# AI-Powered Food Classification and Calorie Estimation Using the Food-101 Dataset

Abdallah Waked

Dr. Mohammed Yousefhussien

Apr 12, 2025

INFO-6147 Deep Learning with PyTorch

# Project Overview

- Goal: Classify food images and estimate caloric value
- Dataset: **Food-101**
- API Integration: **Spoonacular API** for recipe and nutrition details
- Output: Fast food classification and calorie estimation through web app

# Why is this project valuable?

Simplifies food identification and calorie tracking

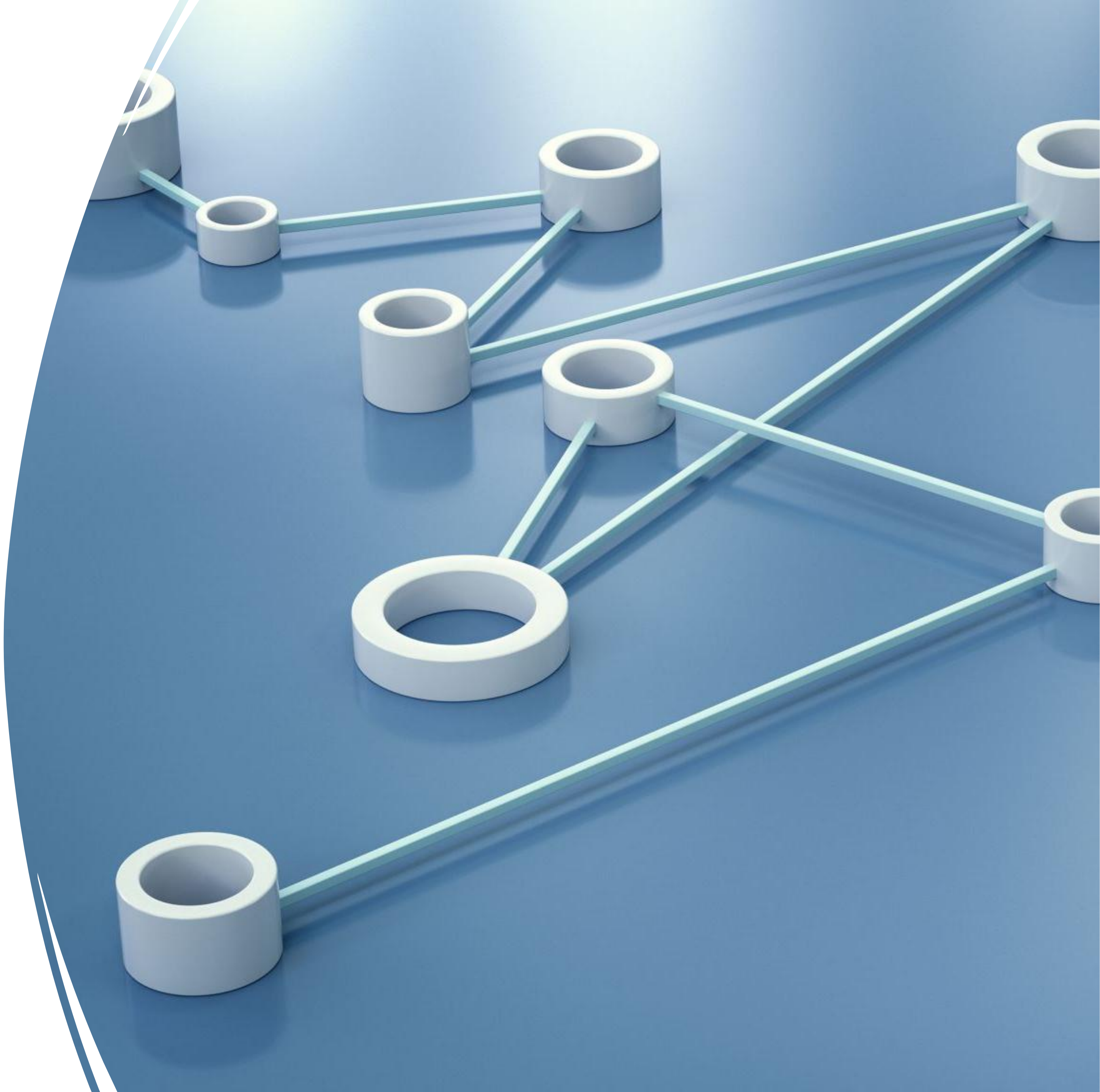
Useful for health-conscious users and diet tracking

