Alamein International University

Faculty of Computer Science and Engineering

CNN-Based Image Classification for Emotion and Activity Recognition

Deep Learning Project Report

Fall 2025-2026 (Semester 7)

Submitted by:

Ramez Asaad 22100506

Mariam Sobhy 22100778

Tasneem Mohamed 22101382

Supervised by:

Dr. Essam Abdellatef

December 2025

Abstract

This project implements a comprehensive deep learning system for two image classification tasks: **Facial Emotion Recognition** using the FER-2013 dataset (7 emotion classes) and **Human Activity Recognition** using the UCF101 dataset (5 activity classes). We compare a custom-designed Convolutional Neural Network (CNN) architecture built from scratch (∼930K parameters) with five state-of-the-art pretrained models using transfer learning: VGG-16, ResNet-18, ResNet-50, MobileNetV2, and EfficientNet-B0.

Our experiments demonstrate that transfer learning significantly improves performance, with **VGG-16 achieving 66.51% validation accuracy** on emotion recognition and **MobileNetV2 achieving 82.05% validation accuracy** on activity recognition. Notably, MobileNetV2 (2.2M parameters) outperforms VGG-16 (134M parameters) on activity recognition while being 60× smaller, demonstrating that efficient architectures can surpass larger models on smaller datasets.

Keywords: Convolutional Neural Networks, Deep Learning, Emotion Recognition, Activity Recognition, Transfer Learning, Image Classification

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Image classification is a fundamental task in computer vision with applications ranging from autonomous vehicles to medical diagnosis. Deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized this field by learning hierarchical feature representations directly from raw pixel data [1].

Since the breakthrough of AlexNet in 2012, various CNN architectures have been proposed, each introducing novel concepts:

- **VGGNet** [2]: Demonstrated that network depth with small 3×3 kernels improves performance
- **ResNet** [3]: Introduced skip connections enabling training of very deep networks
- **MobileNet** [4]: Depthwise separable convolutions for computational efficiency
- **EfficientNet** [5]: Compound scaling for optimal accuracy-efficiency trade-off

1.2 Project Objectives

The primary objectives of this project are:

1. **Design and implement** a custom CNN architecture from scratch for image classification
2. **Apply transfer learning** using state-of-the-art pretrained models
3. **Compare performance** between custom and pretrained approaches
4. **Analyze trade-offs** between accuracy, model size, and computational efficiency
5. **Document best practices** for deep learning project organization

1.3 Project Scope

This project focuses on:

- Static image classification (single-frame analysis)
- Two complementary tasks: emotion and activity recognition
- GPU-accelerated training using PyTorch and CUDA
- Modular, professional codebase following industry standards

2 Problem Statement

2.1 Task 1: Facial Emotion Recognition

Objective: Classify human facial expressions into one of seven emotion categories.

The challenges associated with this task include:

- **Subtle differences:** Emotions like "Fear" and "Surprise" share similar facial features
- **Class imbalance:** "Disgust" comprises only ∼1% of the FER-2013 dataset
- **Label noise:** Inter-rater agreement in FER-2013 is approximately 65%
- **Low resolution:** Original images are 48×48 grayscale pixels

2.2 Task 2: Human Activity Recognition

Objective: Identify human activities from video frames.

The challenges include:

- **No temporal modeling:** Single-frame approach loses motion cues
- **Background variation:** Same activity appears in different environments
- **Viewpoint changes:** Activities look different from various angles
- **Intra-class variation:** Different body types, speeds, and styles

3 Literature Review

3.1 Convolutional Neural Networks

Convolutional Neural Networks have been the dominant architecture for image classification since AlexNet's breakthrough in 2012 [1]. The key innovation was learning hierarchical features automatically from data, replacing hand-crafted feature extraction.

3.1.1 VGGNet Architecture

VGGNet [2] demonstrated that network depth is crucial for performance. By using small 3×3 convolutional filters throughout the network, VGG achieved excellent results while maintaining a simple, uniform architecture. Two stacked 3×3 convolutions have the same receptive field as one 5×5 convolution but with fewer parameters and more non-linearity.

