CNN Models Documentation

Farm Insects Classification Project

Phase 1:

Model 1: LeNet-5 (From Scratch)

1. Data Preparation

- Resized all images to 32x32 pixels
- Converted images to grayscale (1 channel)
- Split data into training, validation, and test sets
- Batch size: 32

2. Model Architecture

- Feature extraction: 2 convolutional layers + 2 average pooling layers
- Classification: 3 fully connected layers
- Activation function: Tanh
- Total trainable parameters: ~61,706

3. Training Configuration

- Number of epochs: 10
- Learning rate: 0.0001
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Validated on validation set after each epoch
- Tracked loss and accuracy for both

5. Evaluation

- Tested on test set
- Calculated accuracy, precision, recall, and F1-score

Final Test Set Metrics:

- Accuracy: 0.1203
- Precision: 0.0874
- Recall: 0.1203
- F1-score: 0.0909

Model 2: AlexNet

1. Data Preparation

- Resized images to 224x224 pixels
- Kept RGB color (3 channels)
- Applied ImageNet normalization
- Batch size: 32

2. Model Architecture

- 5 convolutional layers with ReLU activation
- 3 max pooling layers
- 3 fully connected layers with dropout
- Started with random weights

3. Training Configuration

- Number of epochs: 10
- Learning rate: 0.0001
- Optimizer: Adam

- Loss function: Cross-Entropy Loss

4. Training Process

- Trained and validated each epoch
- Saved best model based on validation accuracy
- Loaded best model for testing

Final Test Set Metrics:

- Training Loss: 0.1310
- Training Accuracy: 0.9683
- Validation Loss: 1.2401
- Validation Accuracy: 0.6917
- Final Test Accuracy: 0.6930

Model 3: VGG16 (From Scratch)

1. Data Preparation

- Resized images to 224x224 pixels
- Applied data augmentation for training:
    - Random horizontal flips
    - Random rotations (10 degrees)
- Used ImageNet normalization
- Batch size: 32

2. Model Architecture

- 13 convolutional layers (all 3x3 kernels)
- 5 max pooling layers
- 3 fully connected layers with dropout (0.5)
- ReLU activation throughout

- Started with random weights

3. Training Configuration

- Number of epochs: 5
- Learning rate: 0.00001 (1e-5)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6482
- Precision: 0.6577
- Recall: 0.6482
- F1-score: 0.6461

Summary

Three CNN architectures were implemented and trained from scratch on a farm insects classification dataset:

1. **LeNet-5:** Simple architecture with ~61K parameters, trained for 10 epochs
2. **AlexNet:** Deeper model with more parameters, trained for 10 epochs
3. **VGG16:** Very deep model with 16 layers, trained for 5 epochs with lower learning rate

Phase 2:

Model: GoogLeNet (Feature Extraction)

1. Data Preparation

- Resized images to 224x224 pixels
- Used ImageNet normalization (mean= [0.485, 0.456, 0.406], std= [0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- Base Model: GoogLeNet
- Weights: Pre-trained on ImageNet (IMAGENET1K_V1)
- Modifications:
    - Auxiliary logits disabled (aux_logits=False)
    - Final fully connected layer replaced to match number of classes
- Freezing: All layers frozen except the final fully connected layer

3. Training Configuration

- Number of epochs: 10
- Learning rate: 0.001 (1e-3)
- Optimizer: SGD (momentum=0.9)
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Validated on validation set per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6234

Model: GoogleNet (Full Tuning)

1. Data Preparation

- Resized images to 224x224 pixels
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- GoogleNet (Inception v1) architecture
- Initialized with pre-trained ImageNet weights (IMAGENET1K_V1)
- Replaced final fully connected layer (1024 input features)
- All layers set to trainable (Full Tuning)

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.0001 (1e-4)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.7057
- Precision: 0.7247
- Recall: 0.7057
- F1-score: 0.7064

Model: GoogLeNet (Partial Tuning)

1. Data Preparation

1. Resized images to 224x224 pixels
2. Used ImageNet normalization (mean= [0.485, 0.456, 0.406], std= [0.229, 0.224, 0.225])
3. Batch size: 32

2. Model Architecture

- GoogleNet (Inception v1) architecture
- Frozen all initial layers
- Unfrozen the last Inception block (inception5b) and the final classifier
- Replaced final fully connected layer (Linear 1024 input features)
- Optimizer updates only unfrozen parameters

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.0001 (1e-4)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6482
- Precision: 0.6577
- Recall: 0.6482

- F1-score: 0.6461

Key Findings:

Full Fine-Tuning achieved the highest performance across all metrics, with an accuracy of 70.57%, representing an 8.23 percentage point improvement over feature extraction and a 5.75 percentage point improvement over partial tuning. This approach allowed the model to adapt all layers to the specific dataset characteristics, resulting in superior classification performance.

