AI Engineering Project

Final Presentation

Group 4 (Jungo, Luegmayer, Müllner & Schweitzer)

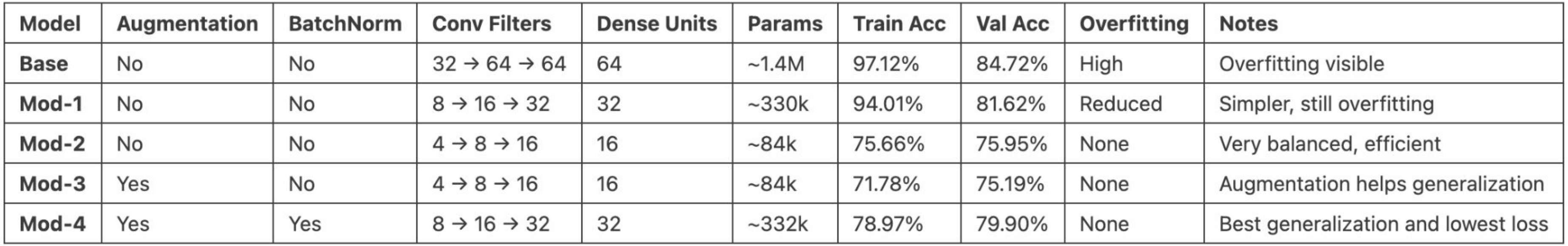
Project - Cats vs Dogs

Dataset

- TensorFlow Dataset - (cats_vs_dogs)
    - **25.000** Pictures in the dataset
    - **23.262** Valid pictures
    - **1.738** Corrupted images that are dropped

No pre trained Model

No pre trained Model (5)

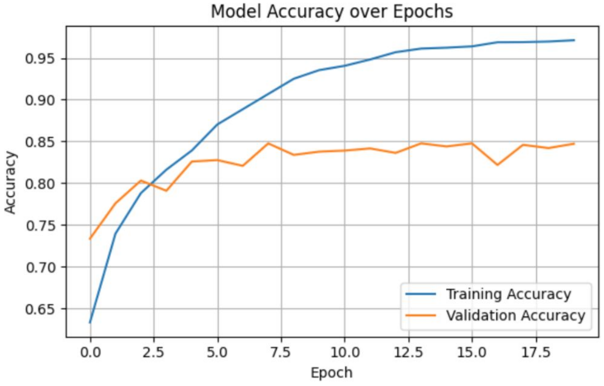


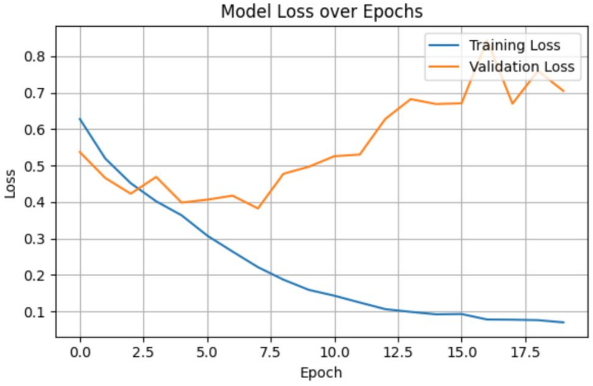
Model	Augmentation	BatchNorm	Conv Filters	Dense Units	Params	Train Acc	Val Acc	Overfitting	Notes
Base	No	No	32 → 64 → 64	64	~1.4M	97.12%	84.72%	High	Overfitting visible
Mod-1	No	No	8 → 16 → 32	32	~330k	94.01%	81.62%	Reduced	Simpler, still overfitting
Mod-2	No	No	4 → 8 → 16	16	~84k	75.66%	75.95%	None	Very balanced, efficient
Mod-3	Yes	No	4 → 8 → 16	16	~84k	71.78%	75.19%	None	Augmentation helps generalization
Mod-4	Yes	Yes	8 → 16 → 32	32	~332k	78.97%	79.90%	None	Best generalization and lowest loss