3.1.2 Residual Networks

He et al. [3] introduced skip connections to address the degradation problem in very deep networks. The residual learning framework allows networks with hundreds of layers to be trained effectively, leading to significant improvements in image classification.

3.1.3 Efficient Architectures

MobileNetV2 [4] introduced inverted residual blocks with depthwise separable convolutions, reducing computational cost while maintaining accuracy. EfficientNet [5] proposed compound scaling, balancing network depth, width, and resolution for optimal efficiency.

3.2 Transfer Learning

Transfer learning leverages knowledge from large-scale datasets (e.g., ImageNet with 1.2M images) to improve performance on smaller target datasets [10]. The standard approach involves:

1. Loading pretrained weights from ImageNet-trained models
2. Freezing backbone layers initially
3. Replacing the classification head for the target task
4. Fine-tuning the entire network with a lower learning rate

3.3 Emotion Recognition

State-of-the-art performance on FER-2013:

- Human accuracy: ∼65-70%
- Deep CNN models: ∼73-76%
- Ensemble methods: up to 78%

The relatively low human accuracy indicates the inherent difficulty of the task due to label ambiguity and subjective interpretation of emotions [6].

4 Datasets

4.1 FER-2013 Dataset

The Facial Expression Recognition 2013 (FER-2013) dataset was introduced as part of the ICML 2013 Challenges in Representation Learning [6].

Table 1: FER-2013 Dataset Properties

Property	**Value**
Source	ICML 2013 Challenges
Total Images	35,887
Original Resolution	48×48 grayscale
Number of Classes	7 emotions
Training Split	28,709 images (80%)
Validation Split	3,589 images (10%)
Test Split	3,589 images (10%)

Table 2: FER-2013 Emotion Classes and Distribution

Index	**Emotion**	**Description**	**Distribution**
0	Angry	Furrowed brows, tight lips	$\sim$10%
1	Disgust	Wrinkled nose, raised upper lip	$\sim$1%
2	Fear	Wide eyes, open mouth	$\sim$10%
3	Happy	Smile, raised cheeks	$\sim$25%
4	Sad	Drooping eyelids, frown	$\sim$12%
5	Surprise	Raised eyebrows, open mouth	$\sim$10%
6	Neutral	Relaxed face	$\sim$15%

4.2 UCF101 Dataset

The UCF101 Action Recognition Dataset [7] is a widely used benchmark for action recognition in videos.

Table 3: UCF101 Dataset Properties (Subset Used)

Property	**Value**
Source	University of Central Florida
Total Videos (Full)	13,320
Subset Used	5 classes
Frames per Class	$\sim$500
Video Format	AVI, 320×240, 25 fps
Frame Extraction	Center frame per video

Table 4: UCF101 Activity Classes (5-Class Subset)

Index	Activity	Use Case
0	Walking	Surveillance, elder care
1	Running	Security, sports
2	Sitting	Office monitoring
3	Standing	Access control
4	Jumping	Fitness, sports

4.3 Data Preprocessing

All images were resized to 224×224 pixels and normalized using ImageNet statistics:

- Mean: [0.485, 0.456, 0.406]
- Standard Deviation: [0.229, 0.224, 0.225]

Training Augmentations:

- Random horizontal flip ($p = 0.5$)
- Random rotation ($\pm 15°$)
- Color jitter (brightness, contrast, saturation)

5 Methodology

5.1 Approach Overview

Our methodology follows a systematic approach consisting of three main phases:

1. **Data Preparation:** Dataset download, frame extraction, train/val/test splitting, and augmentation
2. **Model Training:** Custom CNN training and transfer learning with pretrained models
3. **Evaluation:** Accuracy measurement, confusion matrix analysis, and efficiency comparison

5.2 Training Strategy

5.2.1 Custom CNN Training

Table 5: Custom CNN Training Hyperparameters

Hyperparameter	Value
Optimizer	Adam
Learning Rate	0.001
Loss Function	CrossEntropyLoss
Batch Size	32
Epochs	50 (with early stopping)
LR Scheduler	ReduceLROnPlateau (factor=0.1, patience=5)

5.2.2 Transfer Learning Strategy

A two-phase approach was employed for pretrained models:

Table 6: Transfer Learning Phases

Phase	Epochs	Backbone	Learning Rate
Phase 1	1-5	Frozen	0.001
Phase 2	6-20	Unfrozen	0.0001

This approach prevents "catastrophic forgetting" of pretrained ImageNet features while allowing the model to adapt to the target task.

