# MA477: Data Science

**Lesson 31 Outline — 10 April 2026**

United States Military Academy, West Point
Instructor: MAJ Patrick Kuiper

# 1 Administrative

- Student Lesson
- Project 3 Discussion and Pitch
- Student review
- mentimeter
- Transfer Learning Discussion
- Coding Exercise

# 2 Transfer Lesson Objectives

- Know how to leverage pretrained models from Keras for transfer learning

# 3 Deep CNN Architectures: Key Innovations

## 3.1 LeNet — The Convolutional Pipeline

**Innovation:** Structured pipeline (Convolution $\rightarrow$ Pooling $\rightarrow$ Dense)

- First model to treat images as spatial data rather than flat vectors
- Convolutions extract local features (edges, patterns)
- Pooling reduces spatial resolution and computation
- Dense layers perform final classification

**Key Insight:** Hierarchical feature extraction transforms pixels into semantic representations.

**Takeaway:** Foundation of all modern CNN architectures.

## 3.2 AlexNet — Making Deep Learning Practical

**Innovation 1:** ReLU Activation

$$f(x) = \max(0, x)$$

- Prevents vanishing gradients compared to sigmoid/tanh
- Enables faster and more stable training

**Innovation 2:** Dropout

- Randomly disables neurons during training
- Prevents overfitting and co-adaptation
- Acts like training an ensemble of subnetworks

**Takeaway:** Enabled deep CNNs to train effectively at scale.

## 3.3 VGGNet — Depth Through Simplicity

**Innovation:** Stacked $3 \times 3$ Convolutions

- Replace large filters (e.g., $5 \times 5$) with multiple $3 \times 3$ layers
- Same receptive field with fewer parameters
- More nonlinearities increase model expressiveness

**Key Insight:**

$$\text{Two } (3 \times 3) \approx \text{One } (5 \times 5)$$

**Takeaway:** Deep, uniform architectures improve performance.

## 3.4 GoogLeNet (Inception) — Multi-Scale Learning

**Innovation:** Inception Module (Parallel Convolutions)

- Apply $1 \times 1$, $3 \times 3$, $5 \times 5$, and pooling in parallel
- Concatenate outputs along channel dimension

**Why it works:**

- Captures features at multiple spatial scales
- Lets the network learn which scale is most important

**Secondary Innovation:** $1 \times 1$ Convolutions

- Reduce dimensionality before expensive operations
- Improve computational efficiency

**Takeaway:** Learn multi-scale features efficiently.

## 3.5 ResNet — Residual Learning

**Innovation:** Skip (Residual) Connections

$$F(x) = H(x) - x \quad \Rightarrow \quad y = F(x) + x$$

- Learns residual (adjustment) instead of full mapping
- Allows gradients to flow directly through the network
- Prevents degradation in very deep models

**Key Insight:** Learning small corrections is easier than learning full transformations.
**Takeaway:** Enables extremely deep networks (50+ layers).

### 3.6 DenseNet — Feature Reuse

**Innovation:** Dense Connectivity

$$x_l = [x_0, x_1, x_2, \ldots, x_{l-1}]$$

- Each layer receives all previous feature maps
- Improves gradient flow and information sharing
- Reduces need to relearn features

**Key Insight:** All features remain accessible throughout the network.
**Takeaway:** Maximizes information flow and efficiency.

## 4 Transfer Learning

### 4.1 Core Idea

- CNNs learn general visual features (edges, textures, shapes)
- These features transfer across tasks

**Interpretation:**

- Early layers: general-purpose feature extractors
- Later layers: task-specific decision makers

### 4.2 Strategy 1: Feature Extraction

- Freeze pretrained convolutional base
- Train only new classification head

**Use when:** Small dataset, similar domain

### 4.3 Strategy 2: Fine-Tuning

- Unfreeze top layers
- Train with small learning rate

**Use when:** Moderate data, slight domain shift

### 4.4 Strategy 3: Full Training

- Train entire network from scratch

**Use when:** Large datasets (e.g., ImageNet scale)

### 4.5 Practical Workflow

1. Load pretrained model
2. Remove original output layer
3. Add new task-specific head
4. Freeze base and train head
5. Optionally fine-tune upper layers

## 4.6 Final Summary

- LeNet: CNN structure
- AlexNet: Trainability (ReLU, Dropout)
- VGG: Depth via small filters
- Inception: Multi-scale learning
- ResNet: Residual connections (deep training)
- DenseNet: Feature reuse
- Transfer Learning: Reuse pretrained representations