No pre trained Model - Result

Base

First Model

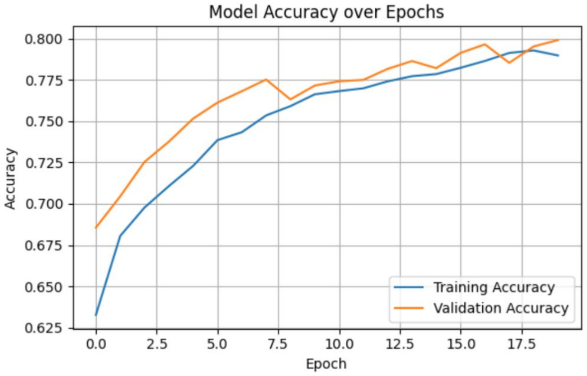

Model Accuracy over Epochs

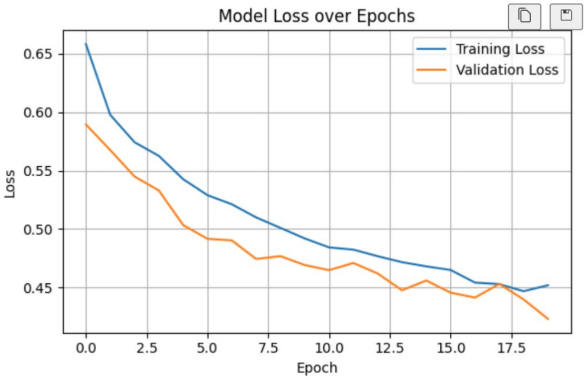

Model Loss over Epochs

```
Final Training Metrics:
Training Accuracy: 0.9712
Validation Accuracy: 0.8472
Training Loss: 0.0703
Validation Loss: 0.7046
```

Modified 4

Last Model

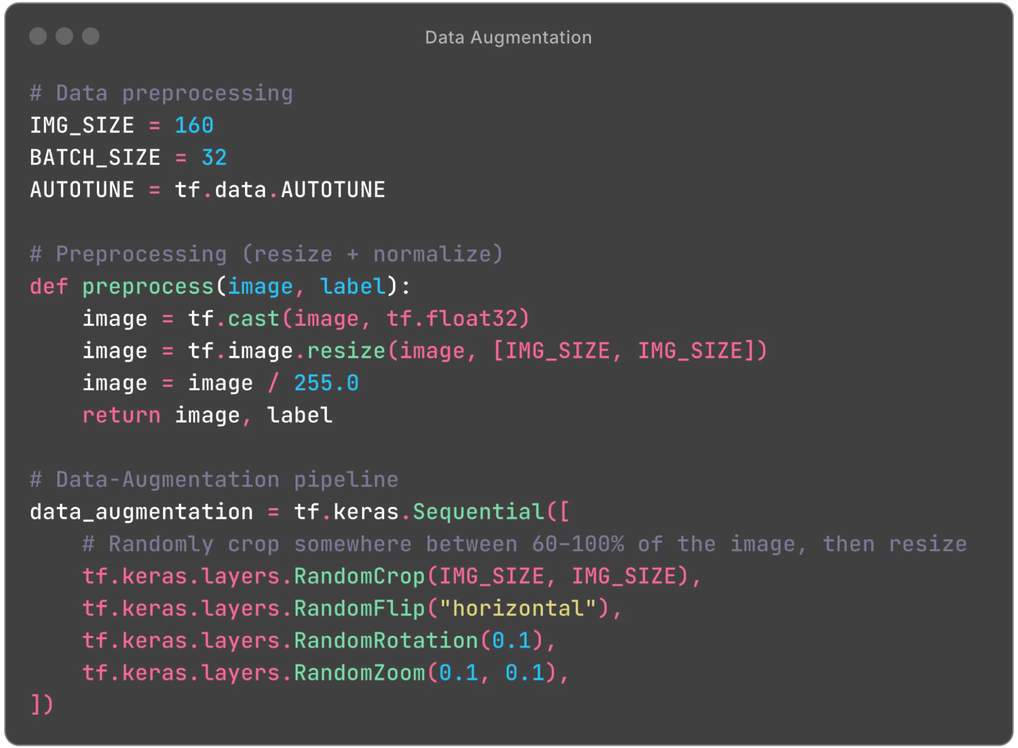

Model Accuracy over Epochs


Model Loss over Epochs

```
Final Training Metrics:
Training Accuracy: 0.7897
Validation Accuracy: 0.7990
Training Loss: 0.4518
Validation Loss: 0.4230
```

