Report on PatchInsight - Providing Insights into Patched vs. Non-Patched Clothing.

Y.S Nimesh

2024.11.16

Introduction

The goal of this project was to develop an image classification model capable of detecting whether an image contains a patched or non-patched clothing item. Using transfer learning techniques and advanced deep learning models, the project successfully identified the best-performing model for this task and evaluated its effectiveness.

Dataset

Classes: Patched and non-patched.

Image Count:

- Training: 200 images across 2 classes.
- Validation: 50 images across 2 classes.

Image Dimensions:

- Patched: 360x640 px.
- Non-Patched: 256x4096 px.

The dataset was preprocessed to normalize pixel values and resize all images to a consistent dimension of 360x640 px.

Model Selection

Three transfer learning models were evaluated:

1. EfficientNetB0
2. ResNet50
3. InceptionV3

Training Process

All models were trained for 10 epochs with frozen base layers.

Validation accuracy for each model:

- EfficientNetB0: 50.0%
- ResNet50: 94.0%
- InceptionV3: 100.0%

Based on the results, InceptionV3 was selected for further fine-tuning and hyperparameter optimization.

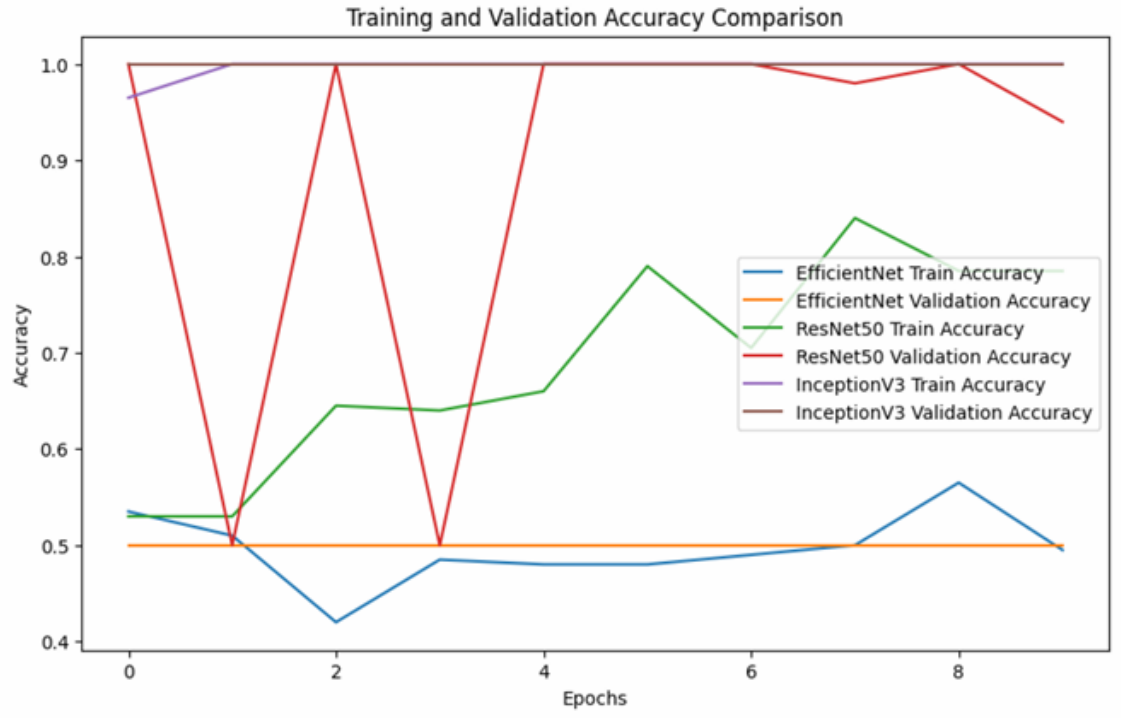


Figure 1 : Accuracy of model

Model Fine-Tuning

- The last 15 layers of the InceptionV3 base model were unfrozen for fine-tuning.
- The model was recompiled with a lower learning rate (`1e-5`) and trained for 5 additional epochs.
- Results after fine-tuning:
  - Validation Accuracy: 100.0%
  - Validation Loss: 0.0002

Hyperparameter Tuning

- Hyperparameters tuned:
  - Dense Layer Units: 32 to 256.
  - Dropout Rate: 0.1 to 0.5.
  - Learning Rate: 1e-4 to 1e-2.
- Tuning Results:
  - Best Hyperparameters:

- Dense Units: 32.

