Deep Learning Algorithms – Study Notes

1. Introduction

Deep learning refers to a class of machine learning methods that employ **artificial neural networks with multiple layers** to model high-level abstractions in data. Over the past decade, several specialized architectures have emerged, each tailored to specific data types and problem domains. These algorithms differ in network design, training techniques, and applications.

2. Feedforward Neural Networks (FNNs)

- **Definition**: The simplest form of deep networks in which information flows strictly forward from input to output, with no cycles.
- **Key Components**:
  - Input layer → Hidden layers → Output layer

- Activation functions (ReLU, Sigmoid, Tanh) provide non-linearity.

- **Applications**: Basic classification and regression tasks.

Limitation: Poor performance on sequential or structured data such as text or time series.

3. Convolutional Neural Networks (CNNs)

- **Definition**: Networks specialized for processing grid-like data, especially images.
- **Core Mechanism**: Convolutional filters (kernels) slide across input data to detect features such as edges, textures, or patterns.
- **Key Features**:
    - Convolutional layers extract spatial features.
    - Pooling layers reduce dimensionality and retain important information.

    - Fully connected layers produce final predictions.
- **Applications**:
    - Image classification (e.g., ImageNet challenge).
    - Object detection (e.g., self-driving cars).
    - Medical imaging (tumor detection).

Important Algorithms: LeNet, AlexNet, VGG, ResNet, EfficientNet.

4. Recurrent Neural Networks (RNNs)

- **Definition**: Networks designed for sequential data by maintaining internal “memory” of past inputs.
- **Key Feature**: Feedback loops that pass information from one time step to the next.
- **Applications**:
    - Natural language processing (NLP).

    - Time-series forecasting (stock prediction, weather).
    - Speech recognition.

Limitation: Struggle with long-term dependencies due to vanishing/exploding gradients.

5. Long Short-Term Memory Networks (LSTMs)

- **Definition**: A refined RNN variant that addresses long-term dependency problems.
- **Key Innovation**: **Memory cells** controlled by input, forget, and output gates.
- **Applications**:
    - Machine translation.
    - Text generation.
    - Handwriting recognition.

6. Gated Recurrent Units (GRUs)

- **Definition**: A simplified alternative to LSTMs with fewer gates, making them computationally more efficient.
- **Use Case**: Similar to LSTMs but preferred when data is abundant and model efficiency is important.

7. Generative Adversarial Networks (GANs)

- **Definition**: A framework involving two networks — a **generator** that creates synthetic data and a **discriminator** that evaluates authenticity.
- **Training Mechanism**: Adversarial process where the generator improves by "fooling" the discriminator.
- **Applications**:
    - Image synthesis (e.g., deepfakes).
    - Data augmentation.
    - Art and design generation.

8. Autoencoders

- **Definition**: Neural networks that learn compressed representations (encodings) of data.
- **Structure**:
  - Encoder: Reduces dimensionality.
  - Bottleneck: Latent representation.
  - Decoder: Reconstructs original data.
- **Applications**:
  - Data compression.
  - Denoising.
  - Anomaly detection.

Variants: Variational Autoencoders (VAEs) add probabilistic modeling, often used in generative tasks.

9. Transformers

- **Definition**: Architectures relying on **self-**

attention mechanisms rather than recurrence or convolutions.

- **Key Idea**: Attention allows the model to weigh the relevance of different input tokens simultaneously.
- **Applications**:
    - Natural language processing (BERT, GPT).
    - Multimodal tasks (text-to-image models).
- **Impact**: Dominant architecture in modern AI research and commercial applications.

10. Deep Reinforcement Learning Algorithms

- **Definition**: Combination of deep learning with reinforcement learning.
- **Algorithms**:
    - Deep Q-Networks (DQN).

- Policy Gradient Methods.
- Actor-Critic Models.

- **Applications**:
  - Game playing (e.g., AlphaGo).
  - Robotics.
  - Autonomous navigation.

11. Hybrid and Emerging Architectures

- **Capsule Networks**: Designed to capture spatial hierarchies beyond CNNs.
- **Graph Neural Networks (GNNs)**: Operate on graph-structured data, e.g., social networks, molecules.
- **Neural Ordinary Differential Equations (ODEs)**: Continuous-time modeling of hidden states.

12. Comparative Overview

Algorithm	Data Type	Key Strength	Typical Applications
FNN	Structured/ tabular	Simplicity	Classification, regression
CNN	Images/ video	Spatial feature extraction	Image recognition
RNN	Sequences	Temporal modeling	NLP, time-series
LSTM/ GRU	Sequences	Long-term dependencies	Language, speech
GAN	Any	Synthetic data generation	Images, augmentation
Autoencoder	Any	Compression, denoising	Anomaly detection
Transformer	Sequences/ text	Contextual understanding	NLP, multimodal AI

Deep RL	Interactive environments	Decision-making	Games, robotics

Review Questions

1. Compare the functioning of CNNs and RNNs. Why is each better suited for different data types?
2. Explain how LSTMs address the vanishing gradient problem inherent in RNNs.
3. Discuss the adversarial training process in GANs.
4. How do transformers differ from RNNs in handling long-range dependencies?
5. Identify a real-world problem and justify which deep learning algorithm would be most suitable.

Suggested Readings

- Goodfellow, I., Bengio, Y., & Courville,

A. (2016). *Deep Learning*. MIT Press.

- Chollet, F. (2021). *Deep Learning with Python*. Manning Publications.
- Vaswani, A., et al. (2017). *Attention is All You Need*. NeurIPS.
- Schmidhuber, J. (2015). *Deep Learning in Neural Networks: An Overview*. Neural Networks.