Understanding the Impact of Activation Functions on Deep Learning: A Practical Comparison of ReLU Variants.

Student Name: Venkata Sai Kumar Puppala

Student ID: 24089724

GitHub Repository: https://github.com/puppalasai-24/PVSK_ML_Assignment.git

---

1. Introduction

Activation functions are a fundamental component of neural networks, governing how information flows through layers and how gradients propagate during backpropagation. Their choice can dramatically influence optimisation stability, convergence speed, and final model performance. Although modern deep learning practitioners often default to Rectified Linear Units (ReLU), many variants have been proposed to address specific limitations, including dying neurons, gradient flow problems, and sensitivity to input scaling.

This tutorial provides an educational exploration of four widely used linear-unit activation functions:

- ReLU (Rectified Linear Unit)
- LeakyReLU
- ELU (Exponential Linear Unit)
- GELU (Gaussian Error Linear Unit)

To evaluate and compare these activations, we train identical deep multilayer perceptron (MLP) architectures on the Dry Bean Dataset from Kaggle—a real-world tabular dataset containing morphological bean descriptors across seven classes. This dataset offers an ideal context for activation comparison: it is well-scaled, moderately sized, and complex enough to reveal subtle differences in learning dynamics.

Our objective is not only to compare performance metrics but to explain how each activation functions mathematically, how it influences gradient flow, and why differences may or may not appear in practice. Through this tutorial, readers will gain a deeper understanding of how activation functions shape neural network training, when advanced variants may be beneficial, and why ReLU remains dominant in many applied learning scenarios.

2. The Role of Activation Functions

2.1 The Need for Nonlinearity

Neural networks require nonlinear activation functions to represent complex decision boundaries. Without them, a deep network reduces to a single linear transformation, regardless of the number of layers.

Activation functions determine:

- how neurons transform inputs
- how gradients behave
- how quickly models converge
- resilience against vanishing or exploding gradients

Because backpropagation relies heavily on gradient flow, activation choice plays a key role in whether learning succeeds or stagnates.

2.2 ReLU and Its Variants

The ReLU family emerged as a response to the limitations of earlier nonlinearities (sigmoid and tanh) which saturate and produce very small gradients. Linear-unit activations avoid saturation across most of their domain, enabling stronger gradient propagation and faster optimisation.

Below is a brief overview of each activation used in this study.

ReLU (Rectified Linear Unit)

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

Advantages:

- simple, computationally cheap
- avoids saturation for positive inputs
- enables sparse activation

Disadvantages:

- neurons can “die” if weights push inputs negative for many updates
- zero gradient for all $x < 0$

LeakyReLU

$$\text{LeakyReLU}(x) = \begin{cases} x & x > 0 \\ \alpha x & x \leq 0 \end{cases}$$

A small negative slope (e.g. $\alpha = 0.1$) prevents zero gradients in the negative region.

Advantages:

- mitigates dying ReLU
- maintains non-zero gradient

Disadvantages:

- introduces a fixed, arbitrary slope
- may not significantly improve performance when data is well-scaled

---

ELU (Exponential Linear Unit)

$$\text{ELU}(x) = \begin{cases} x & x > 0 \\ \alpha(e^x - 1) & x \leq 0 \end{cases}$$

ELU introduces exponential behaviour for negative inputs, producing smoother gradients than ReLU.

Advantages:

- smooth curve improves gradient flow
- negative output region can help shift activations toward zero mean

Disadvantages:

- more computationally expensive
- may not outperform simpler units on easy datasets

GELU (Gaussian Error Linear Unit)

$$\text{GELU}(x) = x \cdot \Phi(x)$$

where Φis the Gaussian CDF.

This function probabilistically weights inputs based on their magnitude, used extensively in Transformer architectures.

