Image classification with convolutional neural netowrks

Deep Learning (Methods) Project 1

Mikołaj Rzepiński & Krzysztof Wolny

Presentation Plan

Used Models

We used three different CNNs:

- **ResNet-50** – older, but still strong
- **EfficientNetV2** – faster and smaller, but still very good
- **ConvNeXt** – one of newest models which takes ideas from transformer models

Methodology Overview

- Dataset: CINIC-10 (10 classes, color images)
- TensorFlow library
- Google Colab (GPU)
- 10 epochs per experiment
- Pretrained weights from ImageNet

Default Configuration

- Learning Rate = 0.0005,
- Batch Size = 128,
- Dropout = 0.5,
- L1 Regularization = 0.0,
- Data Augmentation = none,
- No Few-shot Learning.

Hyperparameter Tuning

- Learning Rate - [0.0001, 0.0003, 0.0005, 0.001]
- Batch Size - [32, 64, 128]
- Dropout - [0.3, 0.5, 0.7]
- L1 Regularization - [0.0, 0.0001, 0.0005]

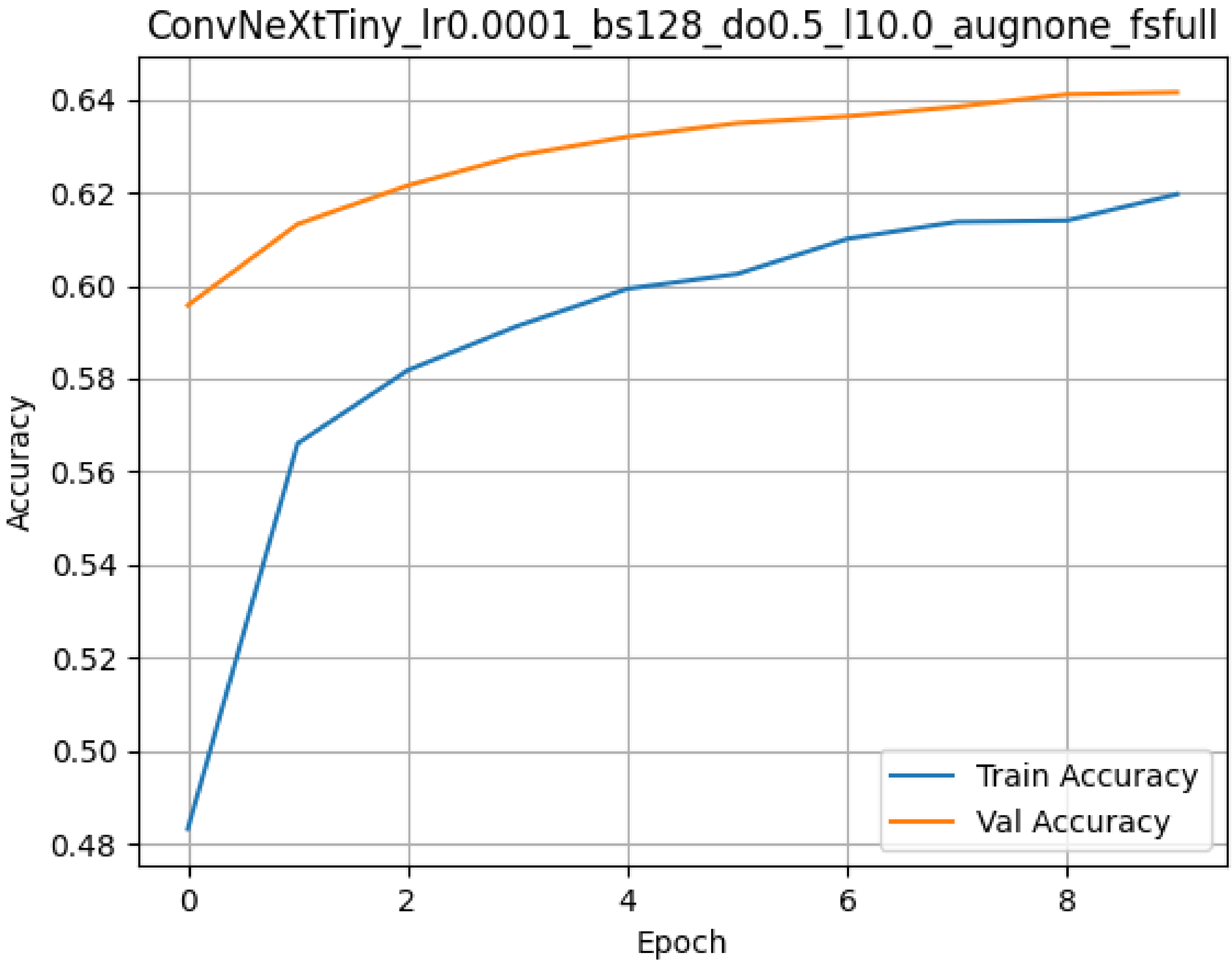
Only one hyperparameter changed at a time

Learning Rate

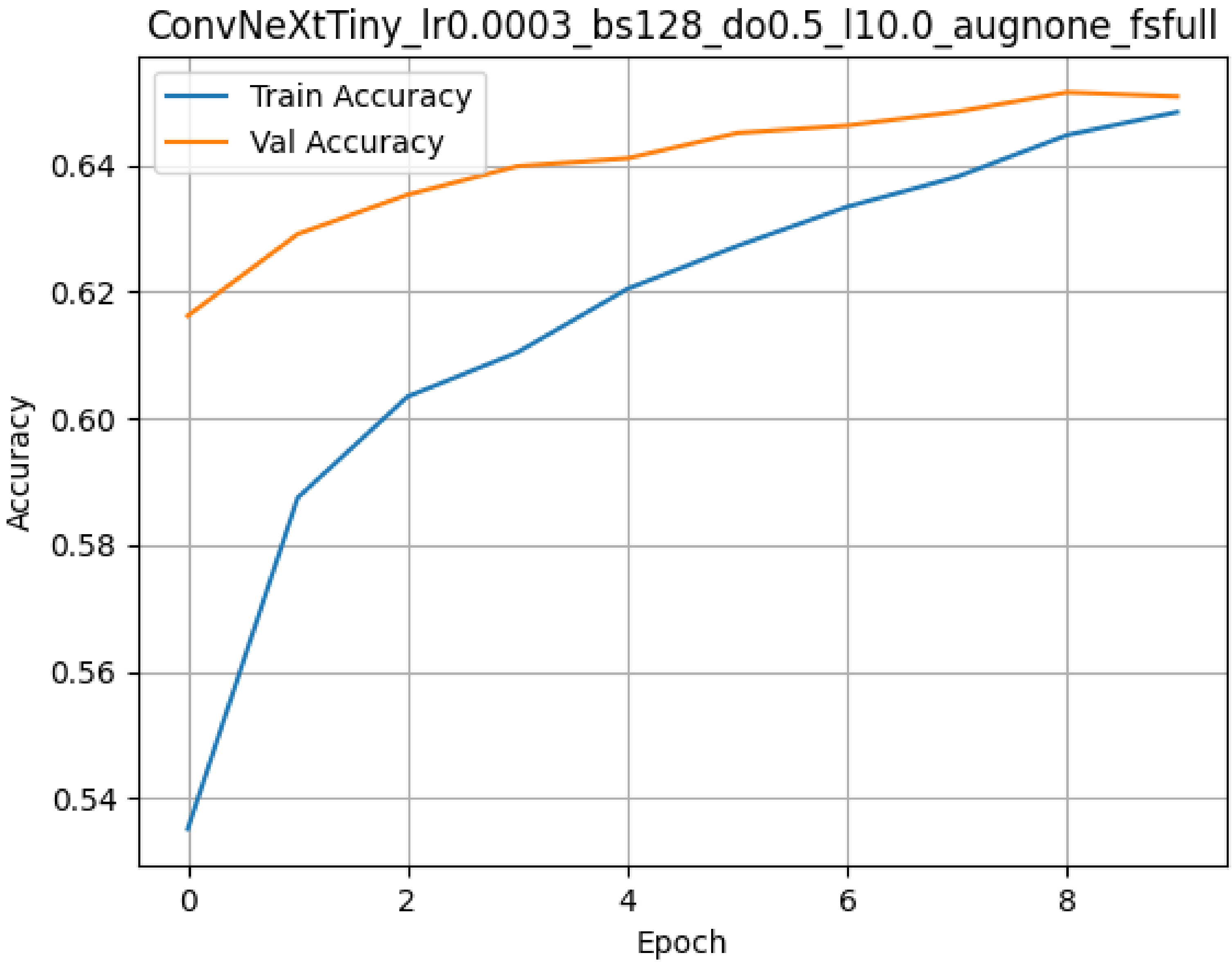
Learning Rate

- Higher learning rates resulted in better training accuracy, but validation accuracy became unstable for 0.001 rate.
- This aligns with the common observation that too large a learning rate leads to overshooting and poor generalization.

