Convolutional Neural Network (CNN)

Study Notes and Explanations with Mathematical Problems

Prepared by Suhi

1. What is CNN?

A **Convolutional Neural Network (CNN)** is a deep learning model that helps computers see and understand images or videos. It is a special type of **Artificial Neural Network (ANN)** designed for image, pattern, and feature recognition.

CNNs are mostly used for:

- Image classification (e.g., cat vs. dog)
- Object detection (finding objects in pictures)
- Face recognition
- Medical image analysis

Why use CNN? Normal neural networks cannot handle image pixels efficiently because images are large and have many values. CNNs reduce image size and find the most important features automatically — such as eyes, mouth, or edges — without human help.

2. How CNN is Different from Normal Neural Networks

- Normal ANN takes 1D flat input data.
- CNN takes 2D or 3D image data (height × width × channels).
- ANN requires more memory and computation time for image data.
- CNN learns spatial relationships between pixels — how nearby pixels are connected — which ANN cannot.

Conclusion: CNNs are faster, smarter, and more accurate for image-based tasks.

3. Structure of a CNN

A CNN has a chain of layers, and each layer has a specific role.

Step	Layer	Work
1	Input Layer	Takes image
2	Convolution Layer	Detects features
3	Activation (ReLU)	Removes negative values
4	Pooling Layer	Makes image smaller
5	Flatten Layer	Turns 2D data into 1D
6	Fully Connected Layer	Combines all features
7	Output Layer	Gives final class or result

4. CNN Layers Explained

(a) Input Layer

The image is represented as numbers (pixel values). Each pixel has a value between 0 and 255. A color image has 3 channels — Red, Green, and Blue (RGB). **Example:** A 32×32 RGB image has shape $32 \times 32 \times 3$.

(b) Convolution Layer

Uses small filters (kernels) like 3×3 or 5×5 that move across the image (stride). Each filter produces a **feature map** showing what it has detected.

Example: If image = $32 \times 32 \times 3$ and filter = $3 \times 3 \times 3$, output may be $30 \times 30 \times 1$.

(c) Activation Layer (ReLU)

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

Keeps useful patterns, ignores negative values. **Example:** Input $[-2, 3, -1, 5]$ → Output $[0, 3, 0, 5]$.

(d) Pooling Layer

Reduces image size and keeps important info. **Example:** Max Pooling 2×2: Input: $\begin{bmatrix} 1 & 3 & 2 & 1 \\ 4 & 6 & 1 & 0 \\ 7 & 2 & 5 & 3 \\ 2 & 1 & 4 & 1 \end{bmatrix}$ → Output: $\begin{bmatrix} 6 & 2 \\ 7 & 5 \end{bmatrix}$

(e) Padding

Adds zeros around image border to maintain size.

(f) Stride

Distance the filter moves: Stride=1 → detailed output; Stride=2 → smaller output.

(g) Flatten Layer

Converts 2D/3D feature maps to 1D vector for dense layers.

(h) Fully Connected Layer

Combines features to make final decision.

(i) Output Layer

Uses Softmax for multi-class classification:

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

5. CNN Training Process

1. Forward Propagation: Image passes through layers.
2. Loss Calculation: Compare predicted vs actual labels.
3. Backward Propagation: Update weights using gradient descent.
4. Repeat until loss is minimized.

6. Important Techniques

- Dropout: Randomly turn off neurons to prevent overfitting.
- Batch Normalization: Balances layer inputs for stable training.
- Data Augmentation: Rotate, flip, crop, zoom images to increase dataset size.
- Transfer Learning: Use pre-trained models (VGG, ResNet) and fine-tune for new tasks.

7. CNN Mathematical Problems and Formulas

7.1 Convolution Operation

Given:

$$\text{Input matrix } (3\times3) = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 3 \\ 2 & 1 & 0 \end{bmatrix}, \quad \text{Filter } (2\times2) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \text{Stride} = 1$$

Output Feature Map:

$$O = \begin{bmatrix} 2 & 5 \\ 1 & 1 \end{bmatrix}$$

7.2 ReLU Activation Function

$$\text{ReLU}(x) = \max(0, x)$$

Example: Input $[-2, 3, -1, 5] \rightarrow$ Output $[0, 3, 0, 5]$

7.3 Softmax Function

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Example: Input $[2, 1, 0] \rightarrow$ Output $[0.66, 0.24, 0.09]$

7.4 Pooling Operation

Max Pooling 2×2: Input: $\begin{bmatrix} 1 & 3 & 2 & 1 \\ 4 & 6 & 1 & 0 \\ 7 & 2 & 5 & 3 \\ 2 & 1 & 4 & 1 \end{bmatrix} \rightarrow$ Output: $\begin{bmatrix} 6 & 2 \\ 7 & 5 \end{bmatrix}$

7.5 Parameter Calculation

$$\text{Parameters} = (f \times f \times \text{input channels} + 1) \times \text{output channels}$$

Example: $f = 3$, input channels = 3, output channels = 16

$$(3 \times 3 \times 3 + 1) \times 16 = 448 \text{ parameters}$$

7.6 Summary of CNN Mathematical Formulas

- Convolution: $O = I * K$
- ReLU: $f(x) = \max(0, x)$
- Softmax: $\text{Softmax}(z_i) = e^{z_i} / \sum_j e^{z_j}$
- Pooling: Take max or average of each region
- Parameter Count: $(f \times f \times \text{input channels} + 1) \times \text{output channels}$

8. Scenario-Based Questions and Answers

Scenario 1: Image Recognition

Question: Which layer detects the round shape of an orange? **Answer:** Convolution layer using filters.

Scenario 2: Overfitting

Question: CNN works well on training but fails on test data. Why? **Answer:** Overfitting; Solution: Dropout or Data Augmentation.

Scenario 3: Large Images

Question: How to reduce image size with limited memory? **Answer:** Use Pooling layers (Max Pooling).

Scenario 4: Face Detection

Question: What features do early/later layers detect? **Answer:** Early: edges/lines; Later: complex features like eyes or face.

Scenario 5: Multi-Class Classification

Question: CNN predicts 10 animals. Which activation for final layer? **Answer:** Softmax, which outputs probabilities for each class.

– End of Study Note –