Comparative Evaluation of a Custom CNN, Mobilenetv2, and A Lightweight Vision Transformer for Automated Woven-Fabric Weave-Pattern Classification

Pugazhenthi T.[1,3*] , Jebathangam J.[2]

Department of Information Technology, School of Computing Sciences, Vels Institute of Science, Technology and Advanced Studies(VISTAS),Chennai, India

***Corresponding Author**

DOI: https://doi.org/10.51244/IJRSI.2026.1309000097

Received: 09 September 2026; Accepted: 14 September 2026; Published: 08 October 2026

ABSTRACT

Weave-pattern identification is one of the more tedious quality-control steps in textile manufacturing, and most mills still rely on a trained inspector holding a swatch up to the light. This paper benchmarks three deep-learning classifiers — a custom convolutional neural network (CNN) built from scratch, a transfer-learning pipeline built on MobileNetV2, and a compact, single-block Vision Transformer (ViT) implemented without ImageNet pre-training — on the task of sorting 224×224 fabric-swatch photographs into four weave families: honeycomb, plain, satin, and twill. All three models were trained under an identical 80/20 split of 1,496 images (1,197 train / 299 validation) for ten epochs, with the input pipeline deliberately desaturated (RGB → grayscale → RGB) so that classification decisions could not lean on dye color. The transfer-learning model reached a validation accuracy of 95.99% (peak 97.32% at epoch 6) with a weighted F1-score of 0.96, comfortably outperforming the from-scratch CNN (52.51% accuracy, F1 = 0.51) and the untrained ViT (68.90% accuracy, F1 = 0.69). Confusion-matrix analysis shows that the plain and twill classes are the primary sources of error for the weaker models, and a qualitative single-image test on the best model reproduces this exact confusion, misclassifying a twill sample as plain with 96.78% confidence. The results argue for pretrained, parameter-efficient backbones over either shallow custom CNNs or non-pretrained transformers when the available fabric dataset is in the low thousands of images, and they surface a preprocessing detail — an unintended double normalization inside the custom-CNN branch — that likely worsened that model's already modest showing.

Index Terms — Convolutional neural network, transfer learning, MobileNetV2, Vision Transformer, weave pattern, fabric classification, texture recognition, textile quality control.

INTRODUCTION

Weave structure is, along with fiber content and finish, one of the three properties that define a woven fabric. Plain, twill, satin, and the more decorative honeycomb weave each arrange warp and weft yarns in a distinct geometric repeat, and that repeat shows up visually as a texture pattern long before a garment or upholstery piece reaches a customer. Getting the weave classification wrong at intake — mislabeling a twill lot as plain, for instance — has downstream consequences for pricing, sorting for recycling, and matching fabric to the correct finishing process. For decades the identification step has been performed by a trained textile technician using a magnifying glass or a thread counter, a method that is accurate in skilled hands but slow, inconsistent between inspectors, and difficult to scale to the throughput of a modern mill [1], [2].

Convolutional neural networks changed the calculus for this kind of texture-recognition problem soon after they demonstrated large gains on natural-image classification benchmarks [21], [22]. Because CNNs learn hierarchical, translation-invariant filters directly from pixels, they sidestep the hand-crafted-feature bottleneck — gray-level co-occurrence matrices, Gabor filters, and similar descriptors — that limited earlier machine-vision approaches to weave recognition [1], [2]. Deeper CNN designs built on residual connections pushed image-

recognition accuracy further still [20], and once these networks could be pretrained on a large, diverse image corpus such as ImageNet [25], reusing their learned filters via transfer learning became an attractive shortcut for domains, like textiles, where labelled data is comparatively scarce. More recently, attention-based architectures originally built for language modelling have been repurposed for vision, on the premise that global self-attention can capture the long-range periodicity of a weave repeat more naturally than a stack of local convolutions [4], [5].