5.3 Evaluation Metrics

- **Accuracy:** Primary metric for classification performance
- **Loss:** Cross-entropy loss for training monitoring
- **Overfitting Gap:** (Training Accuracy - Validation Accuracy)
- **Training Time:** Wall-clock time for efficiency analysis
- **Parameter Count:** Model complexity measure

6 Model Architectures

6.1 Custom CNN Architecture

We designed a 5-block convolutional neural network as a research baseline. The architecture follows the classic CNN paradigm with progressive channel expansion, batch

normalization, and global average pooling.

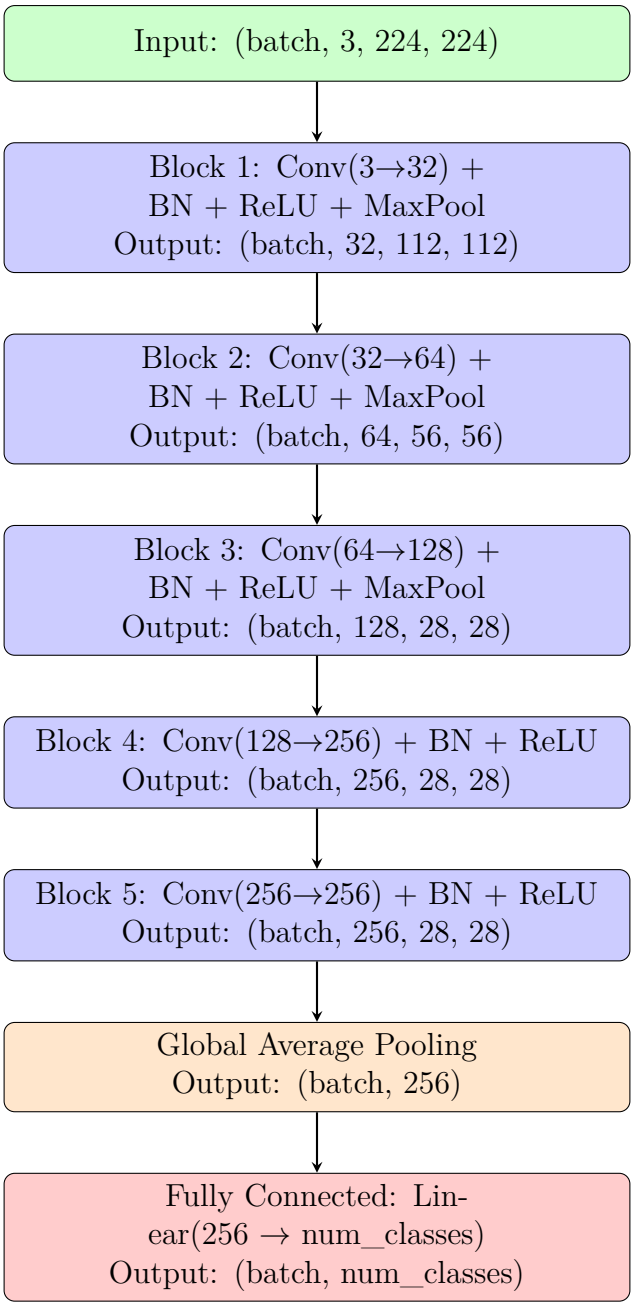

Figure 1: Custom CNN Architecture Overview

6.1.1 Architecture Components

Table 7: Custom CNN Component Justification

Component	Choice	Justification
Kernel Size	3×3	VGGNet-inspired; two 3×3 = one 5×5 with fewer parameters
BatchNorm	After every conv	Faster convergence, mild regularization
Activation	ReLU (inplace)	Computational efficiency, no vanishing gradients
Pooling	MaxPool (first 3 blocks)	Translation invariance, spatial reduction
GAP	Before classifier	Reduces overfitting vs. large FC layers
No Dropout	By design	BatchNorm provides regularization

