Pre-reading Computer Vision Lecture 2

Tasks in Computer Vision ResNet, Inception, Transfer Learning, Batch Normalization, Object Localization & Detection

Questions

1. What is the residual block?

A residual block is a building block of ResNet (Residual Network) where the output of a layer is directly added to its input, forming a "shortcut" or "skip connection."

Why it's needed:

- Deep networks often face the problem of vanishing gradients, where the signal becomes too weak to adjust weights effectively as layers get deeper. By learning the "residuals" (small changes), the network avoids this problem.
- Residual blocks ensure that layers simply "pass through" information when deeper transformations are unnecessary.

Real-world example: Imagine you're working on a complex software project, and a new feature is added that introduces a small bug. Instead of rewriting the entire software from scratch, you only commit a patch that addresses the bug. Similarly, a residual block in a neural network doesn't overhaul the entire output of the previous layers. Instead, it focuses on learning the "patch" or the residual (small corrections) required to improve the network's predictions.

---

2. What is ResNet?

Without ResNet, training a deep model often leads to worse performance due to gradient issues. ResNet solves this with its skip connections, enabling models with hundreds of layers to achieve superior accuracy.

Example: Data Packet Routing in Networks: Think about how data packets are routed over the internet. When a particular server or path becomes congested or slow, routing algorithms identify alternate paths to ensure the packet reaches its destination quickly. Similarly, **ResNet's skip connections** act like alternate routes, allowing crucial information to bypass "congested" or less effective layers in the network, ensuring that the learning process remains efficient and avoids unnecessary delays.

---

3. What is Batch Normalization?

Batch Normalization (BN) normalizes the inputs to a layer within a mini-batch to have a mean of 0 and a variance of 1. This speeds up training and reduces sensitivity to initial weights.

Why it's needed: Neural networks can become unstable if inputs to layers vary too much. BN standardizes these inputs, making training more efficient.

Example (Collaborative Team Alignment): Imagine managing a cross-functional team (data points) for a project. Each team member has different working styles and speeds. To ensure efficient collaboration, you standardize processes and set clear expectations, aligning everyone's pace and output. Similarly, **Batch Normalization** aligns data distributions across layers in a neural network, ensuring that the optimization process progresses smoothly and uniformly, without any one part slowing down or destabilizing the system.

---

4. Connection: Neural Networks → CNN → Residual Block → ResNet.

- **Neural Networks**: General-purpose models for recognizing patterns. Effective for small tasks, like predicting numbers from structured data.
- **CNNs (Convolutional Neural Networks)**: Specialized for image data. They extract hierarchical features like edges, textures, and shapes.
- **Residual Blocks**: Address CNN training issues in deep architectures by learning residuals.
- **ResNet**: Combines CNNs and residual blocks to handle very deep networks effectively.

Real-world example: Imagine teaching a child to recognize objects:

- Neural Networks: Teach shapes (circle, square).
- CNNs: Teach object features (dog fur, cat whiskers).
- Residual Blocks: Help refine errors (e.g., dog vs. wolf confusion).
- ResNet: Combines all steps to confidently identify objects in large, complex datasets.

---

5. What is Depthwise Separable Convolutions, and where is it used?

Depthwise separable convolutions split the convolution into two operations:

1. **Depthwise convolution**: Applies a filter to each channel of the input image separately.
2. **Pointwise convolution**: Combines these results to produce the final output.

Why it's used: Reduces computation and memory usage, making it ideal for lightweight models.

Where used: Popular in lightweight models like MobileNet, optimized for mobile or edge devices.

Real-world example(Manufacturing Automation): Imagine you're running a car assembly line. Instead of having one worker (traditional convolution) perform every task—like installing wheels, painting the body, and adding windows—you assign specialized workers for each task (depthwise convolution) to handle their specific parts independently. Afterward, a supervisor combines these tasks into a complete car (pointwise convolution). This division of labor is more efficient and reduces resource usage, just like **Depthwise Separable Convolutions** improve computational efficiency in neural networks by separating spatial filtering (depthwise) from feature combination (pointwise).

---

6. Differences and similarities between AlexNet, GoogleNet, MobileNet, SENet.

To understand the distinctions and overlaps, let’s explore how each model works and its unique contributions with a comparative table for clarity.

AlexNet

Overview: AlexNet is a deep CNN with 8 layers, including 5 convolutional layers followed by 3 fully connected layers. It was the first model to demonstrate the power of deep learning in large-scale image classification.

Key Components:

- **Convolutional Layers**:
  - Use large filters (11x11 in the first layer, followed by 5x5 and 3x3 in subsequent layers).
  - **Stride of 4** in the first layer reduces spatial dimensions rapidly.
- **ReLU Activation**:
  - Replaces tanh/sigmoid, making training faster by avoiding saturation issues.
- **Local Response Normalization (LRN)**:

    - Normalizes neuron activations for better generalization.
- **Pooling Layers**:
    - Overlapping max-pooling (3x3 filter, stride of 2) to reduce spatial dimensions while retaining key features.
- **Dropout**:
    - Applied in the fully connected layers to prevent overfitting.

