Understanding Neural Network Architectures

Neural networks come in different architectures; each designed for specific types of problems. Understanding when to use each architecture is crucial for effective machine learning.

1. Multi-Layer Perceptrons (MLPs)

MLPs, also called fully connected networks or feedforward neural networks, are the simplest deep learning architecture. Every neuron in one layer connects to every neuron in the next layer.

Architecture

- Input layer: Receives raw features
- Hidden layers: Process information through weighted connections
- Output layer: Produces predictions
- Activation functions: ReLU, Sigmoid, Tanh between layers

Use

- Tabular data (spreadsheet-like data)
- Simple classification or regression tasks
- When data has no spatial or sequential structure

Limitations

- Cannot capture spatial patterns (like in images)
- Poor at handling sequential data
- Many parameters (computationally expensive for large inputs)

2. Convolutional Neural Networks (CNNs)

CNNs are specialized for processing grid-like data, particularly images. They use convolutional layers that detect local patterns and features through sliding filters.

Key Components

- Convolutional layers: Apply filters to detect features (edges, textures, shapes)
- Pooling layers: Reduce spatial dimensions, retain important information
- Fully connected layers: Final classification after feature extraction
- Parameter sharing: Same filter used across entire image (fewer parameters)

Use

- Image classification, object detection, segmentation
- Any grid-structured data with spatial relationships
- When you need translation invariance (feature detected anywhere in image)

Advantages Over MLPs

- Captures spatial hierarchies
- Fewer parameters than MLPs for images

3. Transformers

Transformers are the architecture behind modern NLP breakthroughs (GPT, BERT, ChatGPT). They use self-attention mechanisms to process sequences in parallel, unlike sequential RNNs.

Key Innovation: Attention Mechanism

The attention mechanism allows the model to focus on relevant parts of the input when processing each element. For text, this means understanding which words are most important for understanding other words.

Key Components

- Self-attention layers: Learn relationships between all positions in sequence
- Positional encoding: Adds position information (no inherent sequence order)
- Multi-head attention: Multiple attention mechanisms run in parallel
- Feed-forward networks: Process each position independently

Use

- Natural language processing (translation, summarization, question answering)
- Sequential data where long-range dependencies matter
- Tasks requiring understanding of context and relationships
- Now expanding to vision (Vision Transformers) and other domains

Architecture Comparison

Aspect	MLP	CNN	Transformer
Best For	Tabular data	Images, spatial data	Text, sequences
Parameters	Many (all-to-all)	Fewer (sharing)	Very many (large)
Key Feature	Fully connected	Convolution filters	Self-attention
Processing	Layer by layer	Hierarchical features	Parallel attention

Forward and Backward Pass

Neural networks learn through two key processes: the forward pass (making predictions) and the backward pass (learning from errors).

Forward Pass: Making Predictions

Step 1: Input: Data; enters the network
Step 2: Layer Computation; Each layer: output = activation (weights × input + bias)
Step 3: Propagation; Output of one layer becomes input to next
Step 4: Prediction; Final layer produces prediction
Step 5: Loss Calculation; Compare prediction to actual answer

Backward Pass: Learning from Mistakes

Also called backpropagation, this is where the network learns by adjusting weights to reduce errors.

Step 1: Compute Loss Gradient
Step 2: Backpropagate Gradients
Step 3: Update Weights; Adjust weights in direction that reduces loss
weight_new = weight_old - learning rate × gradient

Step 4: Repeat; Process flows backward through all layers
Step 5: Iterate; Repeat forward and backward passes many times

The Learning Cycle

1. Forward Pass: Make prediction, calculate loss
2. Backward Pass: Calculate gradients, update weights
3. Repeat: Thousands of iterations
4. Result: Weights that minimize loss = learned model

Key Insight: The Chain Rule

Backpropagation works through the chain rule of calculus. Since the network is a chain of functions (layer 1, layer 2, layer 3, output), we can calculate how changing any weight affects the final output by multiplying gradients backward through the chain.

This is why it's called backpropagation; we propagate the gradient backward from output to input.

Summary

Each architecture serves different purposes:

- MLPs: Simple, general-purpose, best for tabular data
- CNNs: Specialized for images and spatial data
- Transformers: State-of-the-art for sequences and NLP

All three learn through the same fundamental process: forward pass to make predictions, backward pass to learn from errors. Understanding both the architectures and the learning process is essential for effective deep learning practice.