Provides instant results using image input

No manual search for nutritional info needed

# Approach & Methodology

- Train a **Convolutional Neural Network (CNN)**
- Use **ResNet18 pretrained model** to improve accuracy
- Integrate with **JSON nutrition database** and Spoonacular API
- Deploy model using **Streamlit web application**

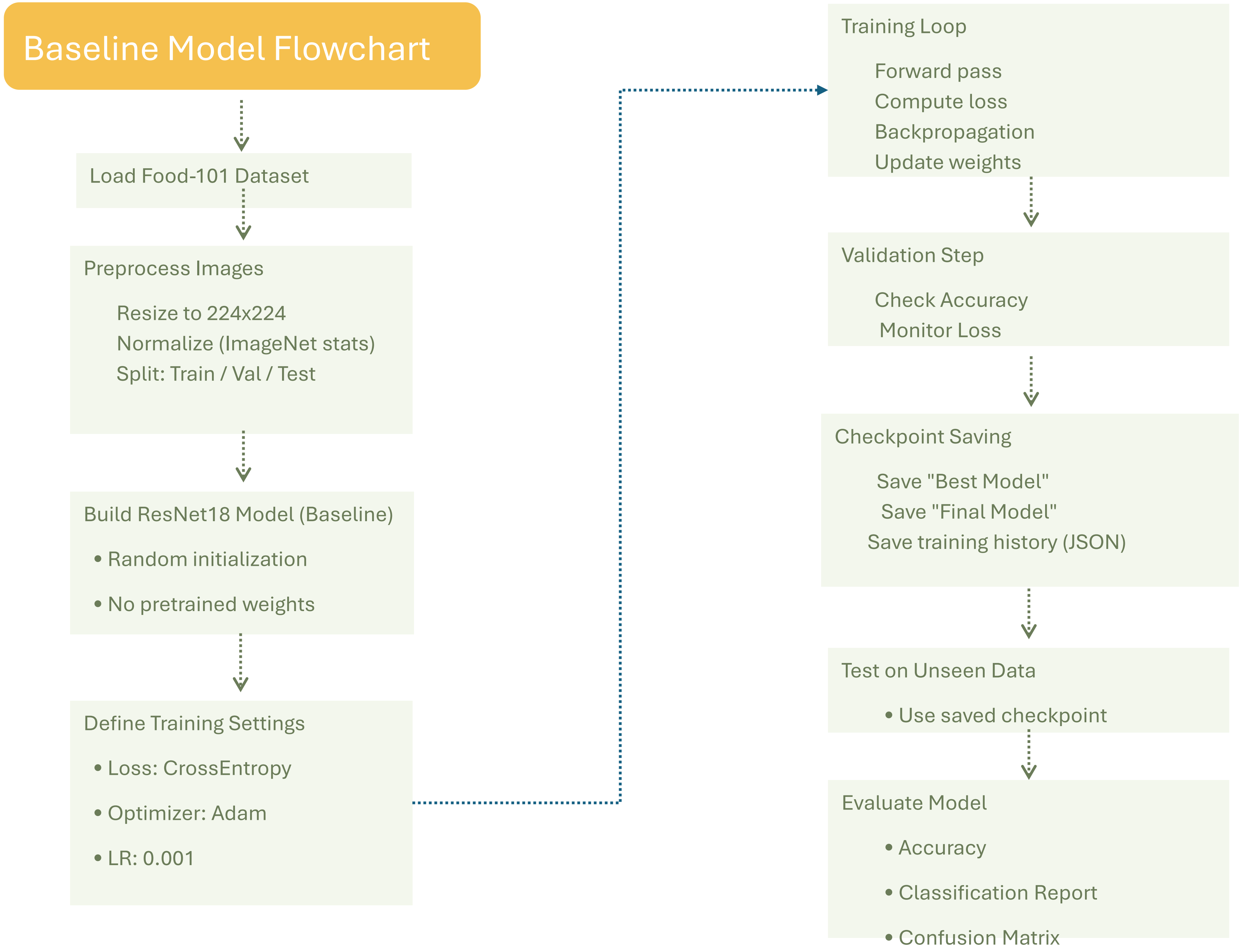


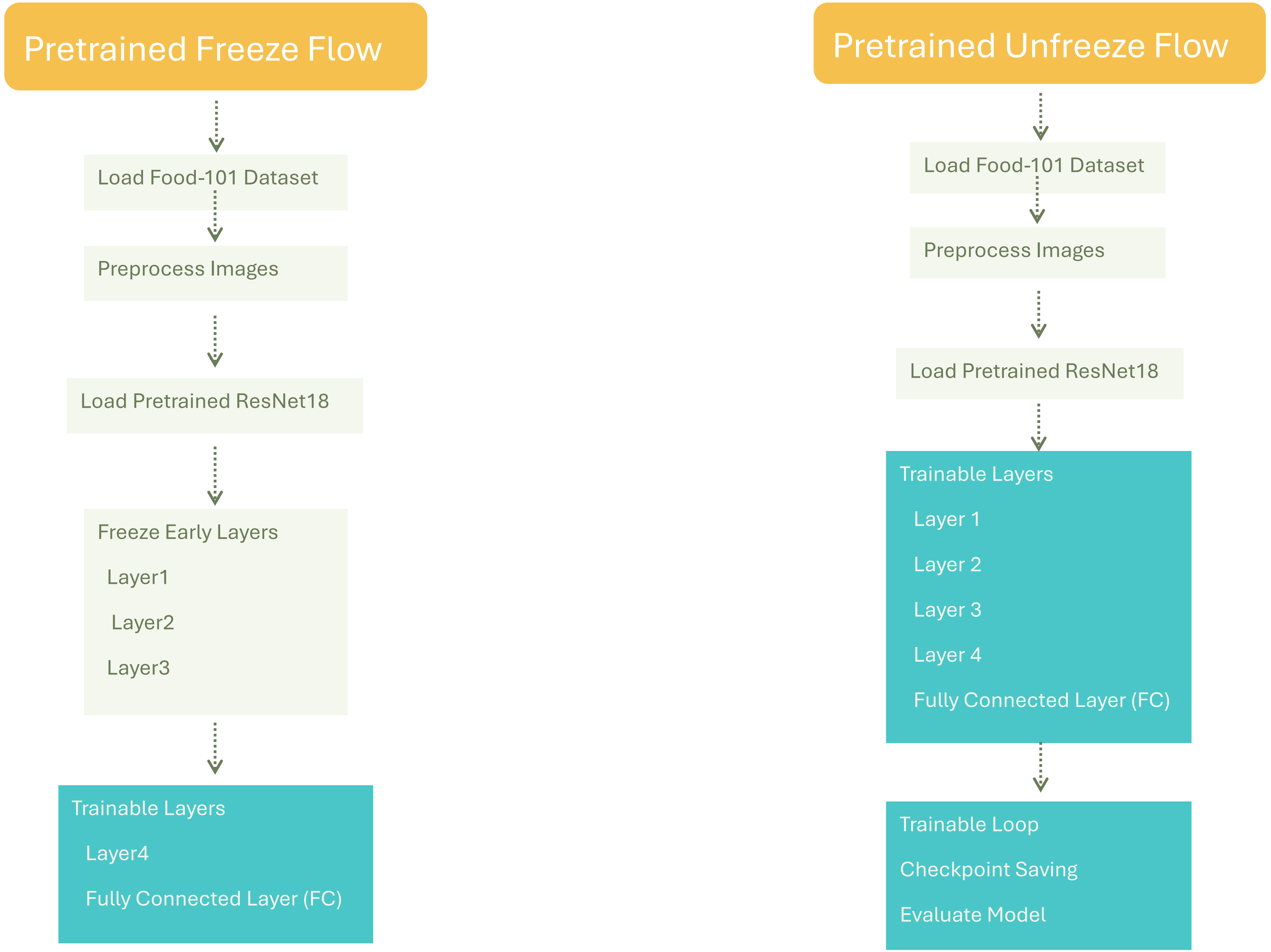
# Dataset

- **Food-101 Dataset**
- Total images: 101,000 (101 categories)
- Split:
  - 70% Training
  - 15% Validation
  - 15% Testing
- Preprocessing:
  - Resize: 224×224
  - Augmentation: Horizontal flip, rotation
  - Normalization: ImageNet mean and std

# Model Architecture (ResNet18)

- Pretrained on ImageNet
- Frozen initial layers, fine-tuned Layer4 and FC
- Custom classifier for 101 food classes
- Efficient for small-to-medium datasets


Baseline Model Flowchart
Load Food-101 Dataset
Preprocess Images
Resize to 224x224
Normalize (ImageNet stats)
Split: Train / Val / Test
Build ResNet18 Model (Baseline)
• Random initialization
• No pretrained weights
Define Training Settings
• Loss: CrossEntropy
• Optimizer: Adam
• LR: 0.001
Training Loop
Forward pass
