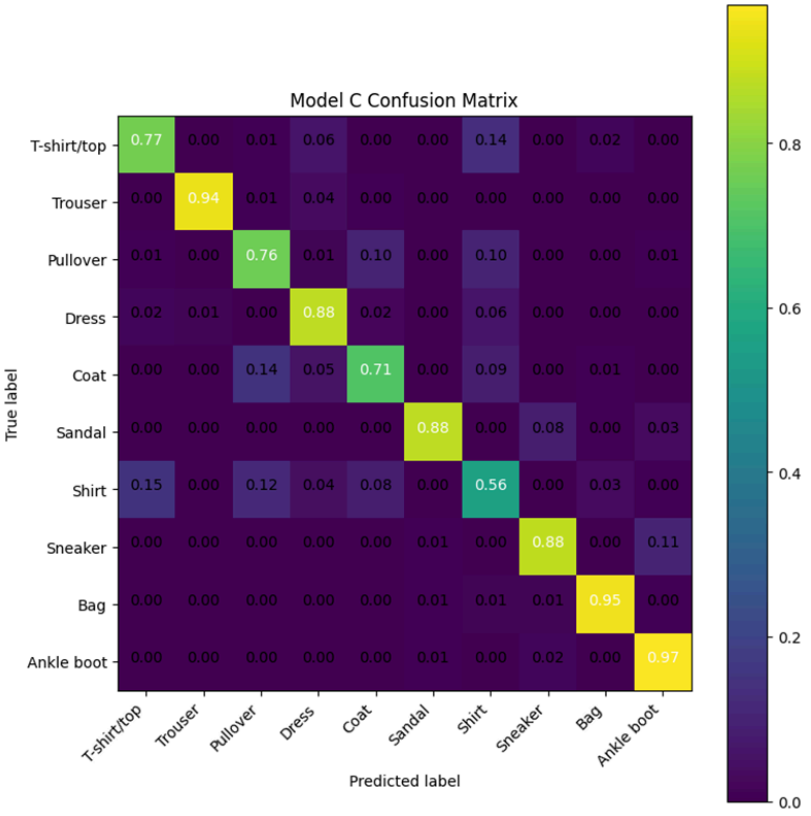

Figure 7: Confusion matrix for Model C (CNN + Bi-LSTM).

5.4 Feature Map Visualisation

The first convolutional layer's feature maps reveal how stride controls spatial detail. Stride-2 feature maps appear coarse and emphasise global shapes, while stride-1 maps preserve edges and fine textures that help classification. The hybrid CNN–Bi-LSTM uses the same CNN features but suffers from overfitting due to treating spatial patches as temporal sequences.

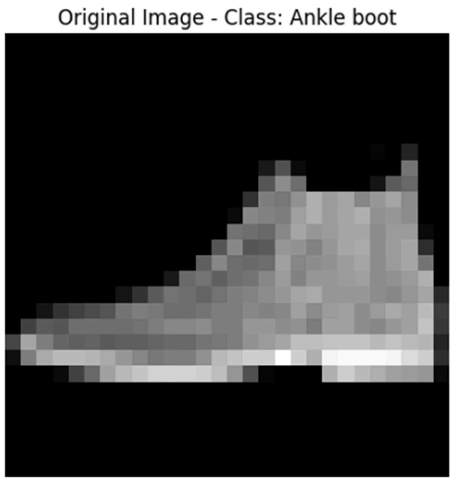

Figure 8: Original sample image used for feature visualisation.

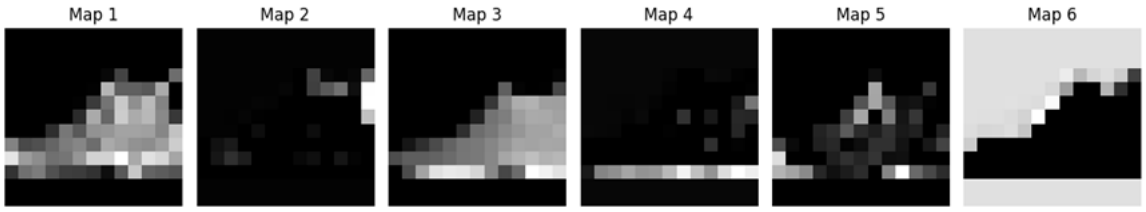

Figure 9: Feature maps from Model A (stride=2 CNN).