Real-world Example: It set the stage for image classification applications, such as detecting cats vs. dogs in photos.

GoogleNet (Inception)

Overview: GoogleNet revolutionized CNNs by introducing **Inception Modules**, allowing multi-scale feature extraction within the same layer. Despite being deeper (22 layers), it was computationally efficient.

Key Components:

1. **Inception Modules**:
    - Combine multiple convolutions (1x1, 3x3, 5x5) and pooling operations.
    - **1x1 Convolutions** are used for dimensionality reduction and feature aggregation.
2. **Auxiliary Classifiers**:
    - Intermediate classifiers help in backpropagation, ensuring gradients flow smoothly in deep networks.
3. **Global Average Pooling**:
    - Replaces fully connected layers to reduce overfitting and model size.

Real-world Example: GoogleNet is used in photo search engines to identify objects like "bicycles in a park."

Unique Feature: Inception modules allow the network to "choose" relevant features at different scales, making it versatile for tasks with varied object sizes.

MobileNet

Overview: MobileNet introduced **Depthwise Separable Convolutions** to create lightweight networks suitable for mobile and edge devices.

Key Components:

- **Depthwise Separable Convolutions**:

    - Decomposes a standard convolution into two parts:
        - **Depthwise Convolution**: Applies one filter per channel.
        - **Pointwise Convolution**: Combines the output using a 1x1 convolution.
    - Reduces computational cost significantly.
- **Width Multiplier**:
    - Reduces the number of filters in each layer to trade-off accuracy for efficiency.
- **Resolution Multiplier**:
    - Scales input image resolution for further computational savings.

Unique Feature: MobileNet is highly modular, allowing customization based on hardware constraints while maintaining competitive accuracy.

Real-world Example: MobileNet powers real-time applications, like object detection in mobile AR apps.

SENet (Squeeze-and-Excitation Network)

Overview: SENet introduced the concept of **Squeeze-and-Excitation (SE) Blocks**, which adaptively recalibrate feature maps to emphasize important channels.

Key Components:

1. **Squeeze Operation**:
    - Applies **global average pooling** to capture global channel-wise information.
2. **Excitation Operation**:
    - Uses two fully connected layers with ReLU and sigmoid activations to compute channel-wise weights.
3. **Recalibration**:
    - Multiplies these weights with the original feature maps, enhancing the focus on crucial channels.

Real-world Example: SENet can be used for medical imaging to emphasize critical features, such as detecting anomalies in X-rays.

Summary

Each model excels in specific scenarios:

- **AlexNet** introduced concepts that revolutionized the field.
- **GoogleNet** handled complex features while saving resources.
- **MobileNet** focused on portability for mobile devices.
- **SENet** refined feature importance with attention mechanisms.

---

7. What is Object Localization and Object Detection?

- **Object Localization**: Identifies one object in an image and marks it with a bounding box.
- **Object Detection**: Finds and marks multiple objects in an image, assigning labels to each.

Real-world example: Imagine a security camera:

- **Localization**: In a security camera feed, a system marks a person in the frame to track their movement.
- **Detection**: Identify and label multiple intruders, distinguishing between people and animals.

Differences:

- Localization handles one object; detection manages multiple.
- Detection combines localization and classification for all detected objects.
- **Real-world example:**
    - A store's security camera uses **localization** to track a single suspicious person, while **detection** identifies all objects in the store, like people, products, and carts.

8. How are object localization and detection performed in computer vision?

1. Object Localization:

Localization focuses on identifying *where* a specific object is in the image.

Steps in Localization:

1. **Feature Extraction:** A convolutional neural network (CNN) processes the image to extract key features like edges, textures, and patterns.
2. **Bounding Box Prediction:** A separate layer in the network predicts the coordinates (x, y, width, height) of a box that best contains the object.
3. **Classification:** Simultaneously, the network determines the class of the object inside the bounding box (e.g., car, cat, etc.).

Real-World Example:

Imagine you are training a drone to locate a single apple in an orchard. Localization tells the drone, "There's an apple at coordinates (100, 150), within a rectangle 50x50 pixels."

2. Object Detection:

Detection goes beyond localization by identifying multiple objects of different classes in the same image and localizing them.

Steps in Detection:

1. **Region Proposals:** Algorithms like Region Proposal Networks (RPNs) suggest multiple regions of interest in the image that are likely to contain objects.
2. **Feature Extraction:** CNN processes each region to extract features for object identification.
3. **Bounding Box Prediction and Classification:** The network predicts the bounding box and class for each detected object.
4. **Non-Max Suppression:** To remove overlapping boxes, only the most confident prediction for each object is kept.

Real-World Example:

Think of a self-driving car that needs to detect and differentiate between pedestrians, vehicles, and traffic signs. Detection would identify:

- A pedestrian at (x1, y1, w1, h1),
- A car at (x2, y2, w2, h2),
- A stop sign at (x3, y3, w3, h3).