What is less settled, and what this paper tries to pin down with a controlled, same-dataset comparison, is how these three broad strategies — a small CNN trained end-to-end from random initialization, a large CNN pretrained on natural images and reused via transfer learning, and a Transformer built from scratch without any pretraining — actually compare once the dataset is modest in size (here, 1,496 fabric images across four classes) and the training budget is fixed (ten epochs, no data augmentation, no learning-rate scheduling). This is a realistic scenario for a small-to-mid-size textile business or a research group without access to a large, professionally curated weave dataset, and the literature on fabric-specific deep learning has so far concentrated more heavily on defect detection than on this three-way architectural comparison for pure weave-type classification [6]–[8].

The contribution of this paper is threefold. First, it reports a same-conditions benchmark of a custom CNN, MobileNetV2-based transfer learning, and a lightweight ViT on a four-class weave dataset, using identical splits, optimizer, loss function, and epoch budget. Second, it reports full per-class precision, recall, and F1-scores alongside confusion matrices for all three models, rather than headline accuracy alone, which turns out to matter because the models disagree sharply about which classes are hard. Third, it documents a preprocessing choice — converting every input image to grayscale before re-expanding it to three channels — that was made specifically to force the models to rely on weave texture rather than yarn or dye color, and it discusses a normalization inconsistency uncovered during the experiment that plausibly disadvantaged the custom CNN relative to the other two architectures.

Related Work

A. Handcrafted-Feature and Classical Machine-Learning Approaches

Early automated weave-recognition systems paired texture descriptors such as the gray-level co-occurrence matrix or Gabor wavelets with a shallow classifier. Jing et al. combined GLCM and Gabor features with principal component analysis and a probabilistic neural network to separate plain, twill, and satin weaves, reporting close to 95% accuracy once the two feature families were fused [9]. Khan et al. proposed a biologically inspired pipeline, modelling the visual-cortex hierarchy with an HMAX-style descriptor and a deep extreme-learning-machine classifier, and showed it was tolerant to yarn-color changes when recognizing weave and yarn color jointly [10]. Approaches of this kind work well within the lighting and dataset conditions they were tuned for, but the features themselves are brittle: rotation, illumination change, and yarn-color variation all shift the descriptor distribution in ways a fixed handcrafted pipeline cannot absorb without re-tuning [1], [2].

B. CNN-Based Weave and Fabric Classification

Hussain et al. were among the first to apply a pretrained ResNet-50 with data augmentation directly to woven-fabric pattern images, training end-to-end rather than hand-engineering features, and reported strong recognition rates across several weave classes on their own image set [3]. Akram, Butt, and Qureshi more recently modified a convolutional network specifically for weave-pattern classification, motivated by the same concern that drives the present paper — that generic architectures trained on natural-image datasets do not automatically transfer their filters well to repetitive, low-color-variance textile textures without some architectural adjustment [11].

A large adjacent literature exists on fabric defect detection rather than weave-type classification. Kahraman and Durmusoglu's 2023 review surveys two decades of deep-learning fabric-defect work and finds an accelerating shift from patch-based CNN classifiers toward end-to-end detectors [6]. Recent defect-detection systems illustrate the range of current CNN engineering in this domain: Nasim et al. benchmarked YOLOv8, YOLOv5, and a MobileNetV2-SSD-FPN-Lite pipeline on an industrially sourced dataset and found the lightweight YOLOv8 variant the best accuracy–speed trade-off [7]; Raghuvardhan Babu and Sastry modified MobileNetV2

specifically to control the computational cost of on-line defect inspection [12]; Wang et al. combined a YOLOv8 backbone with a vision state-space block and an enhanced channel-attention module (Fab-ME) to push small-defect sensitivity further [8]; and Mei et al. and Liu et al. each proposed lighter YOLOv8-family variants aimed squarely at deployability on constrained hardware [13], [14]. Zhang et al. took a different route for defect detection without labelled defects, using a mixed-attention multi-scale autoencoder to flag anomalies by reconstruction error [15]. Across this body of work, MobileNetV2 recurs specifically because its inverted-residual, linear-bottleneck design [16] keeps parameter counts and latency low enough for shop-floor cameras, which is one reason it was chosen as the transfer-learning candidate in the present study.