Data Augmentation

Data Augmentation



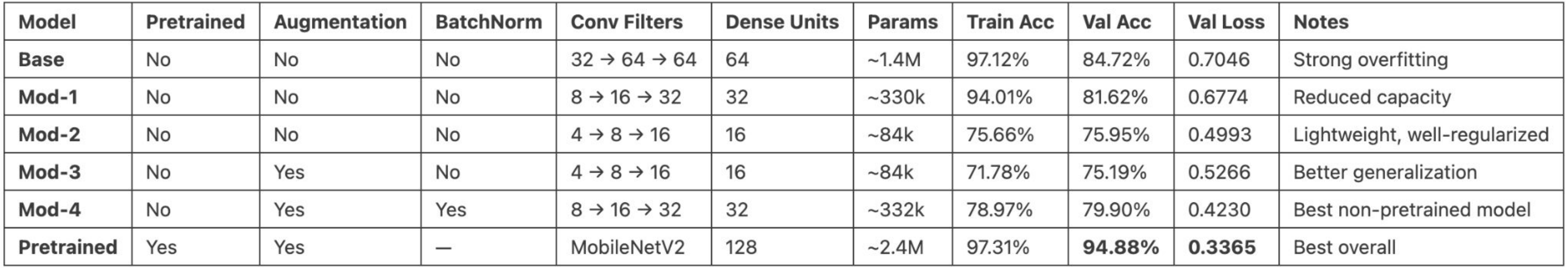
```python
# Data preprocessing
IMG_SIZE = 160
BATCH_SIZE = 32
AUTOTUNE = tf.data.AUTOTUNE

# Preprocessing (resize + normalize)
def preprocess(image, label):
    image = tf.cast(image, tf.float32)
    image = tf.image.resize(image, [IMG_SIZE, IMG_SIZE])
    image = image / 255.0
    return image, label

# Data-Augmentation pipeline
data_augmentation = tf.keras.Sequential([
    # Randomly crop somewhere between 60-100% of the image, then resize
    tf.keras.layers.RandomCrop(IMG_SIZE, IMG_SIZE),
    tf.keras.layers.RandomFlip("horizontal"),
    tf.keras.layers.RandomRotation(0.1),
    tf.keras.layers.RandomZoom(0.1, 0.1),
])
```