- Dropout Rate: 0.4.

- Learning Rate: 0.00049.

- Final Validation Accuracy: 100.0%

Best Value So Far	Hyperparameter
32	units
0.4	dropout
0.00049429	learning_rate
2	tuner/epochs
0	tuner/initial_epoch
2	tuner/bracket
0	tuner/round

Figure 2 : Best parameter values

Testing

The fine-tuned and optimized model was tested with unseen images. The following observations were recorded:

- Test Image 1: Non-Patched. Confidence: 88.6%.

- Test Image 2: Patched. Confidence: 98.5%.

Deployment

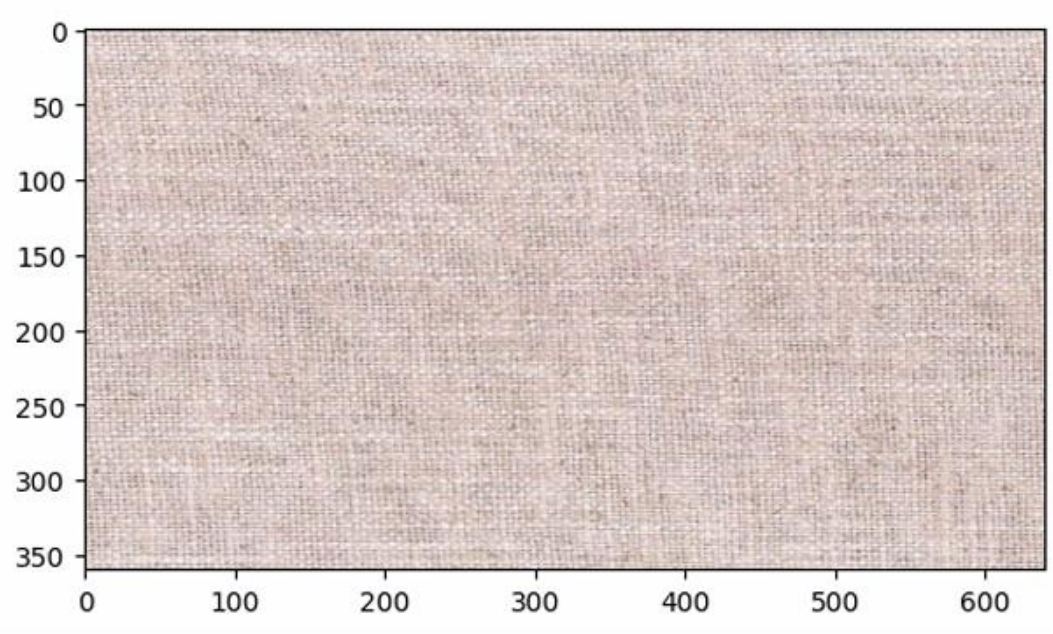
- The model was deployed as a Collab-based application where users can upload images for classification.

- Workflow:

1. Upload an image.
2. The model preprocesses the image and makes a prediction.
3. Outputs whether the image contains a patch or not, along with the confidence score.

One example :

Image :



Output:

```
Ian-Mankin-The-Forfar-and-Newbury-Collection-Newbury-Plain-Podwer-Fabric-FA15
0-218-Swatch-768x768.jpg is not Patch
```

Results

The project achieved:

- Model Selection: InceptionV3 with 100% accuracy after fine-tuning.

- Deployment: A functional application for real-time predictions.

- Optimization: Effective tuning of hyperparameters to enhance model performance.

Conclusion

This project demonstrates a robust workflow for classifying patched and non-patched clothing images using deep learning. The model's high accuracy and reliable deployment make it suitable for real-world applications in fashion, retail, and quality assurance.

References

- **Shanmugamani, R.** (2018). *Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras*. Packt Publishing.

- **Planche, B., & Andres, E.** (2019). *Hands-On Computer Vision with TensorFlow 2: Leverage deep learning to create powerful image processing apps with TensorFlow 2.x*. Packt Publishing.

- **Krizhevsky, A., Sutskever, I., & Hinton, G. E.** (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems, 25*, 1097–1105. https://doi.org/10.1145/3065386

- **Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z.** (2016). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818–2826. https://doi.org/10.1109/CVPR.2016.308

- TensorFlow. (n.d.). *Transfer learning and fine-tuning*. TensorFlow Official Documentation. Retrieved November 16, 2024, from https://www.tensorflow.org/tutorials/images/transfer_learning