C. Transformer-Based and Hybrid Approaches

The Vision Transformer treats an image as a sequence of fixed-size patches and processes them with the same multi-head self-attention block introduced for sequence modelling by Vaswani et al. [4], dispensing with convolutional inductive bias entirely [5]. Dosovitskiy et al. showed that this pays off once the model is pretrained on very large image corpora, but that a ViT trained from scratch on a comparatively small dataset tends to underperform a similarly sized CNN, because it lacks the built-in locality and translation-equivariance that convolution provides for free [5]. Han et al.'s survey of the vision-transformer literature echoes this point, noting that most reported ViT successes depend on either large-scale pretraining or a hybrid design that reintroduces some convolutional structure [17]. This is directly relevant here: the ViT branch in this paper is a single-block, non-pretrained transformer, so any accuracy gap relative to MobileNetV2 is at least partly attributable to the absence of pretraining rather than to a fundamental unsuitability of attention for texture.

D. Data Efficiency and the Gap This Paper Addresses

Because labelled fabric-weave datasets are expensive to assemble, several authors have argued for aggressive data augmentation as a substitute for scale [18], and general CNN-architecture surveys such as Khan et al. make the same point about deep networks generally: depth and capacity only help once the data volume and training budget can support them [19]. What is comparatively under-reported is a head-to-head, same-dataset, same-epoch-budget comparison of a from-scratch shallow CNN, a frozen pretrained CNN, and a from-scratch ViT, on a purely weave-type — rather than defect — classification task. That comparison, together with a transparent report of what went wrong for the weaker models rather than only what went right for the best one, is what this paper contributes.

MATERIALS AND METHODS

A. Dataset and Split

The dataset consists of 1,496 RGB photographs of fabric swatches spanning four weave classes — honeycomb, plain, satin, and twill — collected as a single local image directory and loaded with Keras's image_dataset_from_directory utility. An 80/20 split with a fixed random seed produced 1,197 training images and 299 validation images; Table I gives the per-class validation counts, recovered from the support column of the classification reports discussed in Section IV (the training-side per-class counts were not logged separately, but follow the same approximate 80/20 proportion). All images were resized to 224 × 224 pixels and batched in groups of 32.

Table I. Dataset Composition And Validation-Set Class Support

Weave class	Approx. share of dataset	Validation-set support (of 299)
Honeycomb	17%	52
Plain	40%	119
Satin	9%	26
Twill	34%	102

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue IX September 2026

Weave class	Approx. share of dataset	Validation-set support (of 299)
Total	100%	299

B. Preprocessing

Every image was passed through a two-stage transform before entering any model. First, the RGB image was converted to single-channel grayscale and then immediately expanded back to three channels by channel replication. This step was a deliberate methodological choice, not a bug: because two swatches of the same weave can be dyed completely different colors, collapsing color information before training forces every model to base its decision on the geometric weave texture rather than on a spurious dye-color shortcut. Second, pixel values were rescaled from the native [0, 255] range to [−1, 1].

which matches the input convention MobileNetV2 was pretrained on when it was originally trained on ImageNet [25]. Because this normalization was applied once, upstream, to both the training and validation pipelines, it also reached the custom-CNN branch — which additionally contains its own internal Rescaling(1/255) layer at its input. The practical effect is that the custom CNN receives values that have already been compressed into the [−1, 1] band and are then divided by 255 a second time, squeezing its effective input range to a narrow band around zero rather than the [0, 1] range the layer was written to expect. This detail is reported here rather than quietly corrected after the fact, because it is a genuine and instructive artifact of building three models on a shared preprocessing pipeline, and it is revisited in the discussion of the CNN's comparatively weak result.