Partial Fine-Tuning yielded moderate performance (64.82% accuracy), falling between the frozen feature extractor and full tuning approaches. By unfreezing only the last Inception block (inception5b) and classifier, the model gained some adaptability while maintaining much of the pre-trained knowledge from earlier layers.

Feature Extraction (frozen model) produced the lowest accuracy at 62.34%, as only the final classifier layer was trained. This limited the model's ability to adapt learned features to the specific dataset, though it still leveraged the strong representational power of ImageNet pre-training.

Resource Usage Comparison

Strategy	Trainable Parameters	Training Time	Memory Requirements	Computational Cost
Feature Extraction	Minimal (~1% of model)	Lowest (10 epochs)	Lowest	Lowest
Partial Fine-Tuning	Moderate (~20-30% of model)	Moderate (20 epochs)	Moderate	Moderate
Full Fine-Tuning	Maximum (100% of model)	Highest (20 epochs)	Highest	Highest

Resource Analysis:

Feature Extraction is the most resource-efficient approach, requiring minimal GPU memory and computational resources since only the final classifier is trained. With just 10 epochs needed, training time is significantly reduced. This makes it ideal for quick prototyping or resource-constrained environments.

Partial Fine-Tuning offers a balanced middle ground, updating approximately 20-30% of the model's parameters (the last Inception block and classifier). This requires moderate computational resources and memory, with training time comparable to full tuning due to the 20-epoch duration but with lower per-epoch computational cost.

Full Fine-Tuning demands the highest computational resources, as all layers require gradient computation and parameter updates. This results in the longest training time per epoch and highest memory consumption, requiring more powerful hardware (larger GPU memory) for effective training.

Trade-offs Analysis

Feature Extraction (Frozen Model):

- **Advantages:** Fastest training, lowest resource requirements, minimal risk of overfitting, excellent for small datasets or quick prototyping
- **Disadvantages:** Limited adaptability to target domain, lowest performance, features may not align perfectly with dataset characteristics
- **Best Use Case:** Small datasets, quick experiments, severe computational constraints, datasets like ImageNet

Full Fine-Tuning:

- **Advantages:** Highest performance, maximum adaptability to target dataset, best feature alignment with specific task
- **Disadvantages:** Highest computational cost, longest training time, increased risk of overfitting (especially with small datasets), requires significant GPU memory
- **Best Use Case:** Large datasets, sufficient computational resources available, maximum performance required, dataset differs significantly from ImageNet

Partial Fine-Tuning:

- **Advantages:** Balanced performance-resource trade-off, reduced overfitting risk compared to full tuning, moderate computational requirements, retains low-level features while adapting high-level representations
- **Disadvantages:** Performance between the two extremes, requires careful selection of which layers to unfreeze, may not fully exploit dataset-specific patterns

- **Best Use Case:** Medium-sized datasets, moderate computational resources, when feature extraction underperforms but full tuning is too resource-intensive

---

Phase 3:

InceptionV3 Models

Model: InceptionV3 (Feature Extraction)

1. Data Preparation

- Resized images to 299x299 pixels
- Used ImageNet normalization (mean= [0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- InceptionV3 architecture
- Initialized with pre-trained ImageNet weights
- Auxiliary logits disabled (aux_logits=False)
- Replaced final fully connected layer (2048 input features)
- Frozen all backbone parameters (feature extractor only)
- Optimizer updates only the final classifier

3. Training Configuration

- Number of epochs: 10
- Learning rate: 0.001 (1e-3)
- Optimizer: SGD (momentum=0.9)
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set

- Monitored training and validation loss and accuracy per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6297

Model: InceptionV3 (Partial Tuning)

1. Data Preparation

- Resized images to 299x299 pixels
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- InceptionV3 architecture
- Initialized with pre-trained ImageNet weights
- Frozen all initial layers
- Unfrozen the last Inception block (Mixed_7c) and final classifier
- Auxiliary logits disabled (aux_logits=False)
- Replaced final fully connected layer (2048 input features)
- Optimizer updates only unfrozen parameters

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.0001 (1e-4)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.7215
- Precision: 0.7359
- Recall: 0.7215
- F1-score: 0.7226

Model: InceptionV3 (Full Tuning)

1. Data Preparation

- Resized images to 299x299 pixels
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- InceptionV3 architecture
- Initialized with pre-trained ImageNet weights
- Auxiliary logits disabled (aux_logits=False)
- Replaced final fully connected layer (2048 input features)
- All layers set to trainable (Full Tuning)

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.0001 (1e-4)

- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.7468
- Precision: 0.7542
- Recall: 0.7468
- F1-score: 0.7466

ResNet50 Models

Model: ResNet50 (Feature Extraction)