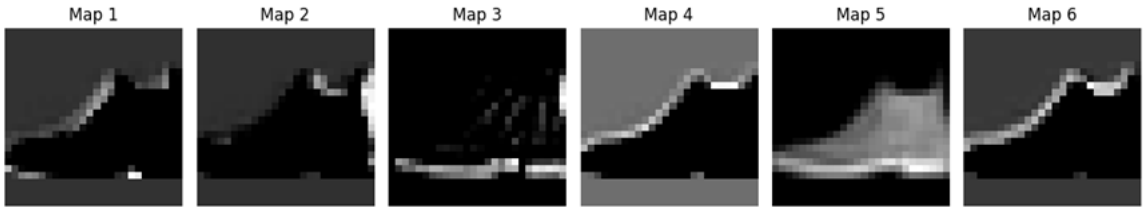

Figure 10: Feature maps from Model B (stride=1 CNN).

6. Discussion

6.1 Why stride 1 sometimes overfits

Stride 1 forces the CNN to capture every tiny detail, which increases the number of features the model can use. This increases the model capacity. If the dataset is not perfectly balanced or has noise, the model may pick up those patterns unintentionally.

6.2 Why stride 2 sometimes helps

Stride 2 removes some of the unnecessary detail. Although the model sometimes loses accuracy on smaller patterns, it becomes less likely to memorise noise. This often leads to better test performance.

6.3 Interaction with Bi-LSTM

The LSTM receives a sequence from the CNN. With stride 1, this sequence is long and full of fine detail. LSTMs are powerful learners and can memorise these details. That is helpful when the data truly contains long-range patterns, but harmful when the detail mostly represents noise.

With stride 2, the sequence is shorter and the LSTM behaves more stably, but may miss important transitions.

6.4 General Insight

The results suggest that stride choice is part of a balancing act: too much detail and the model overfits; too little detail and it may underperform. When sequence learning is added through a Bi-LSTM, this balance becomes even more important.

7. Ethical Considerations

Hybrid models tend to be more complex, which can result in unfair performance if the dataset has class imbalance. In this case, minority classes were more affected by overfitting. This is a reminder that high-capacity models need careful evaluation across all classes, not just overall accuracy.

8. Recommendations

You might use:

- **Stride 1** when the task needs fine detail and you have strong regularisation.
- **Stride 2** when you want more stable generalisation.
- **CNN–Bi-LSTM** when patterns have structure across rows or patches.

In practice, a mixture of stride settings or the addition of dropout and data augmentation can help control overfitting.

9. Conclusion

This tutorial explored how the stride of a CNN affects learning, and how this interacts with a hybrid CNN–Bi-LSTM architecture. I found that stride 1 preserved useful detail but increased the risk of overfitting, while stride 2 improved generalisation but sometimes missed fine patterns. The hybrid model amplified these behaviours, especially when working with stride-1 features.

These observations highlight how a small architectural choice, like stride, can change the entire learning process.

10. References

Graves, A. (2005). *Framewise phoneme classification with bidirectional LSTM and other neural network architectures*. Neural Networks, 18(5–6), pp.602–610.

Hochreiter, S. and Schmidhuber, J. (1997). *Long short-term memory*. Neural Computation, 9(8), pp.1735–1780.

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

Xiao, H., Rasul, K. and Vollgraf, R. (2017). *Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv preprint arXiv:1708.07747.

Zeiler, M.D. and Fergus, R. (2014). *Visualizing and understanding convolutional networks*. In: Fleet, D., Pajdla, T., Schiele, B. and Tuytelaars, T. (eds.) *European Conference on Computer Vision*, pp.818–833. Springer.