C. Model Architectures

1) **Custom CNN:** The baseline model is a small sequential network: a rescaling layer, three convolution–max-pool blocks with 32, 64, and 128 filters (3 × 3 kernels, ReLU activation), a flattening layer, a 128-unit dense layer with ReLU activation, a dropout layer (rate 0.5), and a 4-way softmax output layer. Unlike deeper modern CNNs, this network includes no batch-normalization layers [23], which likely also contributes to its slower, noisier convergence. The model carries no pretrained weights; every filter is learned from scratch on the 1,197 training images.

2) **MobileNetV2** (transfer learning): The second model uses a MobileNetV2 backbone pretrained on ImageNet [25] with its top classification layer removed and its convolutional weights frozen. MobileNetV2's efficiency comes from an inverted-residual block that expands a low-dimensional input to a wider intermediate representation with a depthwise-separable convolution and then linearly projects it back down, avoiding the non-linearity that would otherwise destroy information in the compressed representation [16]. On top of the frozen backbone, a global average-pooling layer collapses the spatial feature map, followed by a 128-unit dense ReLU layer, a dropout layer (rate 0.5), and a 4-way softmax head. Only this head is trained; the backbone convolutional filters are never updated. The MobileNet-V2 method effectively reduces computing costs by extracting essential information throughout the training phase following the addition of additional layers [26].and dropout rate

3) **Lightweight Vision Transformer**: The third model is a compact, from-scratch ViT rather than a pretrained one. A 224 × 224 × 3 input is patch-embedded by a single 16 × 16, stride-16 convolution with 64 output channels, which is mathematically equivalent to slicing the image into a 14 × 14 grid of non-overlapping 16 × 16 patches and linearly projecting each into a 64-dimensional token — 196 tokens in total. One transformer encoder block follows: layer normalization, a 4-head self-attention layer with key dimension 64, and a residual (skip) connection, followed by a second layer normalization and a 128-unit then 64-unit dense feed-forward sub-block. No explicit positional embedding is added before the block, and no ImageNet pretraining is used — the transformer learns everything, including any notion of patch position, purely from the 1,197 training images. A global average-pooling layer over the token dimension and a 4-way softmax layer produce the final class probabilities.

D. Training Configuration

All three models were compiled with the Adam optimizer [24] at its Keras default learning rate and trained against the sparse categorical cross-entropy loss. Every model was trained for exactly ten epochs with a batch

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue IX September 2026

size of 32, no data augmentation, no early stopping, and no learning-rate schedule, so that any accuracy difference between the three models reflects architectural and pretraining differences rather than an uneven training budget.

E. Evaluation Metrics

Performance is reported with overall accuracy, per-class precision, recall, and F1-score, and the confusion matrix. Precision and recall for a class k are

$$P_k = TP_k / (TP_k + FP_k), \qquad R_k = TP_k / (TP_k + FN_k) \tag{4}$$

and the F1-score is their harmonic mean,

$$F1_k = 2 P_k R_k / (P_k + R_k) \tag{5}$$

RESULTS AND DISCUSSION

A. Headline Accuracy and Training Dynamics

Table II summarizes the final-epoch training and validation metrics for all three models. Fig. 1 plots the training and validation accuracy curves recorded across the ten epochs, and Fig. 2 shows the corresponding confusion matrices on the 299-image validation set.