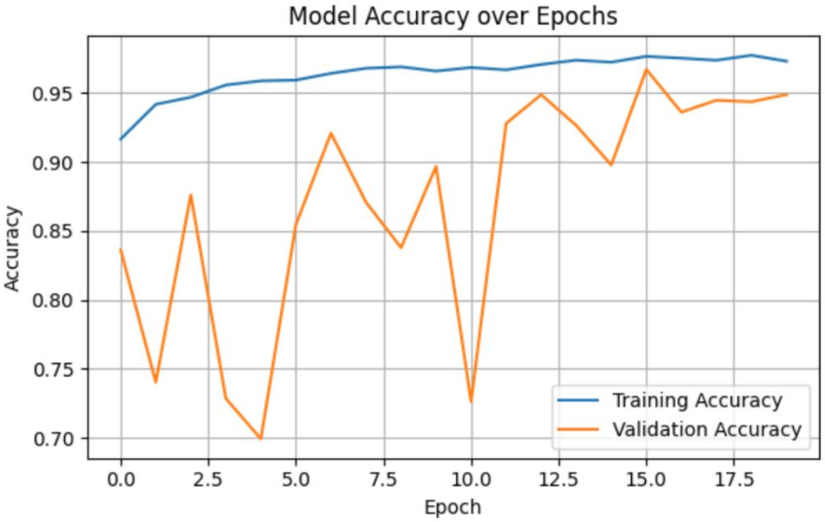
Pre trained Model

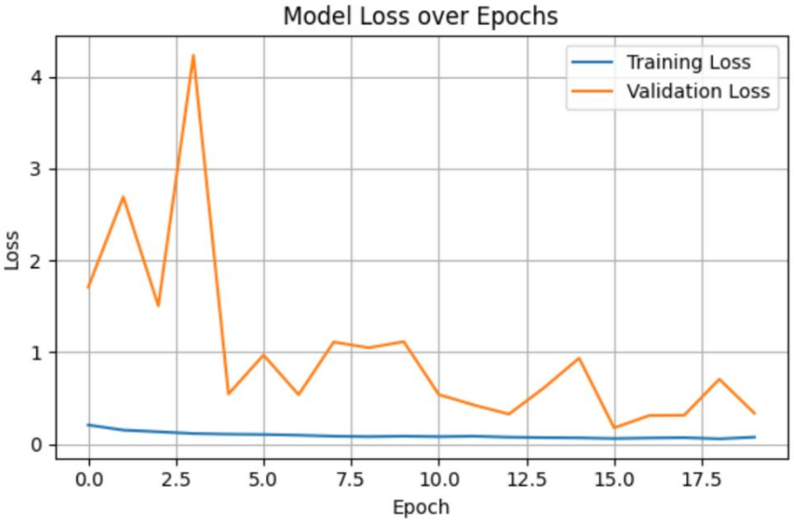
Pre trained Model



Model	Pretrained	Augmentation	BatchNorm	Conv Filters	Dense Units	Params	Train Acc	Val Acc	Val Loss	Notes
Base	No	No	No	32 → 64 → 64	64	~1.4M	97.12%	84.72%	0.7046	Strong overfitting
Mod-1	No	No	No	8 → 16 → 32	32	~330k	94.01%	81.62%	0.6774	Reduced capacity
Mod-2	No	No	No	4 → 8 → 16	16	~84k	75.66%	75.95%	0.4993	Lightweight, well-regularized
Mod-3	No	Yes	No	4 → 8 → 16	16	~84k	71.78%	75.19%	0.5266	Better generalization
Mod-4	No	Yes	Yes	8 → 16 → 32	32	~332k	78.97%	79.90%	0.4230	Best non-pretrained model
Pretrained	Yes	Yes	—	MobileNetV2	128	~2.4M	97.31%	**94.88%**	**0.3365**	Best overall