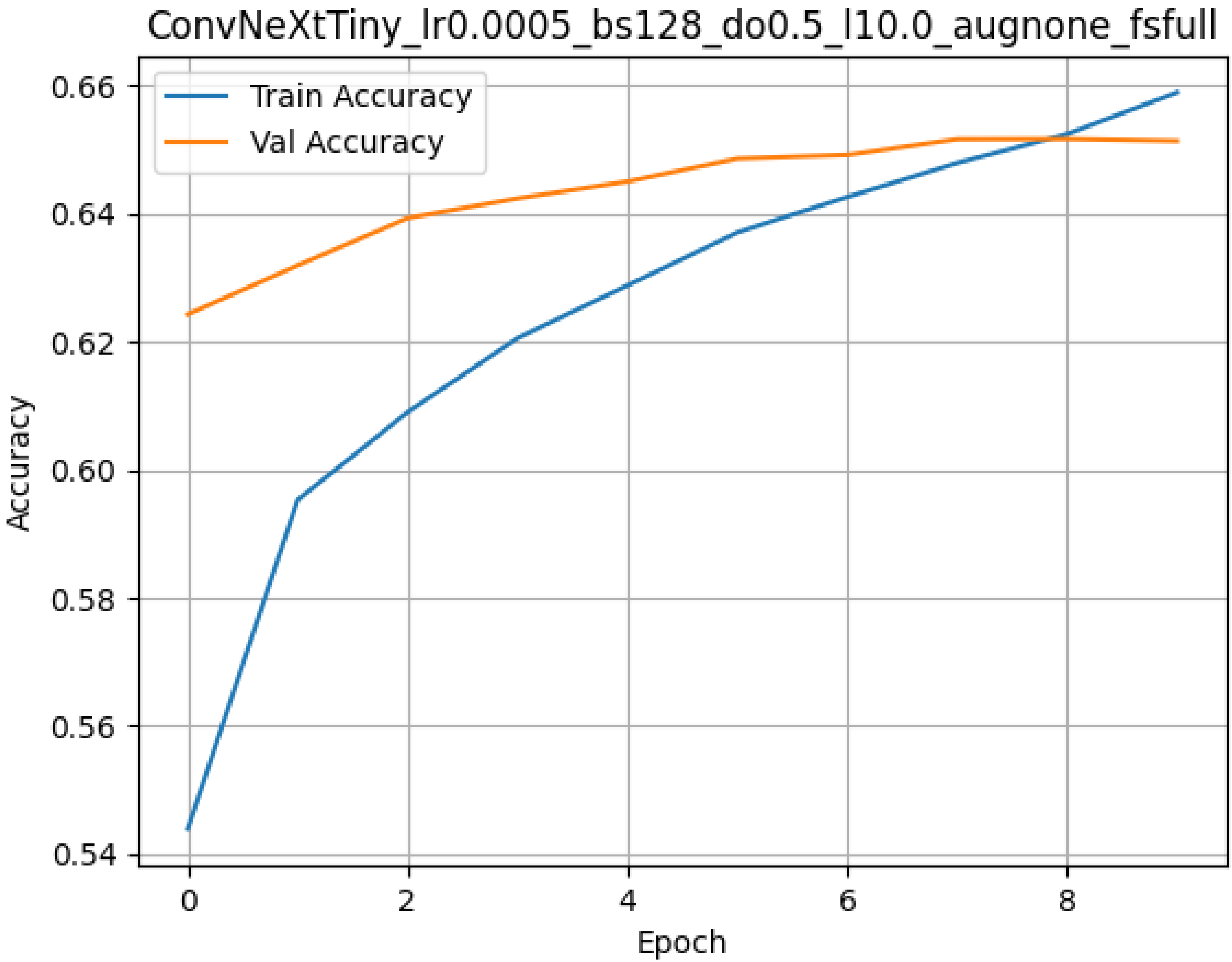
Learning Rate = 0.0001



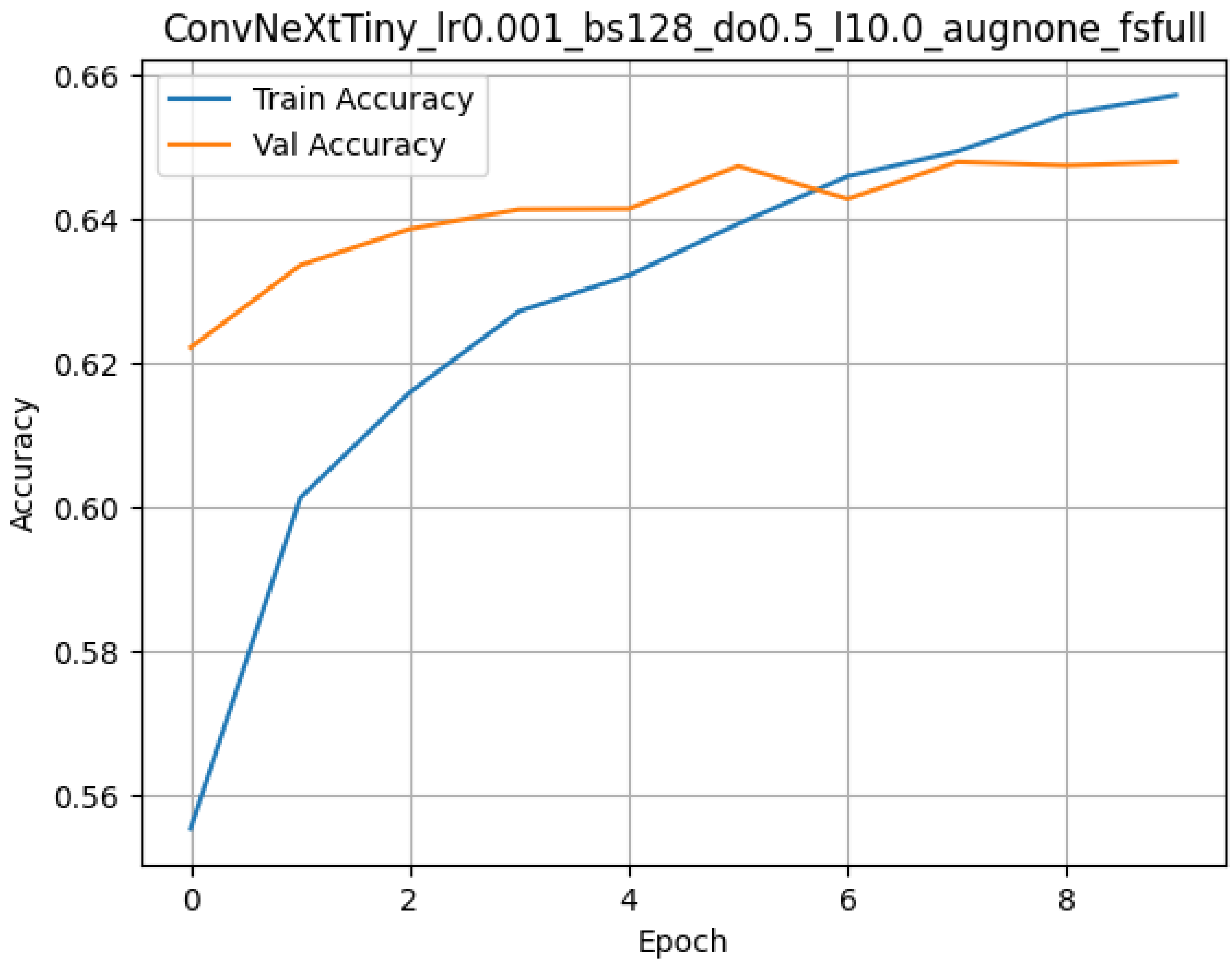
Learning Rate = 0.0003



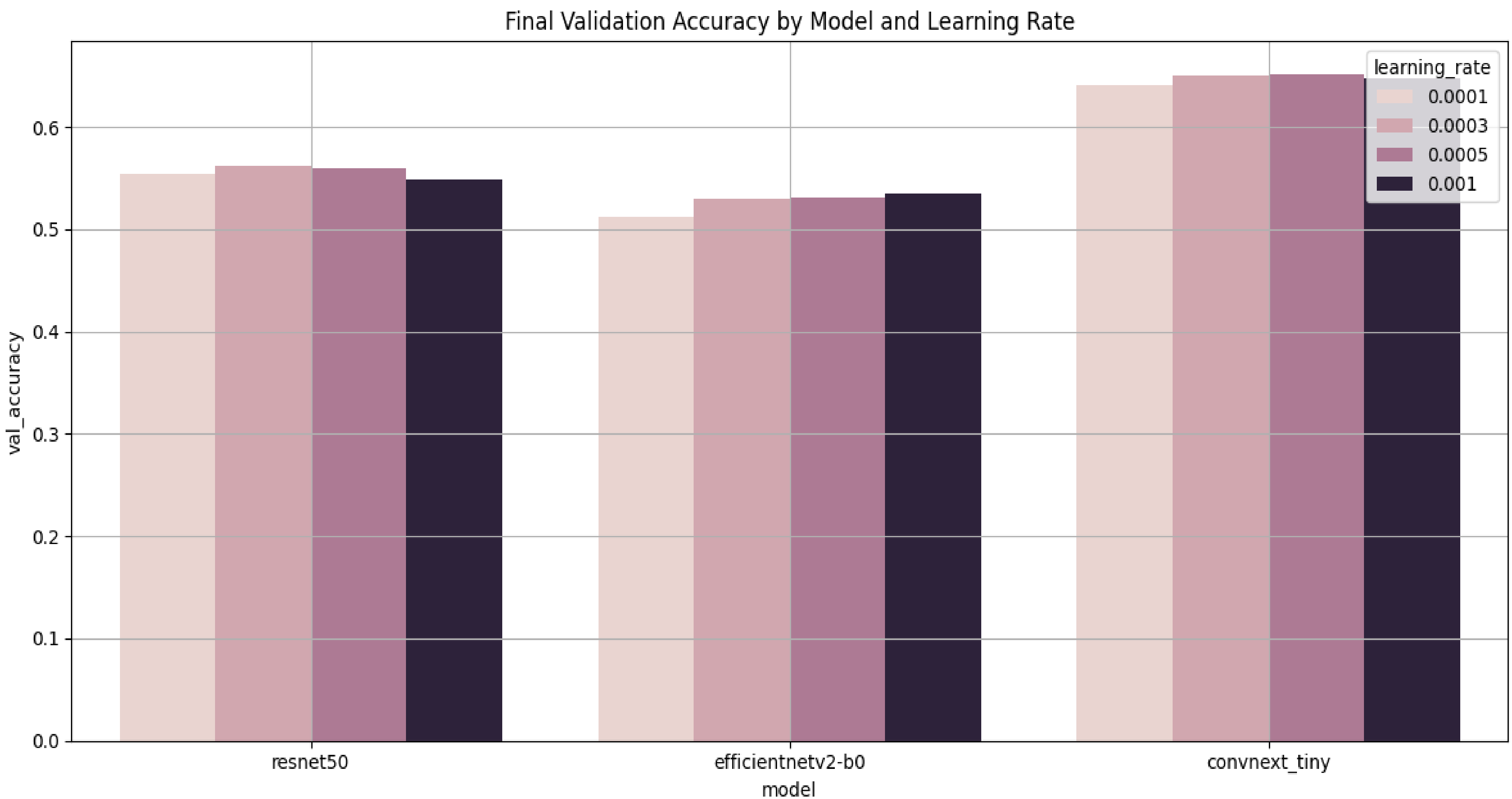
Learning Rate = 0.0005



Learning Rate = 0.001



Comparison chart for Learning Rate

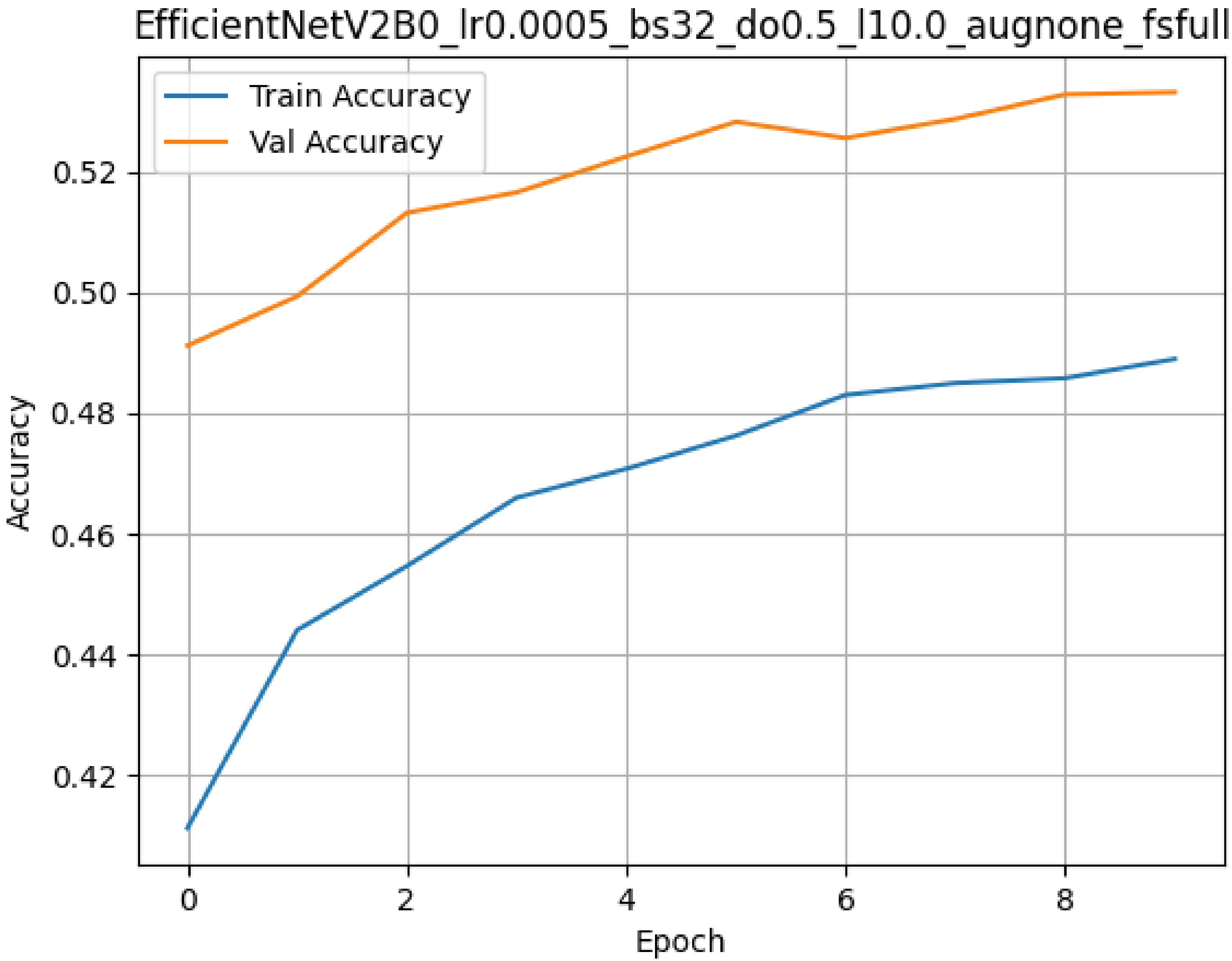