Table Ii. Final-Epoch (Epoch 10) Training And Validation Metrics

Model	Train acc.	Train loss	Val. acc.	Val. loss	Best val. acc. (epoch)
Custom CNN	49.54%	1.0370	52.51%	1.0222	54.85% (ep. 9)
MobileNetV2	99.42%	0.0252	95.99%	0.1086	97.32% (ep. 6)
Lightweight ViT	77.03%	0.6055	68.90%	0.7639	69.90% (ep. 9)

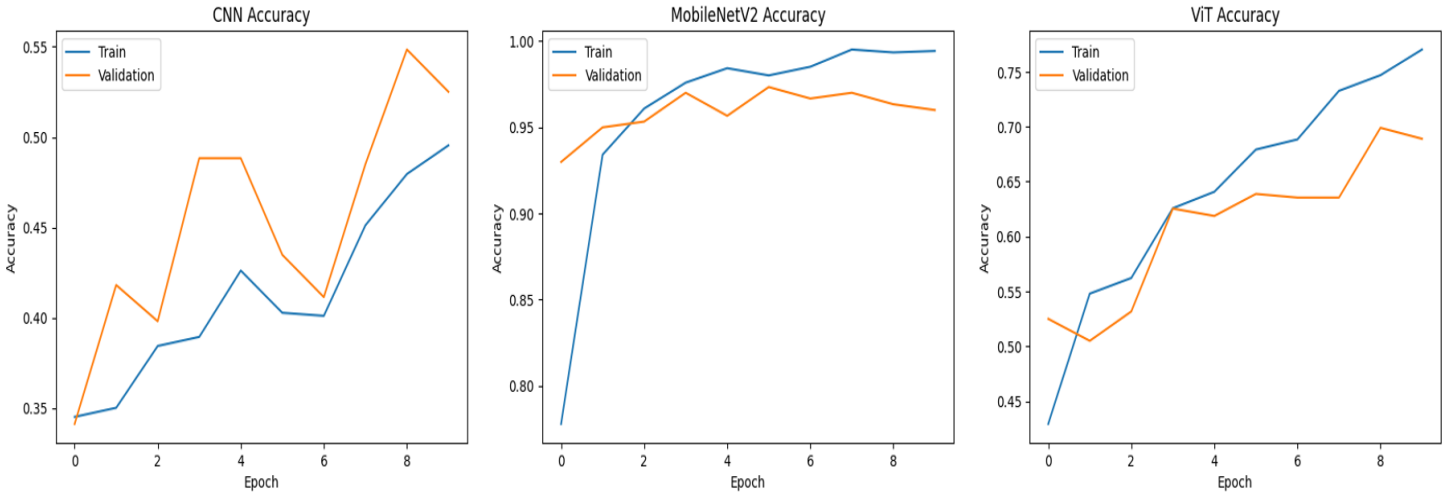


Fig. 1. Training and validation accuracy across ten epochs for the custom CNN (left), MobileNetV2 (centre), and the lightweight ViT (right). Blue = training accuracy, orange = validation accuracy.

MobileNetV2's curve is essentially flat after epoch 4 — it converges almost immediately because only a small classification head is being trained on top of already-informative pretrained features, and its validation accuracy (97.32% at epoch 6) actually edges above the final-epoch figure, a small sign of very mild overfitting on a dataset this size once training continues past its optimum. The custom CNN, by contrast, is still visibly climbing at epoch 10 (49.54% train accuracy) and has clearly not converged within the ten-epoch budget; a longer run might

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue IX September 2026

close some of the gap, but not the majority of it, given how far behind it starts and given the input-scaling issue noted in Section III-B. The ViT sits between the two: it improves in a fairly steady, monotonic way from 42.94% to 77.03% training accuracy over the ten epochs, indicating that a self-attention block can learn useful weave features from this dataset size without pretraining, just more slowly and to a lower ceiling than a network that already carries transferred visual knowledge.