Pre trained Model

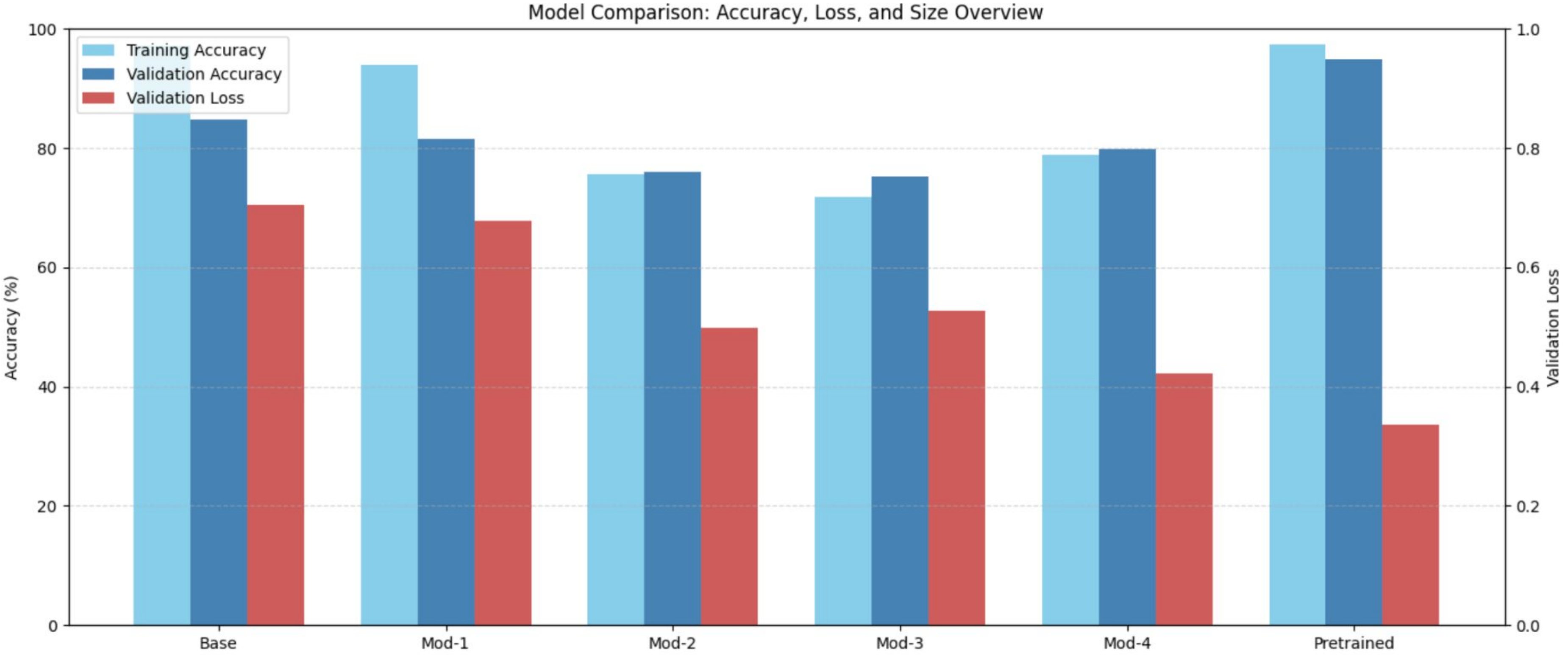




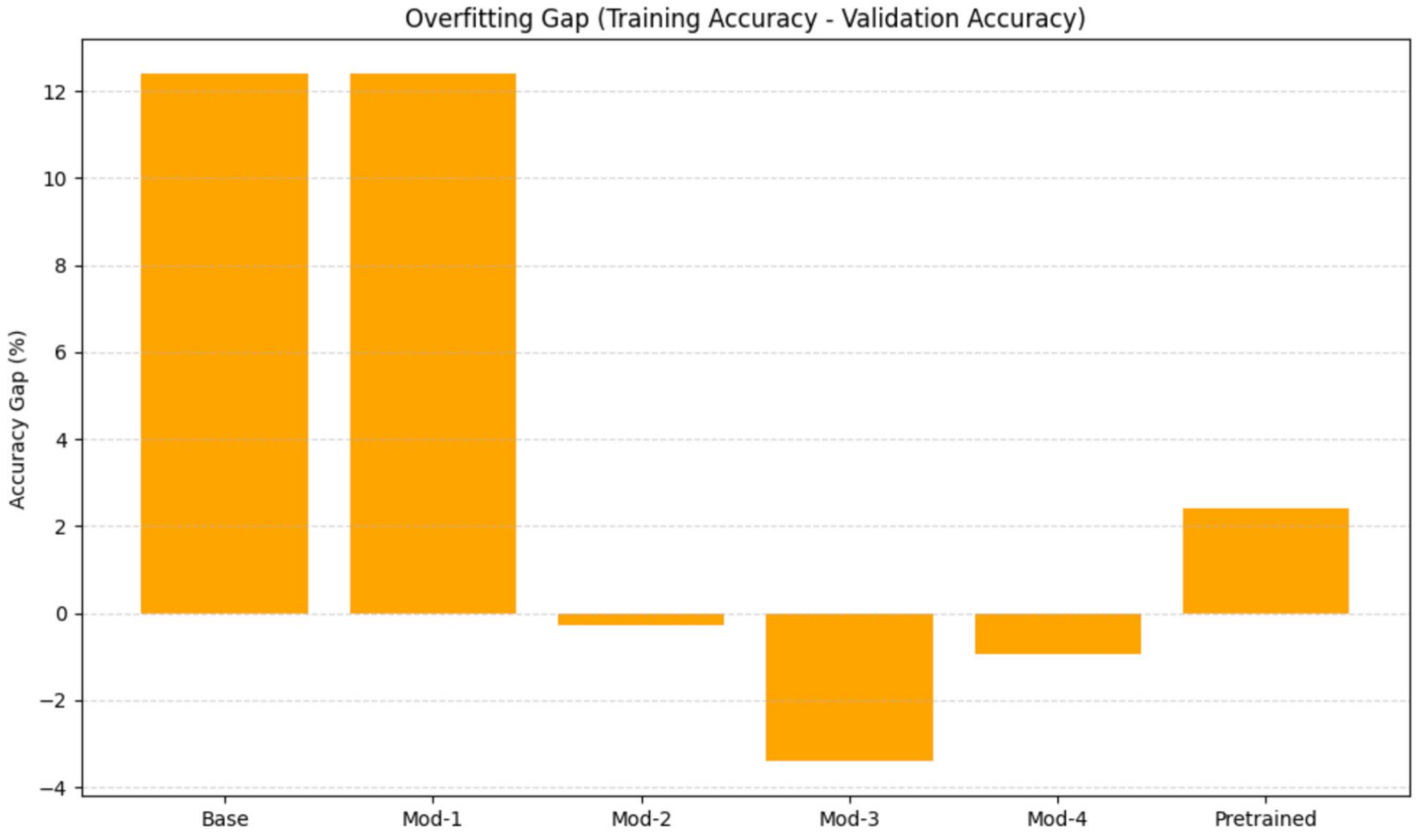
```
Final Training Metrics:
Training Accuracy: 0.9731
Validation Accuracy: 0.9488
Training Loss: 0.0761
Validation Loss: 0.3365
```