Batch Size

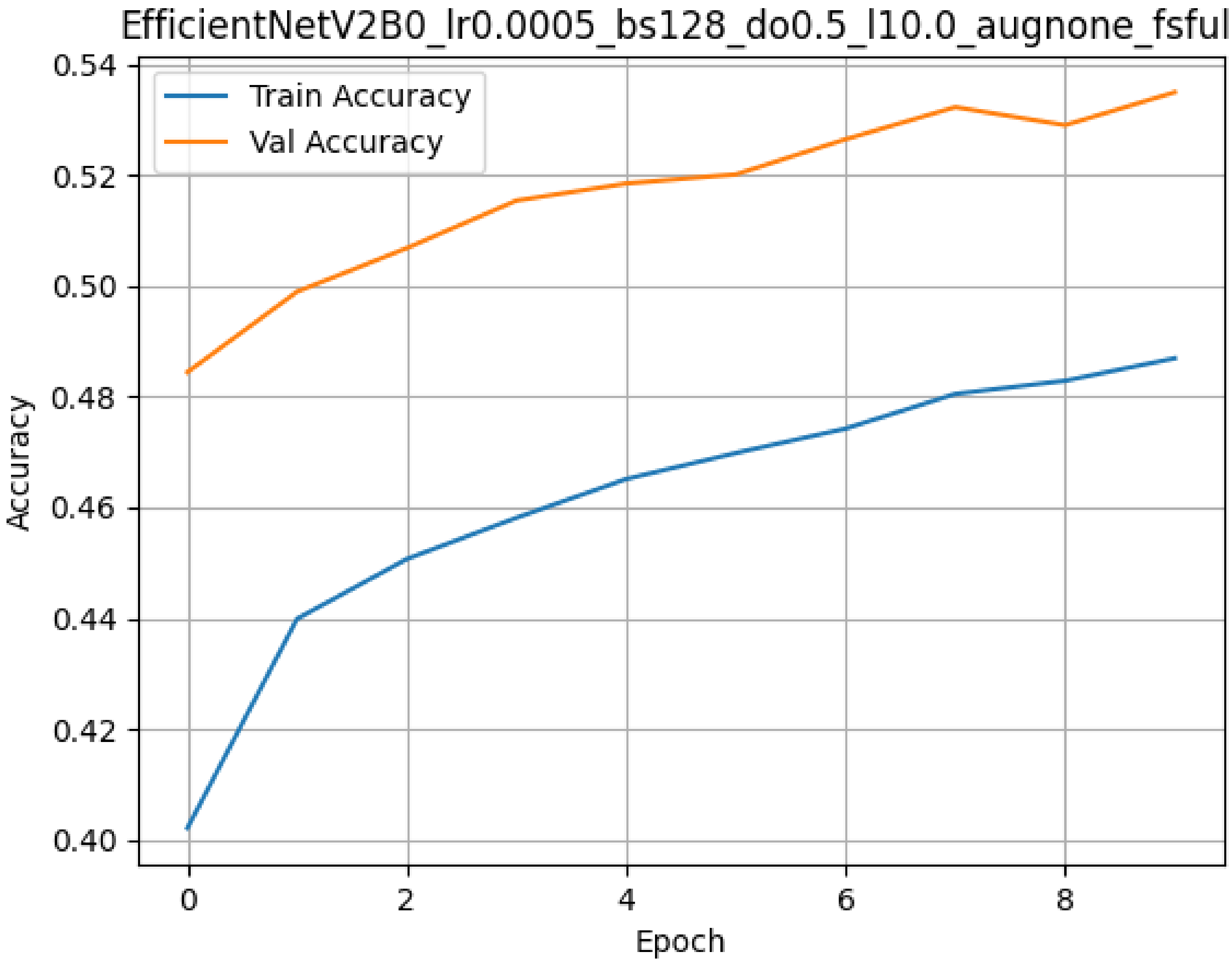
Batch Size

- As expected, the training and validation performance was relatively insensitive to batch size.
- In our setup, changing batch size mostly affected memory usage and iteration speed rather than final accuracy.