B. Per-Class Performance and Confusion Patterns

Table Iii. Per-Class Precision, Recall, And F1-Score On The 299-Image Validation Set

Model	Class	Precision	Recall	F1-score	Support
Custom CNN	Honeycomb	0.37	0.33	0.35	52
Custom CNN	Plain	0.46	0.76	0.57	119
Custom CNN	Satin	0.83	0.77	0.80	26
Custom CNN	Twill	1.00	0.28	0.44	102
MobileNetV2	Honeycomb	0.98	0.98	0.98	52
MobileNetV2	Plain	0.95	0.97	0.96	119
MobileNetV2	Satin	0.96	0.92	0.94	26
MobileNetV2	Twill	0.96	0.95	0.96	102
Lightweight ViT	Honeycomb	0.55	0.71	0.62	52
Lightweight ViT	Plain	0.70	0.68	0.69	119
Lightweight ViT	Satin	0.66	0.81	0.72	26
Lightweight ViT	Twill	0.80	0.66	0.72	102

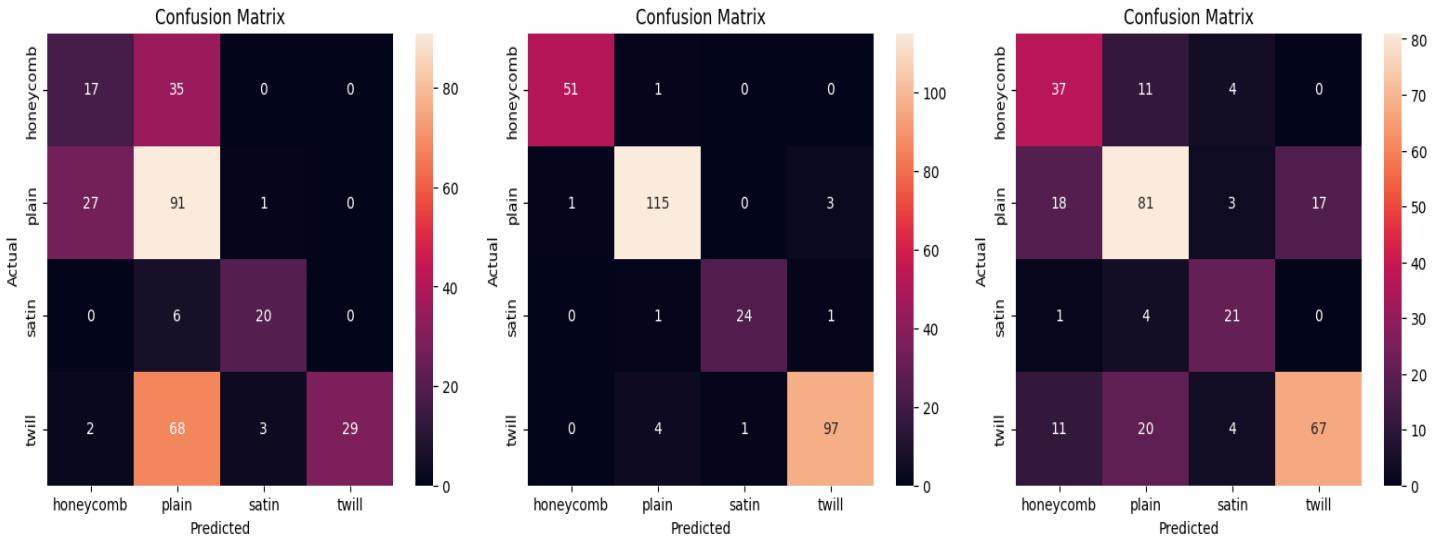


Fig. 2. Validation confusion matrices for the custom CNN (left), MobileNetV2 (centre), and the lightweight ViT (right). Rows are true labels, columns are predicted labels.

The overall weighted F1-scores — 0.51 for the CNN, 0.96 for MobileNetV2, and 0.69 for the ViT — tell only part of the story; the per-class breakdown in Table III shows that the models fail in quite different, and individually informative, ways. The custom CNN's twill recall of only 0.28, next to a twill precision of 1.00, means that whenever the CNN does commit to predicting “twill” it is always right, but it predicts twill far too

rarely — the confusion matrix in Fig. 2 confirms that most true twill swatches are instead pushed into the plain bucket. Given that plain and 2/2 twill weaves share the same basic over-one-under-one yarn logic and differ mainly in the diagonal offset of that pattern, this is a believable failure mode for a shallow, undertrained network that has not yet learned to resolve fine diagonal texture. Satin, by contrast, is comparatively easy even for the weak CNN (F1 = 0.80), likely because satin's long yarn floats produce a smoother, more visually distinct surface than the other three weaves.