1. Data Preparation

- Resized images to 224x224 pixels
- Train augmentation: RandomHorizontalFlip + RandomRotation(10)
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- ResNet50 architecture
- Initialized with pre-trained ImageNet weights
- Frozen all backbone layers
- Replaced final fully connected layer

- Optimizer updates only the new classifier parameters

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.001 (1e-3)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6245
- Precision: 0.6596
- Recall: 0.6245
- F1-score: 0.6167

Model: ResNet50 (Partial Tuning)

1. Data Preparation

- Resized images to 224x224 pixels
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- ResNet50 architecture
- Initialized with pre-trained ImageNet weights

- Frozen all initial layers
- Unfrozen the last residual block (layer4) and final classifier
- Replaced final fully connected layer
- Optimizer updates only unfrozen parameters

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.0001 (1e-4)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.6957
- Precision: 0.7296
- Recall: 0.6957
- F1-score: 0.6946

Model: ResNet50 (Full Tuning)

1. Data Preparation

- Resized images to 224x224 pixels
- Used ImageNet normalization (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Batch size: 32

2. Model Architecture

- ResNet50 architecture
- Initialized with pre-trained ImageNet weights
- Replaced final fully connected layer
- All layers set to trainable (Full Tuning)

3. Training Configuration

- Number of epochs: 20
- Learning rate: 0.00001 (1e-5)
- Optimizer: Adam
- Loss function: Cross-Entropy Loss

4. Training Process

- Trained on training set
- Monitored training loss per epoch
- Evaluated on test set after training

Final Test Set Metrics:

- Accuracy: 0.7154
- Precision: 0.7245
- Recall: 0.7154
- F1-score: 0.7140

Key Findings:

- Full Tuning consistently achieves the highest accuracy for both architectures
- InceptionV3 outperforms ResNet50 across all strategies (2-3% higher accuracy)
- Feature Extraction shows similar baseline performance (~62-63%) for both models
- The performance gap between strategies is more pronounced in InceptionV3

Resource Consumption Analysis

Computational Resources

Model	Strategy	Trainable Parameters	Training Epochs	Learning Rate	Optimizer
InceptionV3	Feature Extraction	Classifier only (~21K)	10	1e-3	SGD
	Partial Tuning	Mixed_7c + Classifier (~5-7M)	20	1e-4	Adam
	Full Tuning	All parameters (~22M)	20	1e-4	Adam
ResNet50	Feature Extraction	Classifier only (~10K)	20	1e-3	Adam
	Partial Tuning	Layer4 + Classifier (~7-9M)	20	1e-4	Adam
	Full Tuning	All parameters (~25M)	20	1e-5	Adam

Resource Intensity Rankings

1. **Feature Extraction**: Lowest (minimal parameters, shorter training)
2. **Partial Tuning**: Moderate (30-40% of total parameters)
3. **Full Tuning**: Highest (all parameters, requires careful learning rate tuning)

Training Time Estimates

- **Feature Extraction**: ~1x baseline (fastest convergence)
- **Partial Tuning**: ~3-4x baseline (moderate backpropagation depth)
- **Full Tuning**: ~5-7x baseline (full network backpropagation)

Strategy Trade-offs

Feature Extraction

Advantages:

- Fastest training time
- Minimal computational resources

- No risk of catastrophic forgetting
- Works well when source and target domains are similar
- Ideal for quick prototyping

Disadvantages:

- Limited performance ceiling (~62-63% accuracy)
- Cannot adapt features to new domain specifics
- Poor performance when domains differ significantly

Partial Tuning

Advantages:

- Good balance between performance and efficiency
- Significant improvement over feature extraction (+9-10% accuracy)
- Moderate resource requirements
- Lower overfitting risk than full tuning
- Adapts high-level features while preserving low-level representations

Disadvantages:

- Requires careful selection of layers to unfreeze
- Still limited compared to full tuning (~2-3% accuracy gap)
- Architecture-specific tuning needed

Full Tuning

Advantages:

- Highest accuracy achieved (74.68% InceptionV3, 71.54% ResNet50)
- Complete adaptation to target domain
- Best metrics across precision, recall, and F1-score

- Maximum model flexibility

Disadvantages:

- Highest computational cost (5-7x longer training)
- Requires very careful learning rate tuning (1e-4 to 1e-5)
- Higher risk of overfitting on small datasets
- Increased memory requirements
- Longer experiment iteration cycles

Key Insights

Performance vs. Resource Trade-off

The relationship follows a diminishing returns pattern:

- Feature Extraction → Partial Tuning: +9-10% accuracy gain (moderate resource increase)
- Partial Tuning → Full Tuning: +2-3% accuracy gain (high resource increase)

Architecture Comparison

- InceptionV3 shows better transfer learning capability across all strategies
- InceptionV3's inception modules may provide more flexible feature representations
- ResNet50 requires lower learning rates for full tuning (1e-5 vs 1e-4), suggesting higher sensitivity