Overall Comparison

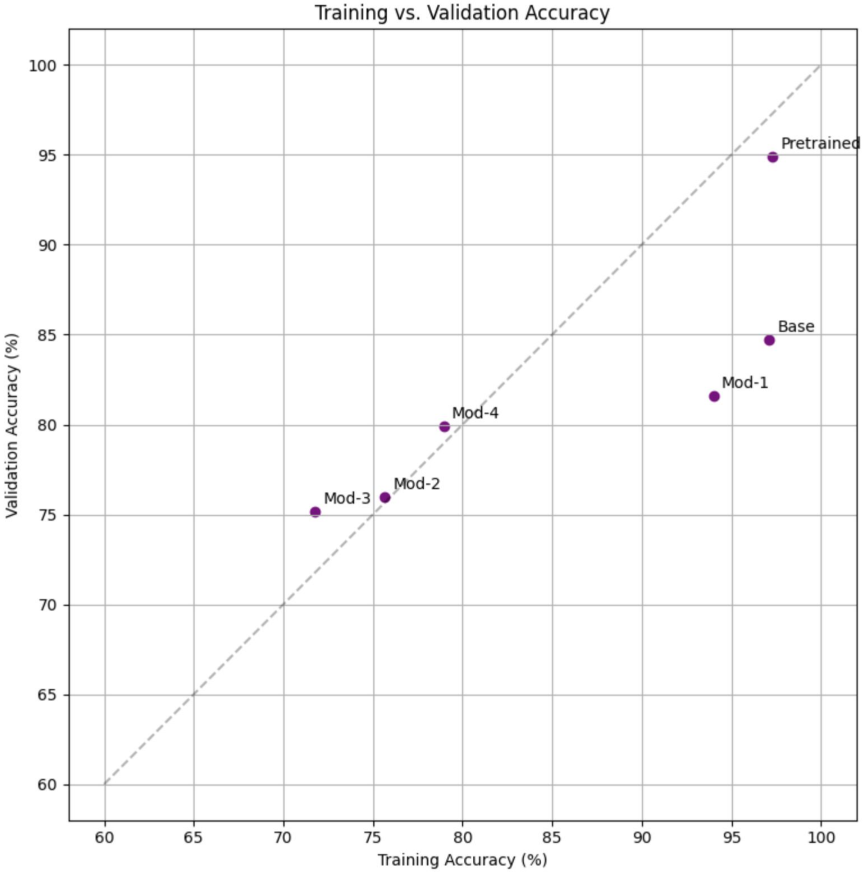
Overall Comparison



Overall Comparison



Overall Comparison


Training vs. Validation Accuracy
Validation Accuracy (%)
Training Accuracy (%)
Pretrained
Base
Mod-1
Mod-4
Mod-2
Mod-3