MobileNetV2's per-class scores are uniformly high (0.92–0.98 across precision and recall for every class), which is the clearest evidence that a backbone pretrained on a large, diverse natural-image corpus already contains texture-sensitive filters that transfer well to a domain — woven cloth — that ImageNet itself does not contain. The ViT sits in an interesting middle position: its class-by-class pattern (satin recall 0.81, honeycomb recall 0.71, plain and twill both lower at 0.66–0.68) suggests it is not simply a weaker version of the CNN's error pattern; it handles honeycomb noticeably better than the CNN does (0.71 vs. 0.33 recall), plausibly because self-attention over the full 14×14 patch grid can pick up the repeating hexagonal cell structure of a honeycomb weave more directly than the CNN's local 3×3 filters can at the depth this small network reaches.

C. Practical Implications and Limitations

For a deployment scenario — a camera on a sorting line or a handheld inspection app — the practical case for MobileNetV2 over either alternative is not only its accuracy but its training economy: because the backbone is frozen, only a small head is optimized, which converges in a handful of epochs and is far less prone to the slow, unfinished learning curve seen for the custom CNN. This mirrors the broader fabric-defect literature's preference for MobileNetV2-family backbones in constrained deployment settings [7], [12], and it suggests the same preference generalizes from defect detection to weave-type classification.

Three limitations temper these conclusions. First, the comparison uses a single random train/validation split rather than k-fold cross-validation, so the reported numbers, especially for the smaller satin class (26 validation images), carry some sampling variance. Second, no data augmentation was applied to any model; given that augmentation is known to help exactly the kind of data-limited, texture-classification setting studied here [18], the custom CNN and ViT in particular were likely left short of their achievable ceiling. Third, the grayscale-then-replicate preprocessing step, while methodologically deliberate, discards genuine information for fabrics where weave and dye pattern interact — for example, certain honeycomb weaves are commercially distinguished partly by contrasting-color yarns — and a production system might reasonably keep color as a secondary cue rather than removing it entirely.

CONCLUSION AND FUTURE WORK

This paper compared three deep-learning strategies for classifying woven-fabric images into four weave types — honeycomb, plain, satin, and twill — under matched data, preprocessing, and training-budget conditions. A MobileNetV2 transfer-learning pipeline reached 95.99% validation accuracy and a 0.96 weighted F1-score, far ahead of a from-scratch custom CNN (52.51% accuracy) and a from-scratch, non-pretrained lightweight Vision Transformer (68.90% accuracy). The gap is best explained not by any inherent flaw in convolution or attention as mechanisms, but by the value of ImageNet pretraining when the target dataset is only in the low thousands of images, together with a preprocessing detail that likely disadvantaged the custom CNN specifically. Both weaker models showed a shared blind spot between the plain and twill classes, which persisted, in a much smaller form, even in the best-performing model.

Future work should extend this comparison in three directions: applying geometric and photometric data augmentation to test whether it narrows the gap between the pretrained and non-pretrained models; fine-tuning the later MobileNetV2 layers rather than freezing the entire backbone, to see whether weave-specific texture cues can be recovered without the overfitting risk of training the full network; and repeating the ViT branch with ImageNet-pretrained weights (e.g., ViT-B/16) rather than training the transformer from scratch, which the literature suggests is the single change most likely to close its remaining gap with MobileNetV2 [5]. A larger, multi-mill dataset with k-fold validation would also let these comparisons be reported with confidence intervals

rather than single-split point estimates, and would support extending the same three-architecture comparison to the fabric-defect-detection task that currently dominates the literature [6]–[8].