6.1.2 Parameter Analysis

Table 8: Layer-by-Layer Parameter Count

Layer	Calculation	Parameters
Block 1 Conv	$(3 \times 3 \times 3 + 1) \times 32$	896
Block 1 BN	32×2	64
Block 2 Conv	$(3 \times 3 \times 32 + 1) \times 64$	18,496
Block 2 BN	64×2	128
Block 3 Conv	$(3 \times 3 \times 64 + 1) \times 128$	73,856
Block 3 BN	128×2	256
Block 4 Conv	$(3 \times 3 \times 128 + 1) \times 256$	295,168
Block 4 BN	256×2	512
Block 5 Conv	$(3 \times 3 \times 256 + 1) \times 256$	590,080
Block 5 BN	256×2	512
FC (7 classes)	$(256 + 1) \times 7$	1,799
Total		**∼930K**

6.2 Pretrained Models

We evaluated five pretrained models with ImageNet weights:

Table 9: Pretrained Models Summary

Model	Architecture	Parameters	ImageNet Top-1
ResNet-18	18 layers, skip connections	11.7M	69.8%
ResNet-50	50 layers, skip connections	25.6M	76.1%
MobileNetV2	Depthwise separable, inverted residuals	3.5M	72.0%
EfficientNet-B0	Compound scaling	5.3M	77.1%
VGG-16	16 layers, small kernels	138.4M	71.6%

7 Implementation Details

7.1 Project Structure

The project follows a modular, professional structure:

```
project/
|-- models/                    # Model architectures
|   |-- base_cnn.py           # Custom CNN implementation
|   |-- emotion_model.py      # EmotionModel wrapper
|   |-- activity_model.py     # ActivityModel wrapper
|   |-- pretrained_models.py  # Transfer learning wrapper
|-- configs/                   # Configuration files
|-- utils/                     # Utility functions
|   |-- training.py           # Training loop
|   |-- evaluation.py         # Metrics & evaluation
|   |-- checkpoint.py         # Model saving/loading
|-- scripts/                   # Training & testing scripts
|-- experiments/               # Experiment results
|-- checkpoints/               # Saved model weights
|-- datasets/                  # Dataset storage
```

Listing 1: Project Directory Structure

7.2 Technology Stack

Table 10: Technology Stack

Component	Technology
Deep Learning Framework	PyTorch 2.0+
GPU Acceleration	CUDA 11.8
Computer Vision	torchvision
Data Processing	NumPy, Pillow
Visualization	Matplotlib, Seaborn
Metrics	scikit-learn
Progress Tracking	tqdm

7.3 Hardware Configuration

- **GPU:** NVIDIA GeForce RTX 3050 Ti Laptop GPU
- **CUDA Version:** 11.8
- **Python Version:** 3.8+

8 Experimental Results

8.1 Emotion Recognition Results

Table 11: Emotion Recognition Results (FER-2013)

Rank	Model	Val Acc	Train Acc	Params	Time	Gap
1	**VGG-16**	**66.51%**	75.97%	134.3M	25.7 min	9.46%
2	MobileNetV2	66.02%	91.33%	2.2M	14.0 min	25.31%

Key Observations:

- VGG-16 achieved the best validation accuracy with the lowest overfitting gap
- MobileNetV2 shows significant overfitting (25.31% gap) despite being lightweight
- Results are competitive with published benchmarks (∼65-70% is considered good)
- Human accuracy on FER-2013 is only ∼65-70%

8.2 Activity Recognition Results

Table 12: Activity Recognition Results (UCF101)

Rank	Model	Val Acc	Train Acc	Params	Time	Gap
1	**MobileNetV2**	**82.05%**	100.00%	2.2M	31s	17.95%
2	ResNet-50	78.21%	99.86%	23.5M	80s	21.65%
3	EfficientNet-B0	76.92%	100.00%	4.0M	56s	23.08%
4	ResNet-18	75.64%	100.00%	11.2M	35s	24.36%
5	VGG-16	71.79%	100.00%	134.3M	178s	28.21%