Compute loss
Backpropagation
Update weights
Validation Step
Check Accuracy
Monitor Loss
Checkpoint Saving
Save "Best Model"
Save "Final Model"
Save training history (JSON)
Test on Unseen Data
• Use saved checkpoint
Evaluate Model
• Accuracy
• Classification Report
• Confusion Matrix


Pretrained Freeze Flow
Load Food-101 Dataset
Preprocess Images
Load Pretrained ResNet18
Freeze Early Layers
Layer1
Layer2
Layer3
Trainable Layers
Layer4
Fully Connected Layer (FC)
Pretrained Unfreeze Flow
Load Food-101 Dataset
Preprocess Images
Load Pretrained ResNet18
Trainable Layers
Layer 1
Layer 2
Layer 3
Layer 4
Fully Connected Layer (FC)
Trainable Loop
Checkpoint Saving
Evaluate Model

# Training Process

- Device: GPU (if available), fallback to CPU
- Epochs: ~10 with early stopping
- Batch Size: 32 → 8 (due to device limits)
- Learning Rate:
  - Started: 0.001
  - Planned: 0.0003 (but limited by Colab quota and hardware)
- Features:
  - Checkpoint saving
  - Best model saving
  - Training history logged

# Hyperparameter Tuning

- Experiments:
  - Learning rates: 0.001, planned 0.0003
  - Batch sizes: 32 → 8
  - Model modes:
    - Pretrained Freeze
    - Pretrained Unfreeze
- Observations:
  - Pretrained models outperformed baseline
  - Freezing sped up training
  - Unfreezing all layers achieved higher accuracy

# Deployment Application

- **Web App:** Streamlit Link
- Auto-downloads model from Google Drive
- User uploads image → Model predicts category
- Click on prediction → Spoonacular API fetches:
    - Recipes
    - Nutritional information
- Code available on GitHub: GitHub Repository

# Challenges & Limitations

- Google Colab quota limits GPU usage
- Local device performance is limited
- Large model size (~5 GB) requires external storage
- API rate limits from Spoonacular
- Time constraints limited fine-tuning
- Complex dataset with varied image conditions

# Results & Evaluation

- Model accuracy: *(Insert your number here)*
- Classification report: *(Add sample screenshot or table)*
- Confusion matrix visualization: *(Optional screenshot)*
- Efficient web deployment achieved

# Conclusion

- Successfully classified food images and estimated calories
- Integrated external API for dynamic recipe and nutrition data
- ResNet18 pretrained model improved accuracy and efficiency
- Deployed as an interactive Streamlit web application

# Thank You!