REFERENCES

1. A. Kumar, "Computer-vision-based fabric defect detection: A survey," IEEE Trans. Ind. Electron., vol. 55, no. 1, pp. 348–363, 2008.
2. C. Li, J. Li, Y. Li, L. He, X. Fu, and J. Chen, "Fabric defect detection in textile manufacturing: A survey of the state of the art," Security and Communication Networks, vol. 2021, pp. 1–13, 2021.
3. M. A. I. Hussain, B. Khan, Z.-J. Wang, and S. Ding, "Woven fabric pattern recognition and classification based on deep convolutional neural networks," Electronics, vol. 9, no. 6, art. 1048, Jun. 2020.
4. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998–6008.
5. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," in Proc. Int. Conf. Learning Representations (ICLR), 2021.
6. Y. Kahraman and A. Durmusoglu, "Deep learning-based fabric defect detection: A review," Textile Research Journal, vol. 93, no. 5–6, pp. 1485–1503, 2023.
7. M. Nasim, R. Mumtaz, M. Ahmad, and A. Ali, "Fabric defect detection in real world manufacturing using deep learning," Information, vol. 15, no. 8, art. 476, Aug. 2024.
8. S. Wang, H. Kong, B. Li, and F. Zheng, "Fab-ME: A vision state-space and attention-enhanced framework for fabric defect detection," arXiv preprint arXiv:2412.03200, 2024.
9. J. Jing, M. Xu, P. Li, Q. Li, and S. Liu, "Automatic classification of woven fabric structure based on texture feature and PNN," Fibers and Polymers, vol. 15, pp. 1092–1098, 2014.
10. B. Khan, F. Han, Z. Wang, and R. J. Masood, "Bio-inspired approach to invariant recognition and classification of fabric weave patterns and yarn color," Assembly Automation, vol. 36, no. 2, pp. 152–158, 2016.
11. N. Akram, R. A. Butt, and M. A. Qureshi, "Modification of a convolutional neural network for the weave pattern classification," Mehran Univ. Res. J. Eng. Technol., vol. 43, no. 2, pp. 79–90, Apr. 2024.
12. A. Raghuvardhan Babu and P. Narahari Sastry, "Fabric defect detection using modified MobileNetV2 deep learning model," Industrial Engineering Journal, vol. 53, no. 7, Jul. 2024.
13. S. Mei, Y. Shi, H. Gao, and L. Tang, "Research on fabric defect detection algorithm based on improved YOLOv8n algorithm," Electronics, vol. 13, no. 11, art. 2009, 2024.
14. B. Liu, H. Wang, Z. Cao, Y. Wang, L. Tao, J. Yang, and K. Zhang, "PRC-Light YOLO: An efficient lightweight model for fabric defect detection," Applied Sciences, vol. 14, no. 2, 2024.
15. H. Zhang, Y. Wu, S. Lu, L. Yao, and P. Li, "A mixed-attention-based multi-scale autoencoder algorithm for fabric defect detection," Coloration Technology, vol. 140, no. 3, pp. 451–466, 2023.
16. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510–4520.
17. K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, and D. Tao, "A survey on vision transformer," IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 1, pp. 87–110, 2023.
18. C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," Journal of Big Data, vol. 6, art. 60, 2019.
19. A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," Artificial Intelligence Review, vol. 53, pp. 5455–5516, 2020.
20. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
21. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems (NeurIPS), 2012, pp. 1097–1105.
22. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. Int. Conf. Learning Representations (ICLR), 2015.

23. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proc. Int. Conf. Machine Learning (ICML), 2015, pp. 448–456.
24. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. Int. Conf. Learning Representations (ICLR), 2015.
25. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248–255.
26. Pugazhenthi, T., & Chavan, P. P. (2026). Deep learning approaches for automatic recognition of textile weave structures. International Journal of Latest Technology in Engineering, Management & Applied Science (IJLTEMAS), 15(6), 788–794. https://doi.org/10.51583/IJLTEMAS.2026.150600060