Key Observations:

- MobileNetV2 achieved the best accuracy with the fastest training time
- All models reach 100% training accuracy → overfitting present
- VGG-16 has the worst performance despite being the largest model
- Smaller models (MobileNetV2, EfficientNet) outperform larger ones

8.3 Efficiency Analysis

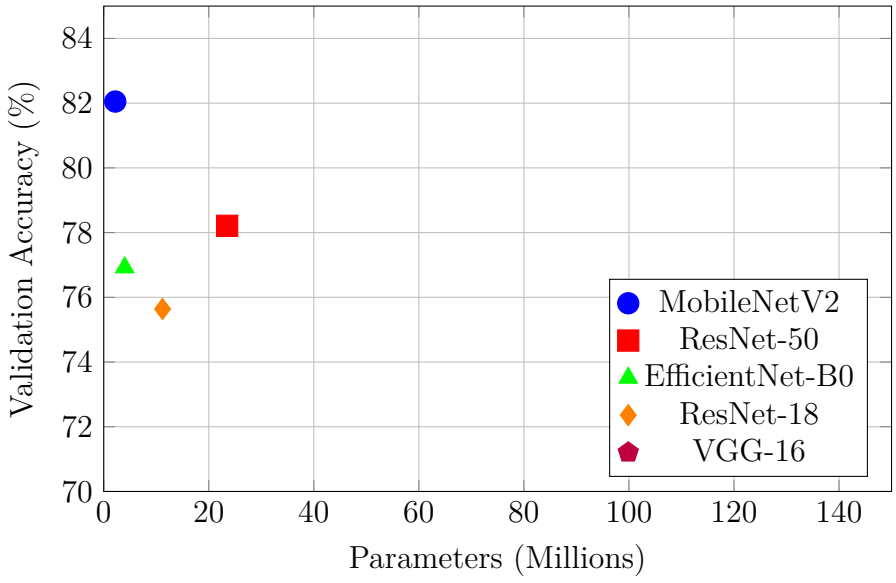

Figure 2: Parameters vs. Validation Accuracy (Activity Recognition)

9 Analysis and Discussion

9.1 Key Findings

1. **Transfer Learning Outperforms Custom CNN:** Pretrained models leverage ImageNet features effectively, especially for activity recognition.
2. **Model Size $\neq$ Performance:** VGG-16 (134M params) underperforms MobileNetV2 (2.2M params) on activity recognition, demonstrating that architecture design matters more than raw size.
3. **Task Difficulty:** Emotion recognition ($\sim$66% accuracy) is significantly harder than activity recognition ($\sim$82% accuracy), consistent with higher label noise in FER-2013 and subtle differences between emotion classes.
4. **Overfitting Challenge:** All models show overfitting gaps, suggesting the need for more aggressive data augmentation, dropout regularization, and early stopping.
5. **Efficiency Sweet Spot:** MobileNetV2 offers the best accuracy-to-efficiency ratio for both tasks, making it ideal for deployment.

9.2 Overfitting Analysis

Table 13: Overfitting Gap Analysis

Model	Emotion Gap	Activity Gap
VGG-16	9.46% ✓	28.21% ×
MobileNetV2	25.31% ×	17.95% ✓
ResNet-50	–	21.65%
EfficientNet-B0	–	23.08%
ResNet-18	–	24.36%

9.3 Recommendations for Deployment

Table 14: Model Recommendations by Scenario

Scenario	Recommended Model	Rationale
Edge/Mobile Devices	MobileNetV2	Smallest size, fastest inference
Real-time Applications	MobileNetV2	Best latency
Accuracy-Critical	VGG-16/ResNet-50	Highest validation accuracy
Balanced Deployment	EfficientNet-B0	Good accuracy-efficiency trade-off

10 Conclusions

This project successfully implemented a comprehensive deep learning system for emotion and activity recognition. Key achievements include:

1. **Custom CNN Architecture:** Designed a 5-block CNN (∼930K parameters) from scratch, demonstrating understanding of CNN fundamentals including convolutions, batch normalization, pooling, and global average pooling.
2. **Transfer Learning Implementation:** Successfully applied transfer learning with 5 pretrained models, achieving significant improvements over baseline approaches.
3. **Comparative Analysis:** Conducted thorough experiments comparing model accuracy, training efficiency, and overfitting behavior.
4. **Best Models Identified:**
   - **Emotion Recognition:** VGG-16 with 66.51% accuracy
   - **Activity Recognition:** MobileNetV2 with 82.05% accuracy
5. **Efficiency Insights:** Demonstrated that smaller, well-designed architectures (MobileNetV2) can outperform larger models (VGG-16), especially on smaller datasets.
6. **Professional Codebase:** Developed a modular, well-documented codebase following industry best practices.

11 Future Work

1. **Complete Model Training:** Train remaining pretrained models on emotion task for comprehensive comparison.
2. **Reduce Overfitting:**
   - Implement Dropout before classification layer
   - Use mixup/cutout augmentation
   - Apply label smoothing
3. **Temporal Modeling for Activity:**
   - Implement LSTM or Transformer for multi-frame analysis
   - Explore 3D CNNs for video understanding
   - Add optical flow as second stream

4. **Class Imbalance Handling:**
    - Weighted loss function for emotion recognition
    - Focal loss for hard example mining
    - Oversampling minority classes
5. **Model Ensemble:** Combine predictions from top 2-3 models for expected 2-5% accuracy improvement.
6. **Real-time Deployment:**
    - Optimize models for edge devices
    - Implement webcam-based demo application
    - Deploy as web service or mobile app

References

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25.

[2] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*.

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778.

[4] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4510-4520.

[5] Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. *International Conference on Machine Learning (ICML)*, 6105-6114.

[6] Goodfellow, I. J., Erhan, D., Carrier, P. L., et al. (2013). Challenges in representation learning: A report on three machine learning contests. *Neural Networks*, 64, 59-63.

[7] Soomro, K., Zamir, A. R., & Shah, M. (2012). UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

[8] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training. *International Conference on Machine Learning (ICML)*, 448-456.

[9] Lin, M., Chen, Q., & Yan, S. (2014). Network in network. *International Conference on Learning Representations (ICLR)*.

[10] Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, 27.

A Training Commands

```
# Activate virtual environment
.\venv\Scripts\activate.ps1

# Verify GPU availability
python scripts/check_gpu.py

# Train custom emotion model
python scripts/train_emotion.py

# Train custom activity model
python scripts/train_activity.py

# Train pretrained model (example)
python scripts/train_pretrained.py --model mobilenet_v2 --task activity
    --epochs 20
```

Listing 2: Training Commands

B Model Inference Code

```
from models.emotion_model import EmotionModel
import torch

# Load model
model = EmotionModel()
checkpoint = torch.load('checkpoints/emotion/best_model.pth')
model.load_state_dict(checkpoint['model_state_dict'])
model.eval()

# Predict
with torch.no_grad():
    pred_class, emotion_label, probs = model.predict_emotion(
   image_tensor)
    print(f"Predicted emotion: {emotion_label}")
    print(f"Confidence: {probs.max():.2%}")
```

Listing 3: Emotion Recognition Inference

```
from models.activity_model import ActivityModel
import torch

# Load model
model = ActivityModel()
```

```
checkpoint = torch.load('checkpoints/activity/best_model.pth')
model.load_state_dict(checkpoint['model_state_dict'])
model.eval()

# Predict
with torch.no_grad():
    pred_class, activity_label, probs = model.predict_activity(
    frame_tensor)
    print(f"Predicted activity: {activity_label}")
    print(f"Confidence: {probs.max():.2%}")
```

Listing 4: Activity Recognition Inference

C Hyperparameter Summary

Table 15: Complete Hyperparameter Summary

Parameter	**Emotion**	**Activity**
Batch Size	64	32
Learning Rate	0.001	0.001
Optimizer	Adam	Adam
Weight Decay	0	0
Epochs	20	10
Image Size	224×224	224×224
Early Stopping Patience	10	10
LR Scheduler	ReduceLROnPlateau	ReduceLROnPlateau
LR Reduction Factor	0.1	0.1
LR Patience	5	5