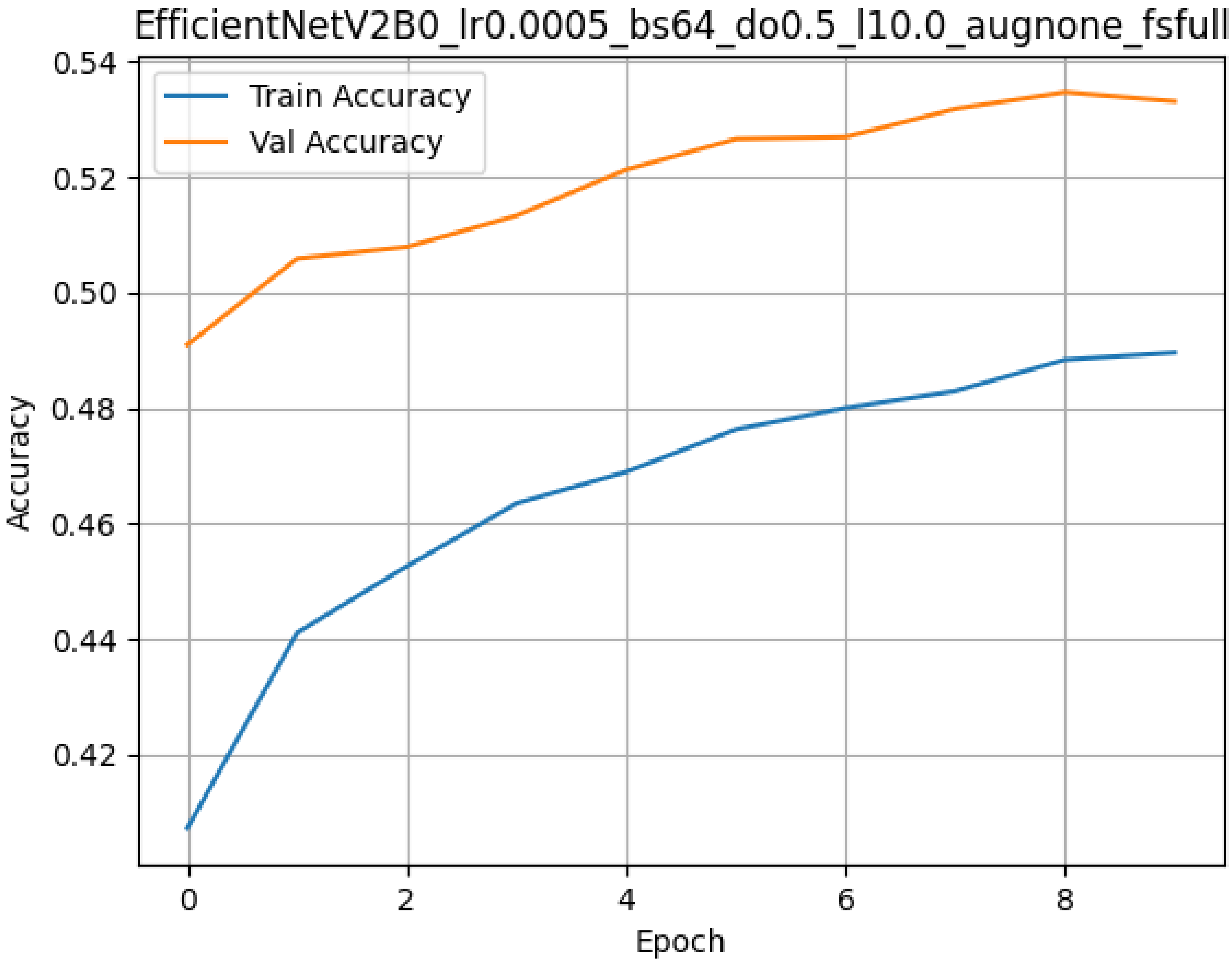
Batch Size = 32



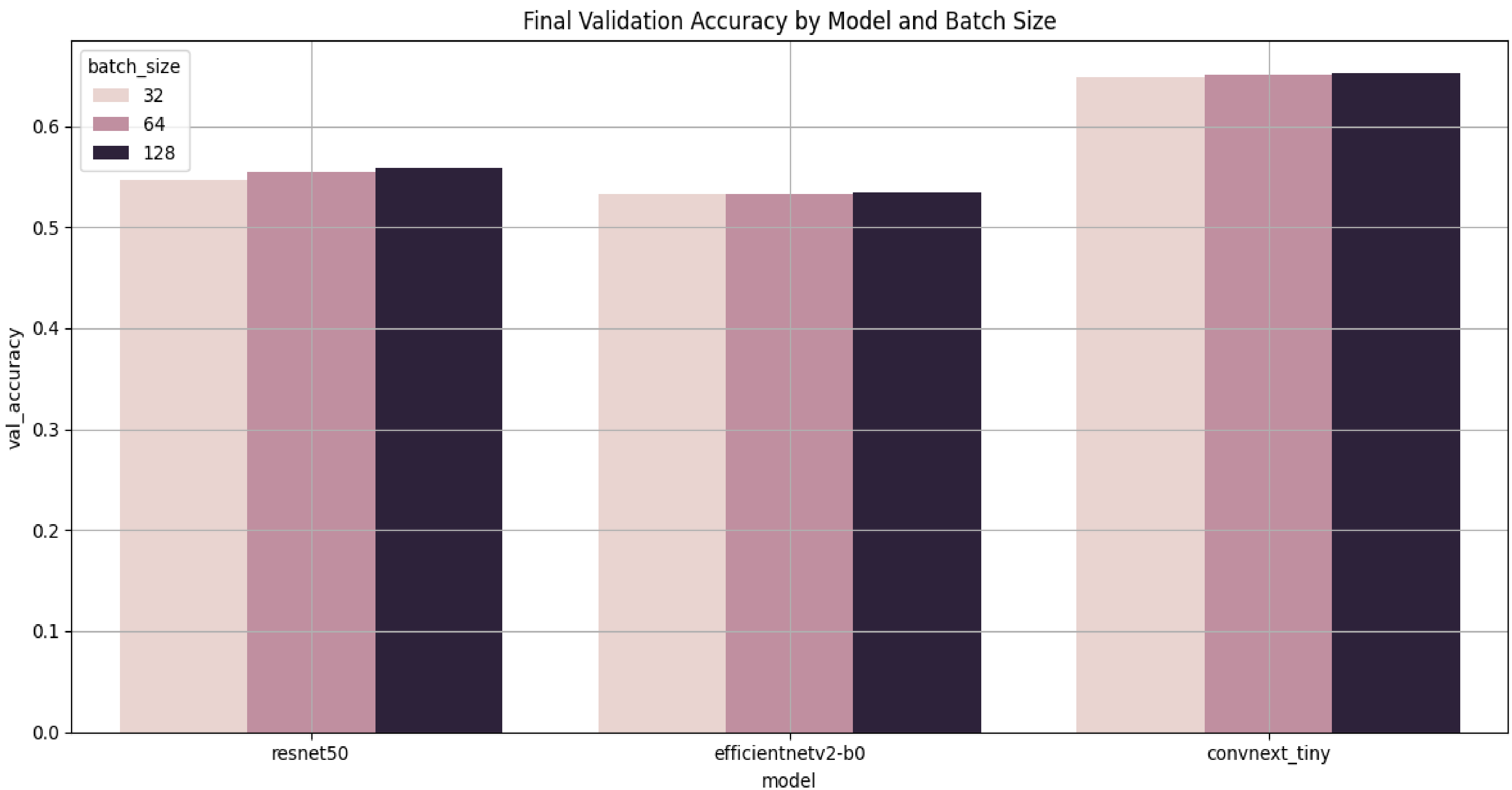
Batch Size = 64



Batch Size = 128



Comparison chart for Batch Size

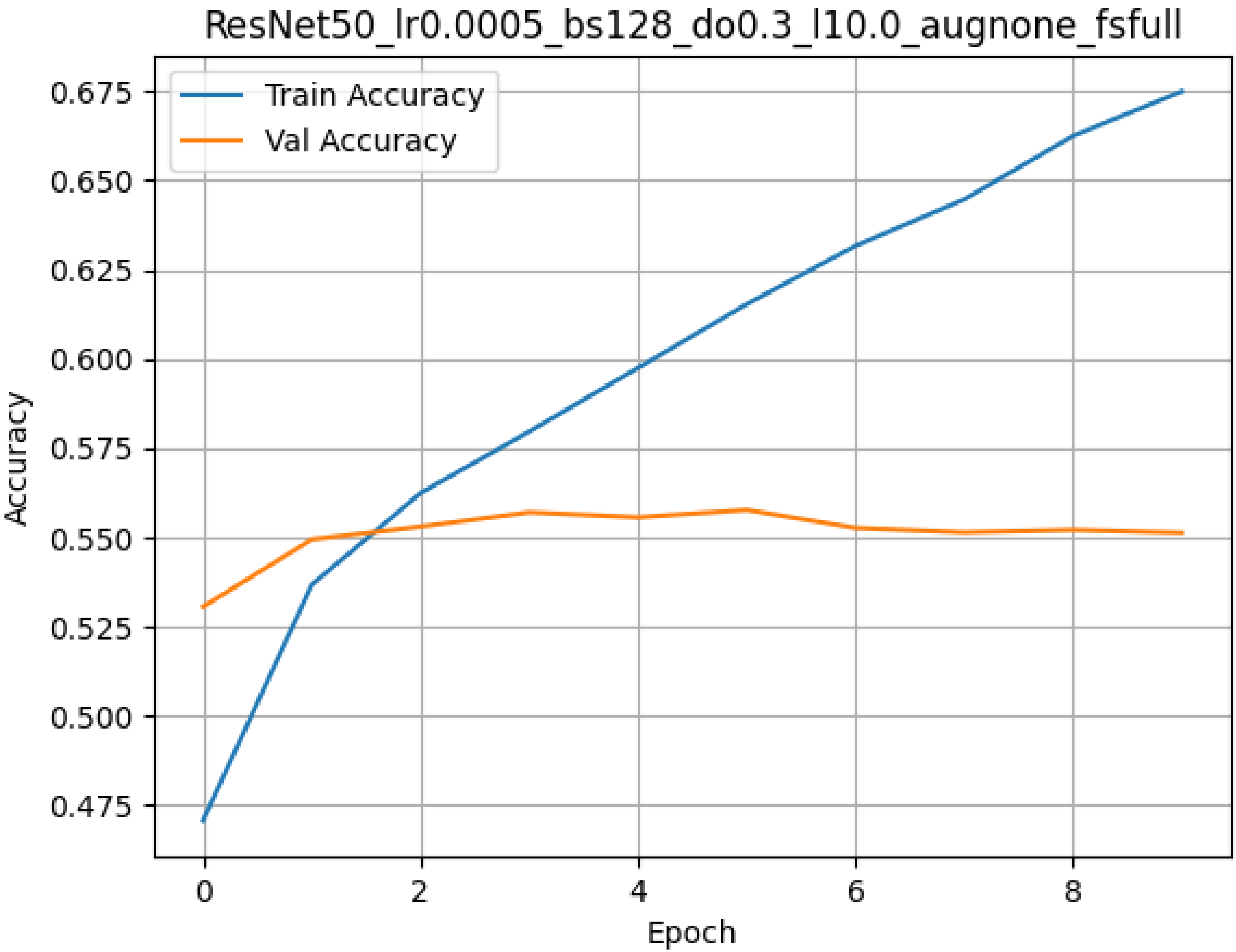


Dropout

Dropout

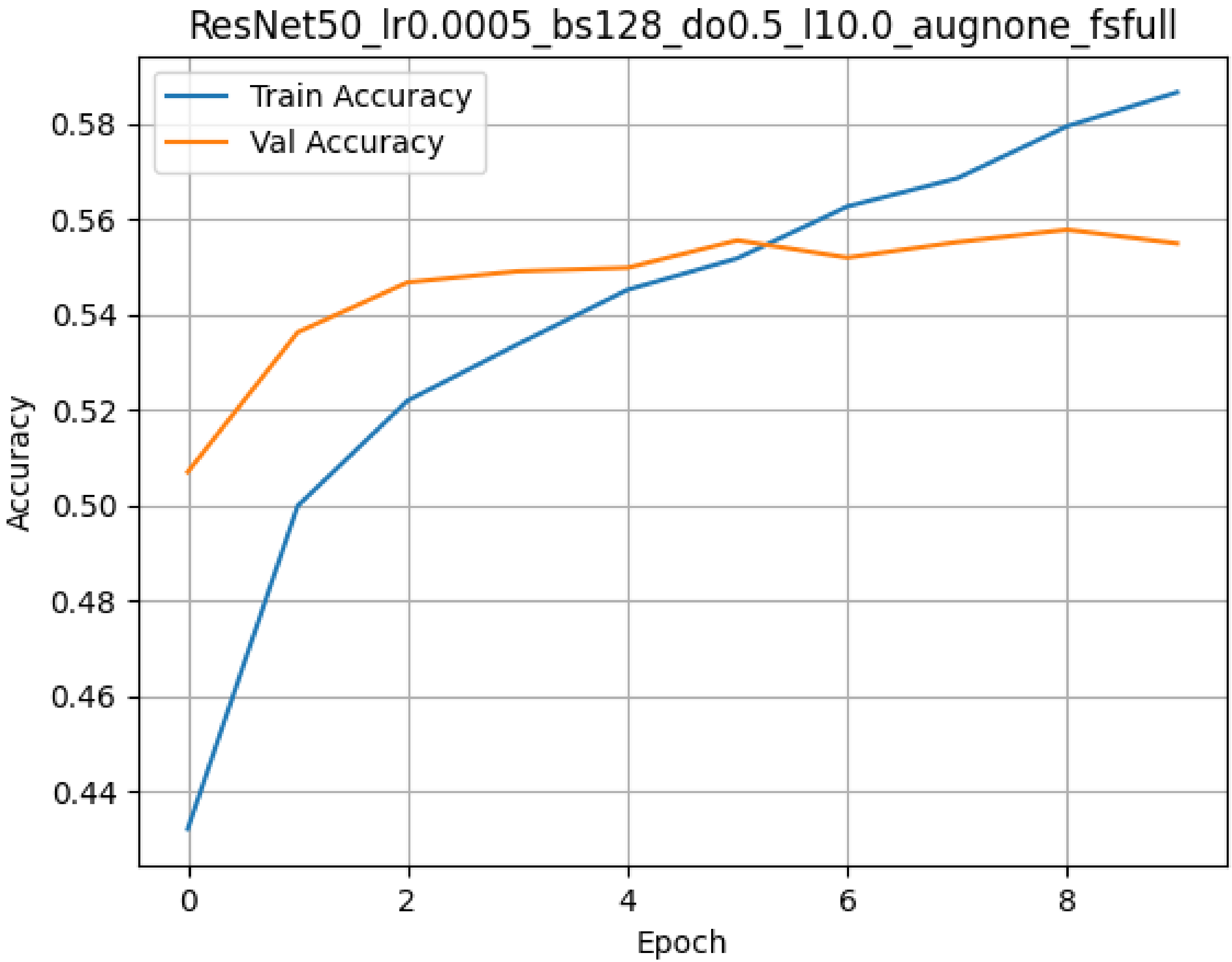
- A high dropout rate (0.7) caused clear underfitting, while 0.5 slightly underperformed compared to 0.3.
- Given the short training time, aggressive dropout was unnecessary and even made results worse.

