Deep Learning Course – L 7 - Detailed Notes

🔢 Activation Functions in Neural Networks

Activation functions introduce **non-linearity** into neural networks, allowing them to learn complex patterns. They play a crucial role in how a network **learns, adapts, and performs** across various tasks. In this document, we explore popular activation functions—**ReLU**, **Sigmoid**, **Tanh**, and their variants—along with their **advantages, disadvantages, and use cases**.

---

1. ⚡ ReLU (Rectified Linear Unit)

➤ What is ReLU?

- ReLU returns the **input if positive**, and **zero if negative**:

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

✅ Advantages

- **Computationally efficient** (simple comparison).
- **Avoids vanishing gradient** problem.
- Works well in deep networks.

❌ Disadvantages

- **Dying ReLU Problem**: Neurons can stop learning if inputs are always negative (output is always zero).

---

2. 🔁 Sigmoid Activation Function

➤ What is Sigmoid?

- Outputs values between **0 and 1**:

  σ(x)=11+e−x\sigma(x) = \frac{1}{1 + e^{-x}}σ(x)=1+e−x1

✅ Advantages

- Ideal for **binary classification** tasks.
- Outputs can be interpreted as **probabilities**.

❌ Disadvantages

- **Vanishing gradient** problem in deep networks.
- **Not zero-centered**.
- **Computationally expensive** (exponentiation).

---

3. 🔄 Tanh Activation Function

➤ What is Tanh?

- Similar to sigmoid but ranges from **-1 to 1**:

  $$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

✅ Advantages

- **Zero-centered** output.
- Better for cases where **positive and negative signals** are needed.

❌ Disadvantages

- Suffers from the **vanishing gradient** problem.
- Requires exponentiation (costly computation).

---

4. 🔧 Variants of ReLU

ReLU has been modified into several variants to overcome its limitations, especially the **dying neuron** issue.

4.1. 💧 Leaky ReLU

f(x)={xif x>0αxotherwisef(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases}f(x)={xαxif x>0otherwise

Where α\alphaα is typically 0.01.

✅ *Pros:*

- Mitigates **dying ReLU** problem.

❌ *Cons:*

- Still produces **small negative outputs**, which may not be ideal in some contexts.

4.2. 🧠 Parametric ReLU (PReLU)

- Same as Leaky ReLU, but α\alphaα is **learned** during training.

✅ *Pros:*

- **Adaptable** to the data.

❌ *Cons:*

- Introduces **additional learnable parameters** and complexity.

4.3. 📈 Exponential Linear Unit (ELU)

f(x)={xif x>0α(ex−1)otherwisef(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{otherwise} \end{cases}f(x)={xα(ex−1)if x>0otherwise

✅ *Pros:*

- **Zero-centered** output.
- Reduces the **dying neuron** issue.

❌ *Cons:*

- **Computationally expensive** due to exponentiation.

✅ Final Thoughts

Activation	Best Used In	Pros	Cons
ReLU	Hidden Layers (General DL tasks)	Fast, effective, avoids vanishing gradients	Dying neurons
Sigmoid	Binary Classification (Output Layer)	Probabilistic outputs	Vanishing gradients, not zero-centered
Tanh	Binary Classification (Hidden Layers)	Zero-centered	Vanishing gradients
Leaky ReLU	Deep Networks	Prevents dying neurons	Small negative output
PReLU	Advanced Models	Learnable flexibility	More complex
ELU	Complex Architectures	Smooth, zero-centered	Computationally expensive

Remember: The choice of activation function depends on the **problem**, **data**, and **network depth**. There's no one-size-fits-all solution.