Dropout = 0.3



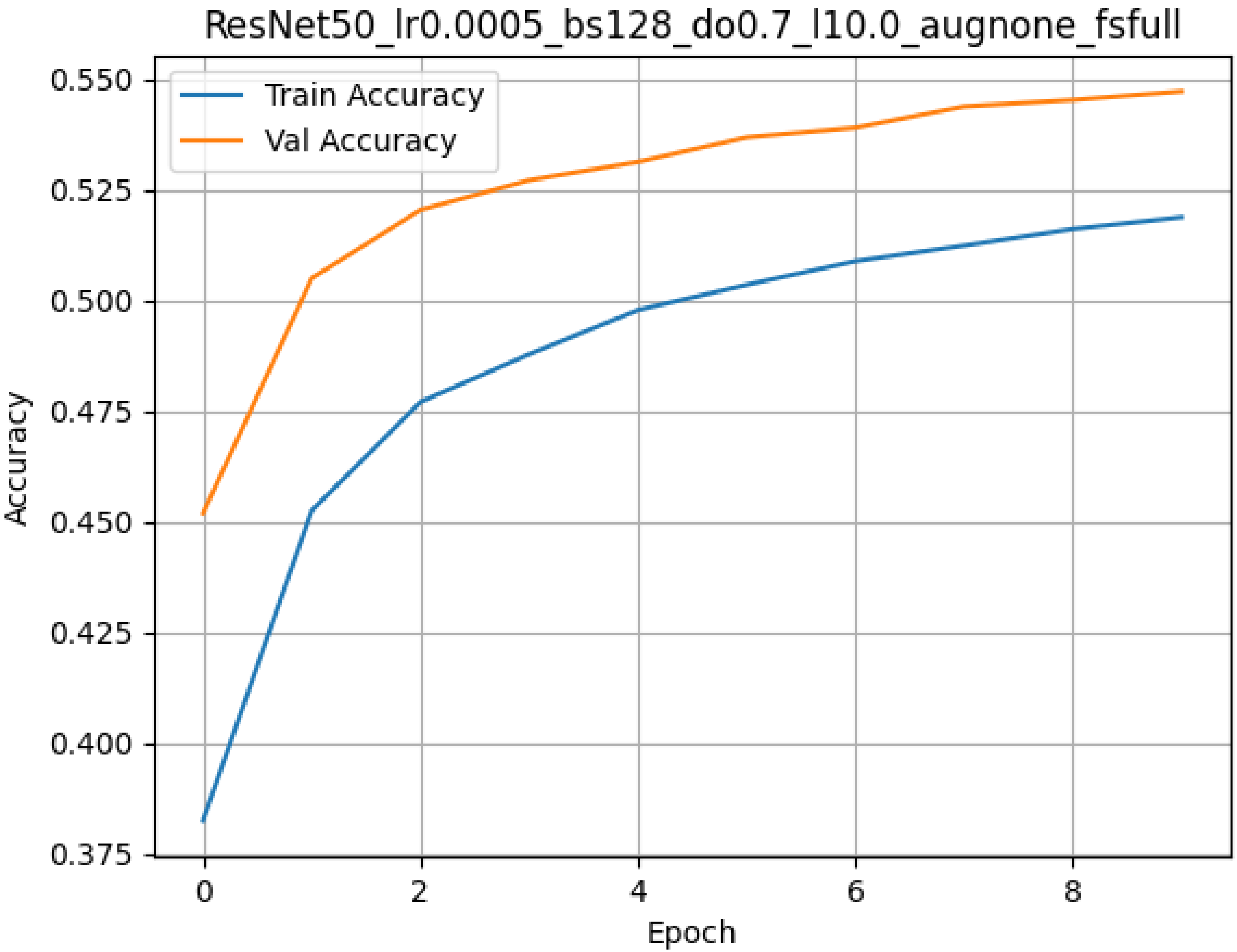
ResNet50_lr0.0005_bs128_do0.3_l10.0_augnone_fsfull
Train Accuracy
Val Accuracy
Accuracy
Epoch
0.675
0.650
0.625
0.600
0.575
0.550
0.525
0.500
0.475
0
2
4
6
8

Dropout = 0.5



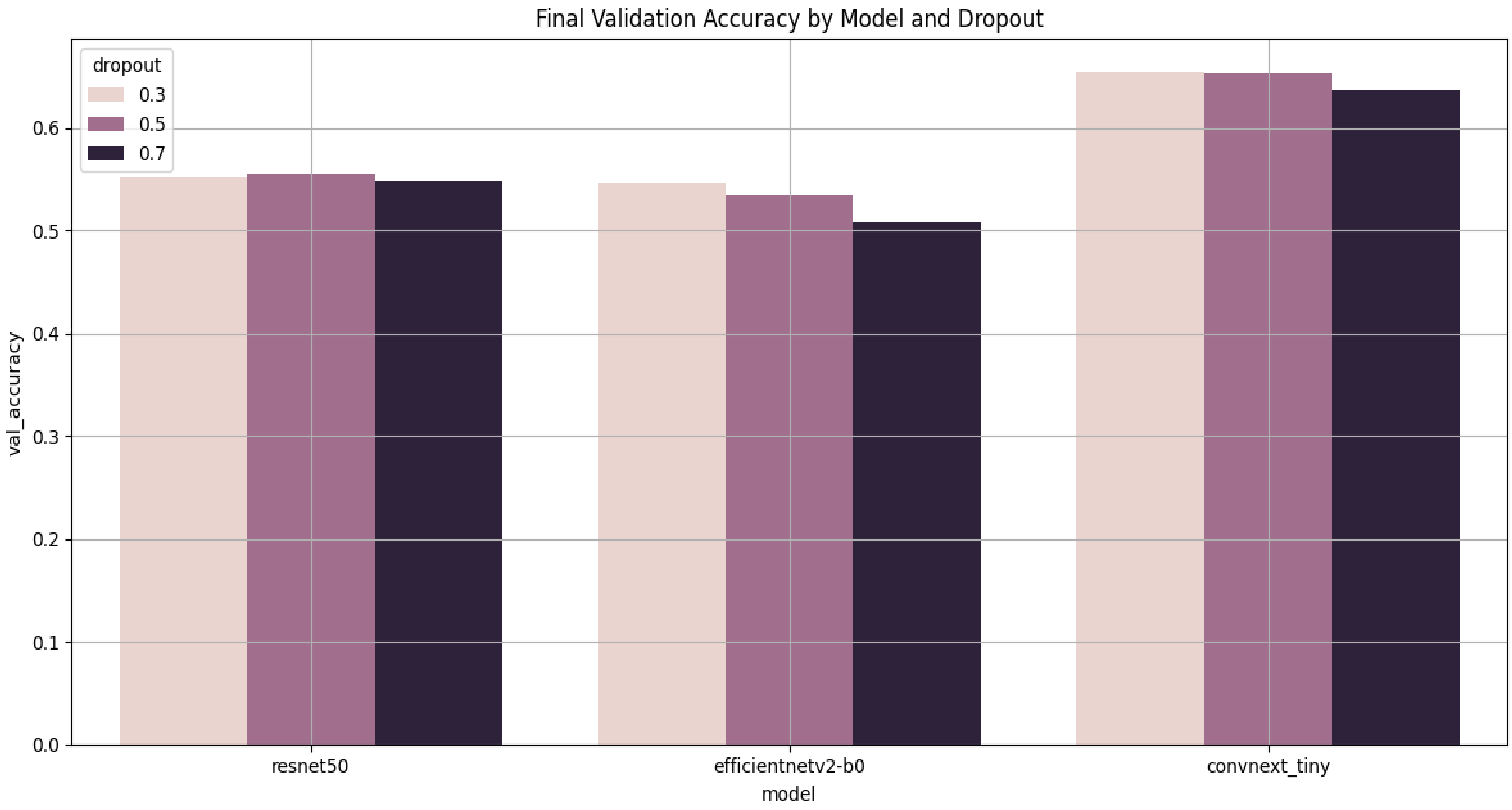
ResNet50_lr0.0005_bs128_do0.5_l10.0_augnone_fsfull
Train Accuracy
Val Accuracy
Accuracy
Epoch
0.58
0.56
0.54
0.52
0.50
0.48
0.46
0.44
0
2
4
6
8

Dropout = 0.7



ResNet50_lr0.0005_bs128_do0.7_l10.0_augnone_fsfull
Train Accuracy
Val Accuracy
Accuracy
Epoch
0.550
0.525
0.500
0.475
0.450
0.425
0.400
0.375
0
2
4
6
8

Comparison chart for Dropout

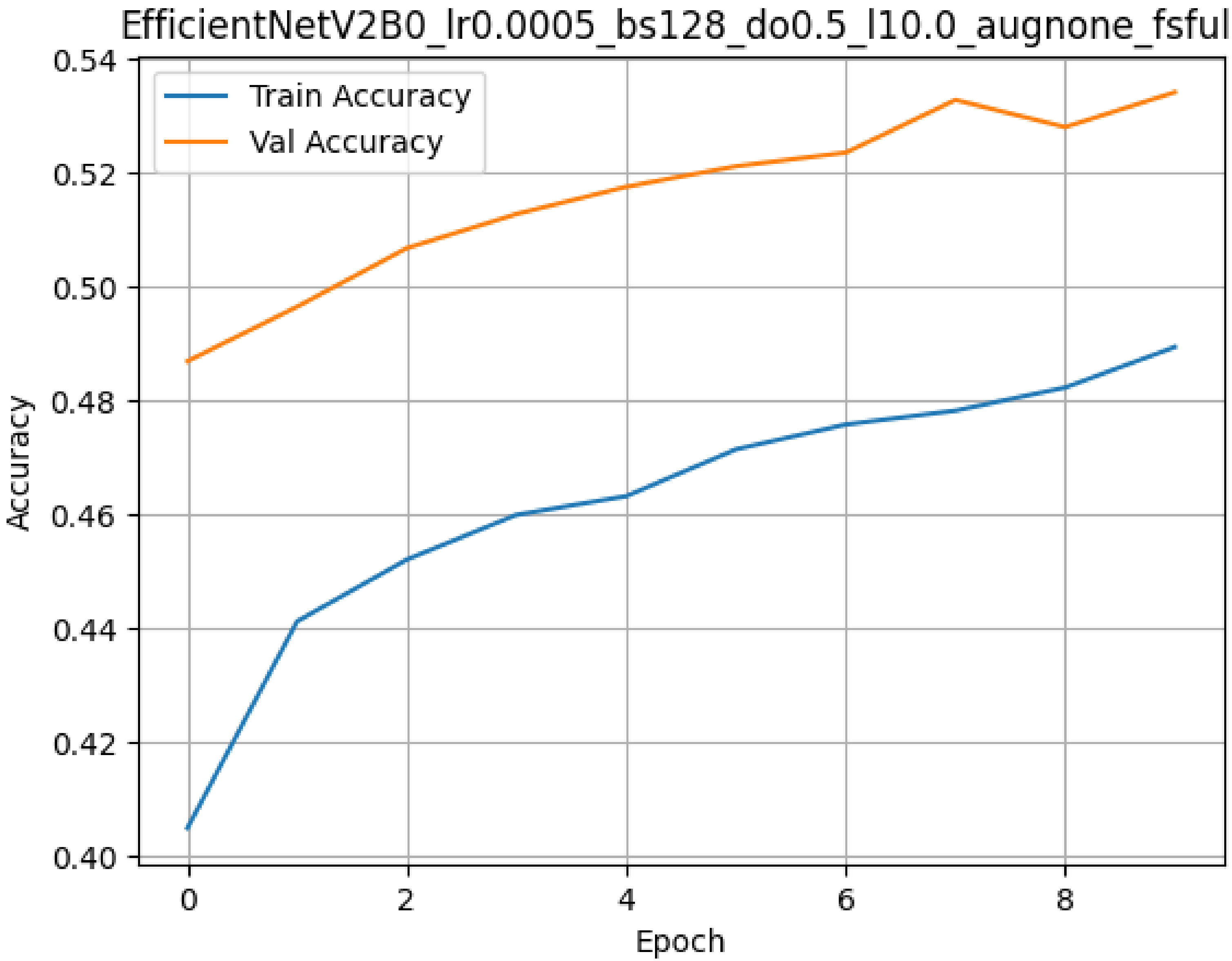


L1 Regularization

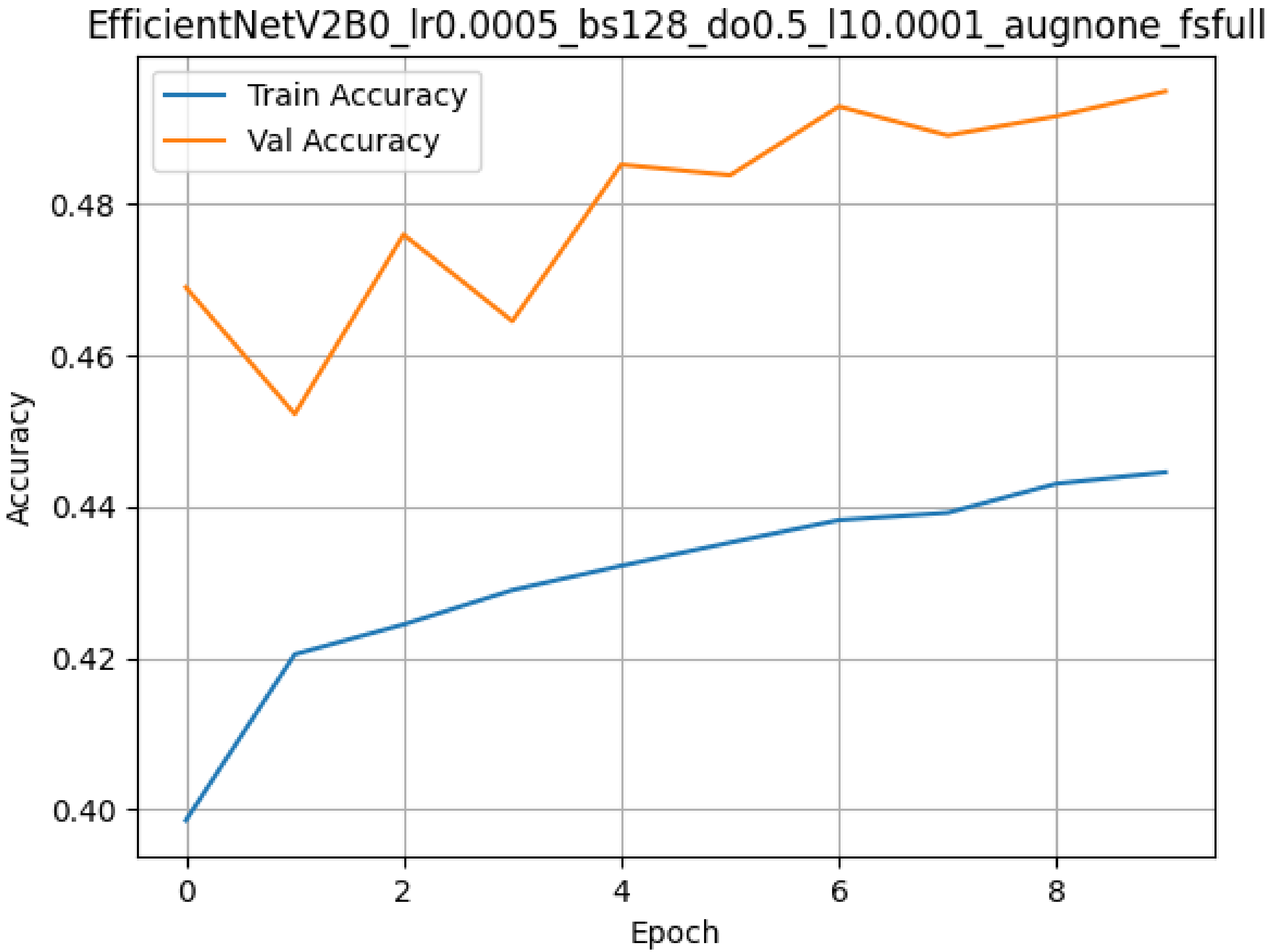
L1 Regularization

- Results indicate that increasing L1 penalty led to decreased accuracy, most notably with 0.0005.
- This means that with short training (10 epochs), L1 regularization can limit the model too much and stop it from learning the training data well.

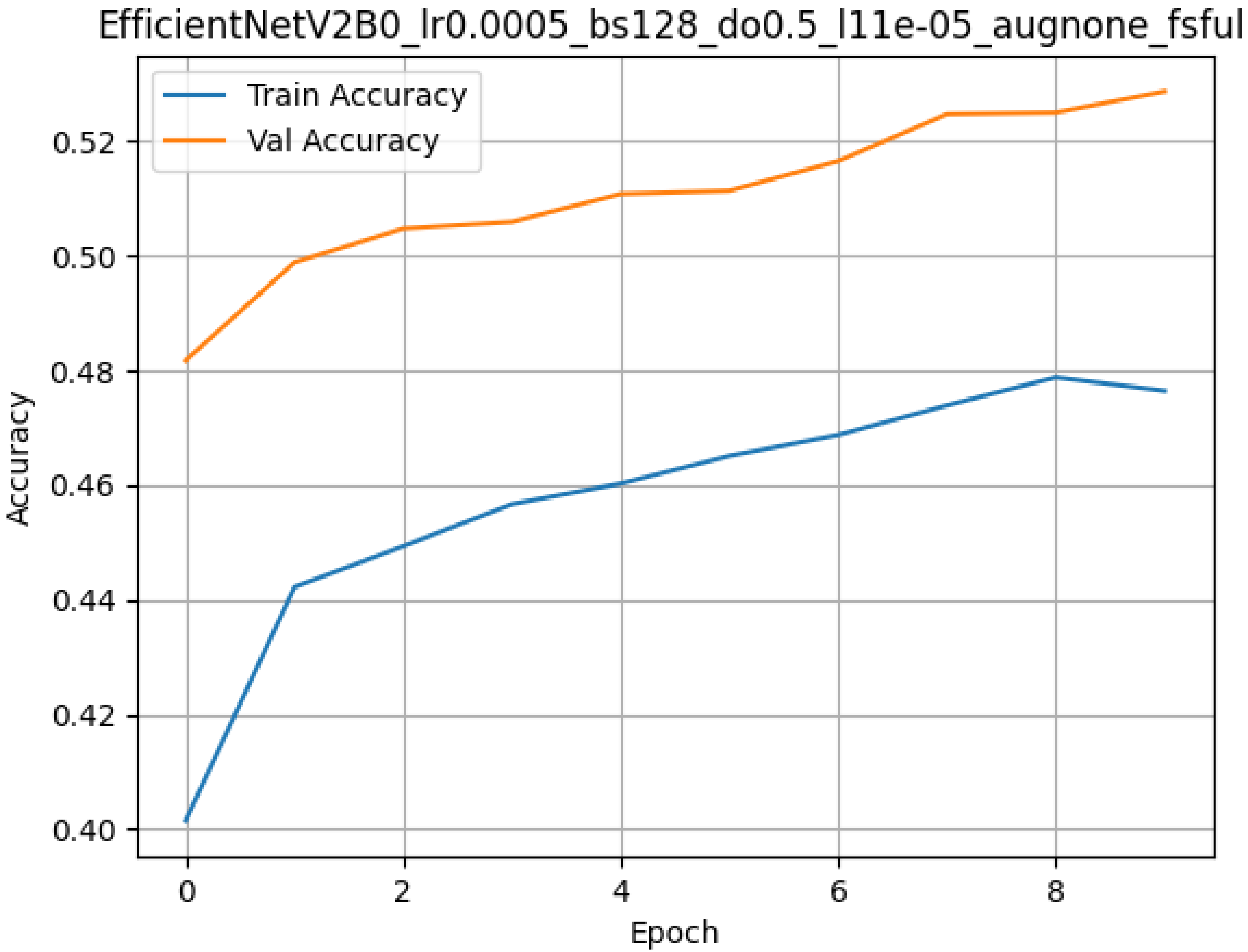
L1 Regularization = 0.0



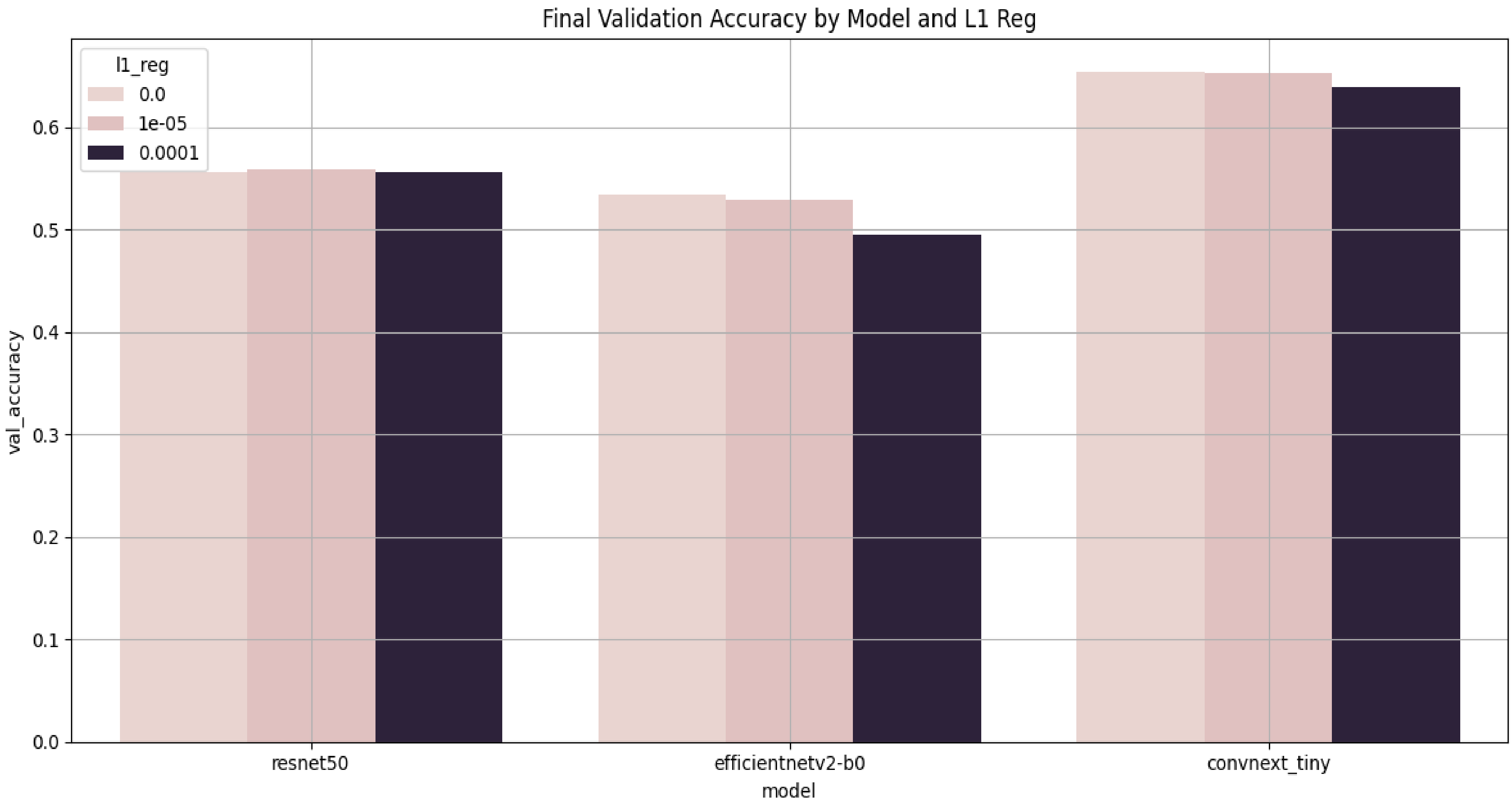
L1 Regularization = 0.0001



L1 Regularization = 0.0005



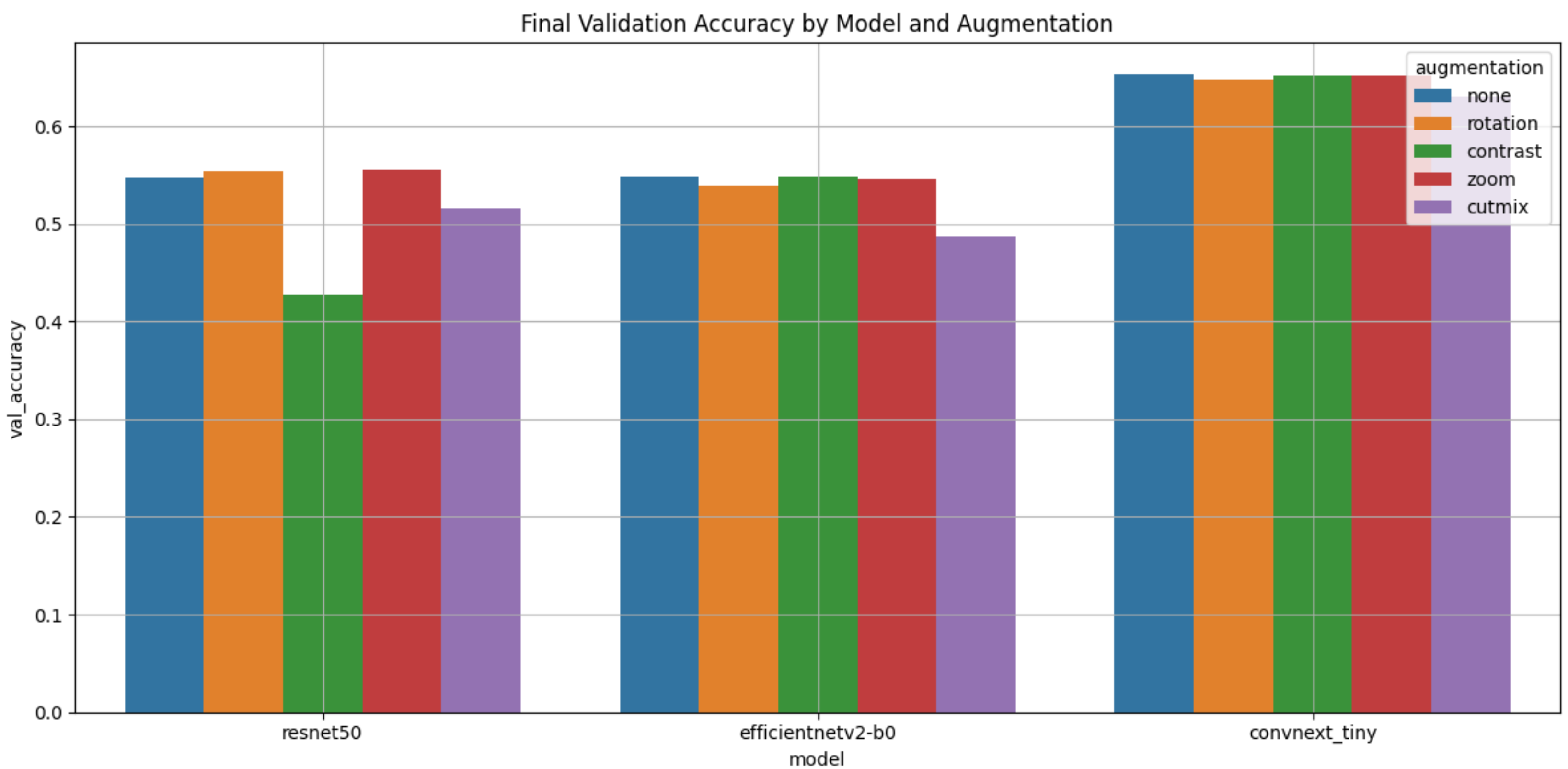
Comparison chart for L1 Regularization



Augmentation

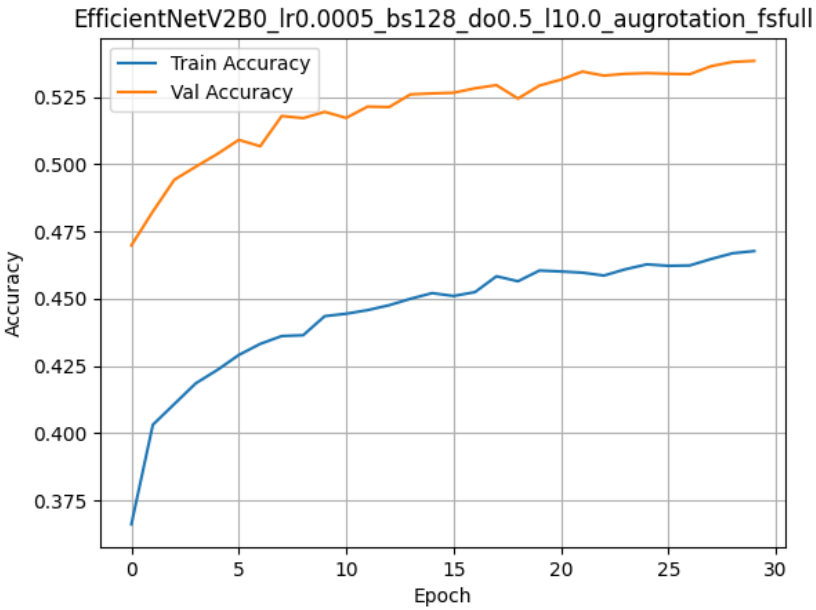
Augmentation

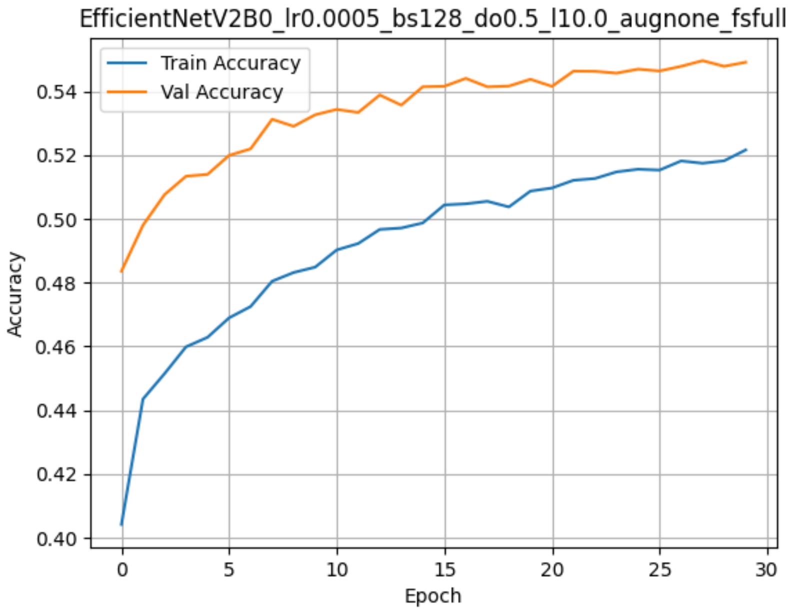
- Rotation
- Contrast
- Zoom
- Cutmix



Augmentation

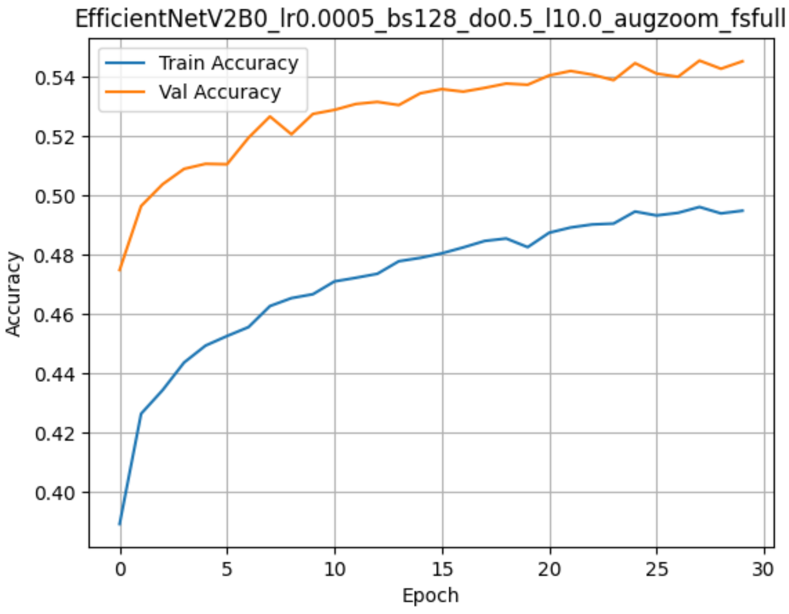
Rotation | Contrast

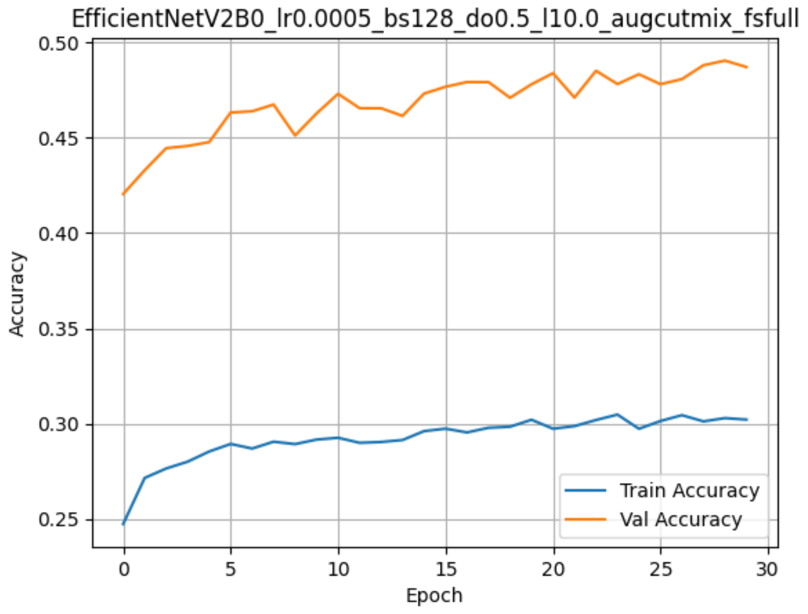




Augmentation

Zoom | Cutmix

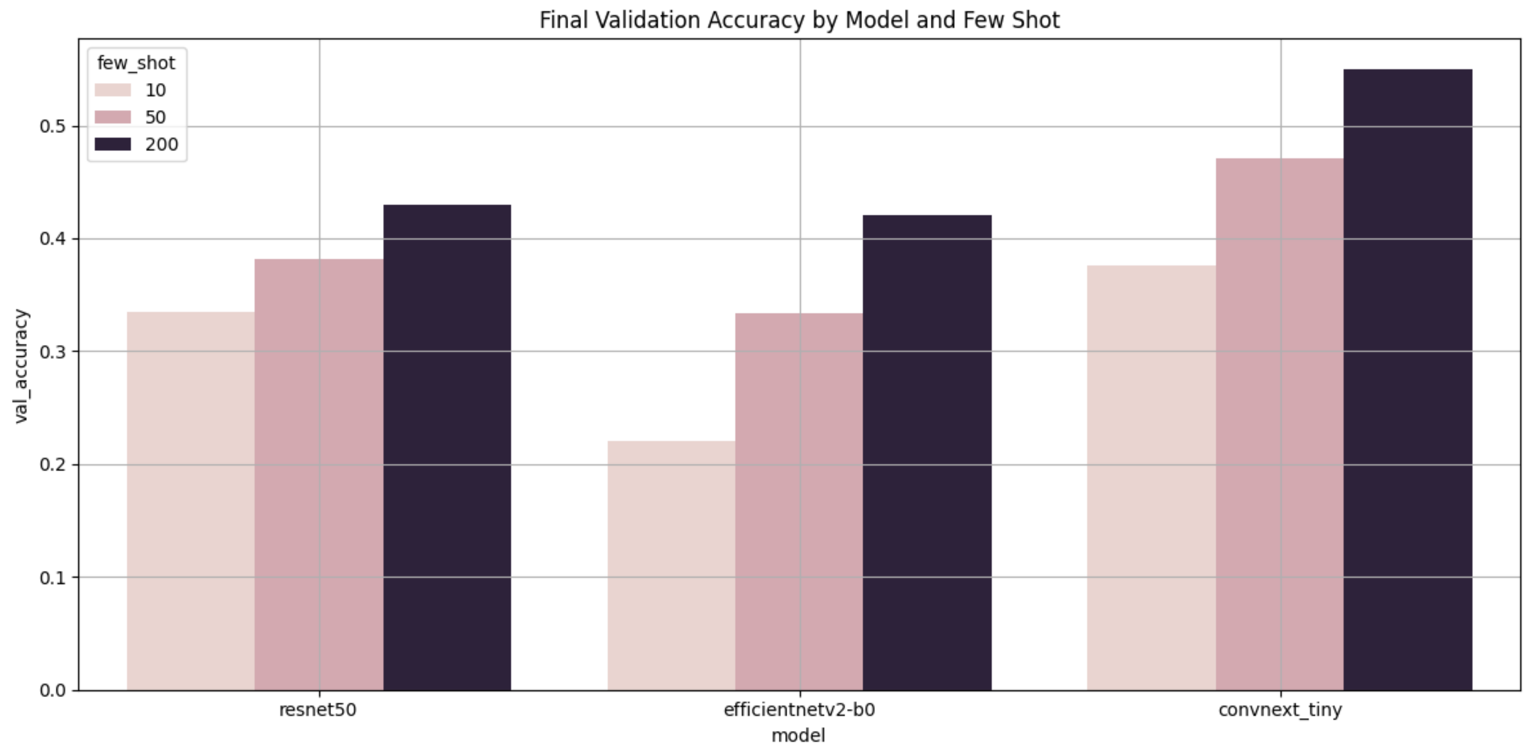




Few-shot learning

Few-shot Learning

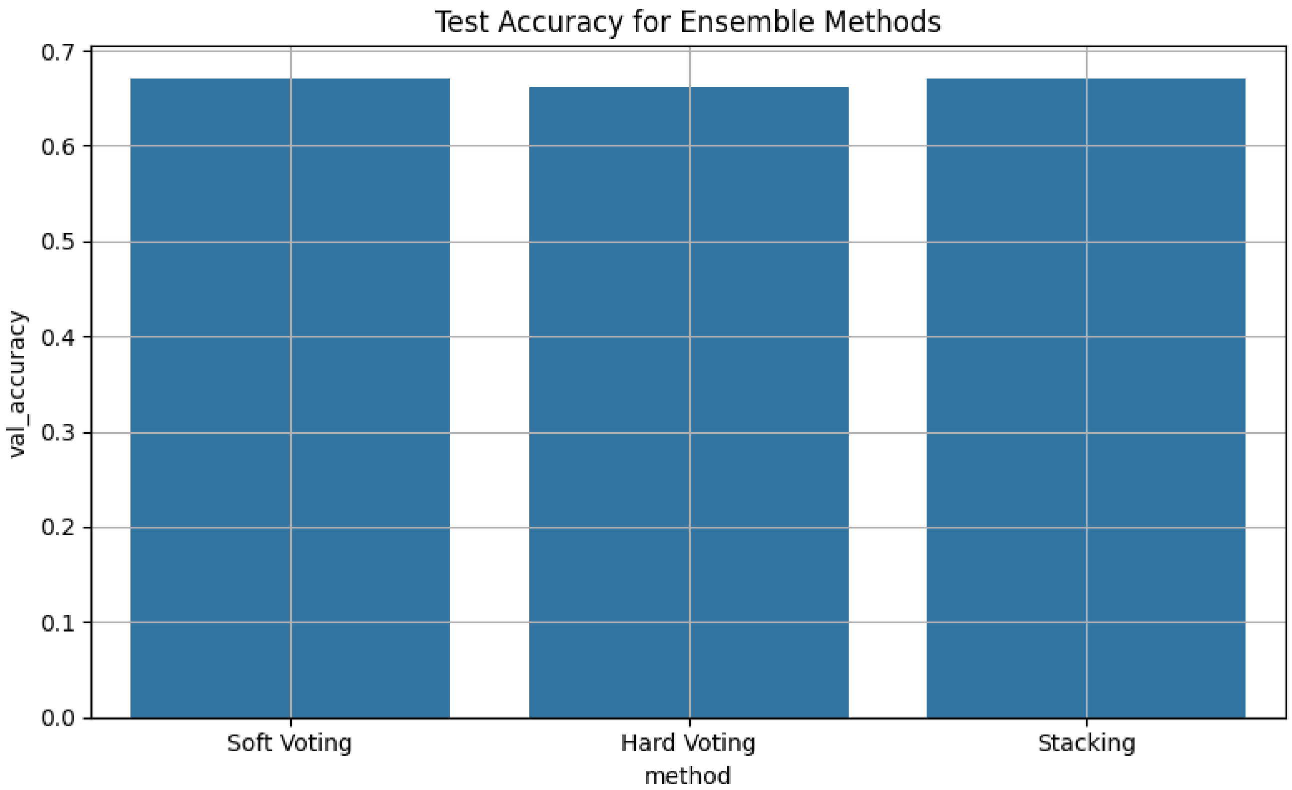
- Increasing the number of training images per class (from 10 to 50 to 200) improved model accuracy, with ConvNeXt showing significant gains, especially between 10 and 50 images.
- Using ImageNet weights performed well even with very limited data (10 images per class), demonstrating the power of transfer learning in few-shot learning scenarios.



Ensembling

Ensembling

- Hard-voting
- Soft-voting
- Stacking – Logistic Regression



Conclusions

Key Results & Observations

- ConvNext got best results out of 3 models we used
- Best Learning Rate: 0.003–0.005
- Batch Size had minor effect (best at 128)
- Lower Dropout (0.3) outperformed higher values
- L1 Regularization hurt short training performance
- CutMix underperformed in short training setup
- Few-shot learning worked even with 10 images/class
- Pretrained models helped generalization
- Best test accuracy: Stacking ensembling 67.85%

References

- A. F. Agarap, “Deep learning using rectified linear units (relu)”, CoRR, vol. abs/1803.08375, 2018. arXiv: 1803 . 08375. [Online]. Available: http : / / arxiv . org / abs / 1803 .08375.
- K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition”, CoRR, vol. abs/1512.03385, 2015. arXiv:1512 . 03385. [Online]. Available: http ://arxiv.org/abs/1512.03385.
- M. Tan and Q. V. Le, “Efficientnetv2: Smaller models and faster training”, CoRR, vol. abs/2104.00298, 2021. arXiv:2104.00298. [Online]. Available: https://arxiv.org/abs/2104.00298.
- N. Srivastava et al., “Dropout: A simple way to prevent neural networks from overfitting”, Journal of Machine Learning Research, 2014, https://jmlr.org/papers/v15/srivastava14a.html.
- C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning”, Journal of Big Data, vol. 6, no.1, pp. 1–48, 2019. [Online]. Available: https://arxiv.org/abs/1901.11196.
- S. Yun et al., “Cutmix: Regularization strategy to train strong classifiers with localizable features”, in ICCV, https://arxiv.org/abs/1905.04899, 2019.

Thank you!