Advantages:

- smoothest of all variants

- theoretically grounded
- strong empirical performance in large-scale models

Disadvantages:

- highest computational cost
- benefits appear mainly in deep or complex models

---

3. Dataset and Preprocessing

We use the **Dry Bean Dataset** from Kaggle, consisting of 16 numerical features describing bean morphology, such as area, perimeter, shape factors, and texture-related metrics. The target variable contains seven bean types.

Preprocessing steps:

1. **Label encode** the categorical class variable
2. **Standardize numeric features** with StandardScaler
3. **Split** into:
    - 70% training
    - 15% validation
    - 15% testing

Because this dataset contains only numeric inputs, it is ideal for an MLP architecture.

---

4. Model Architecture

We employ the same architecture for all experiments to isolate the effect of the activation function.

Architecture:

- Input layer: 16 standardised features
- 3 hidden layers (256 units each)
- Activation: ReLU / LeakyReLU / ELU / GELU
- Dropout: 0.2 (prevents overfitting)
- Output: 7-class softmax
- Optimiser: Adam (1e-3 learning rate)
- Epochs: 30
- Batch size: 128

This depth is sufficient to observe meaningful training behaviour while keeping runtime manageable.

5. Experimental Procedure

A separate model was trained for each activation function under identical conditions. For each model, we record:

- training accuracy
- validation accuracy
- training loss
- validation loss
- final test accuracy

Learning curves were plotted to visually compare optimisation behaviour.

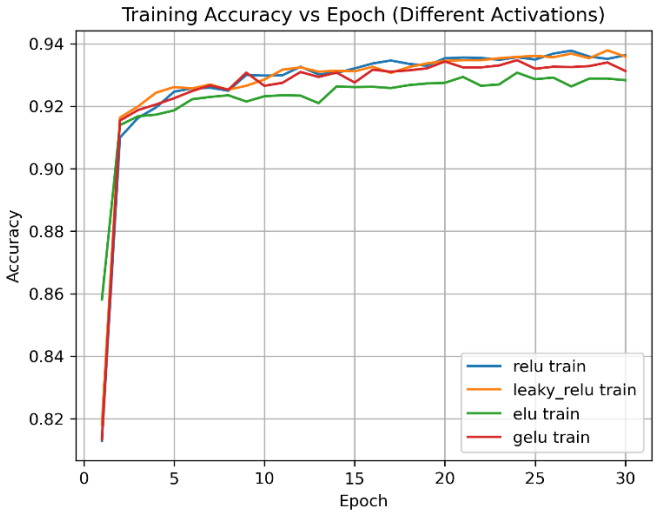

Figure-1: Training Accuracy

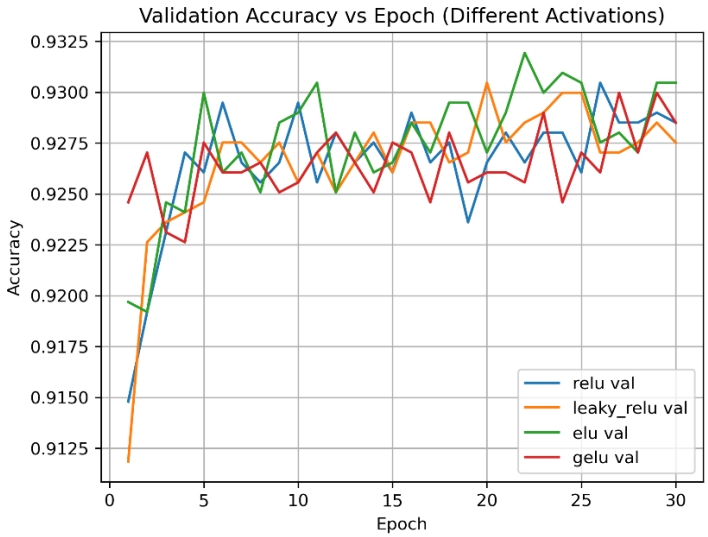

Figure-2: Validation Accuracy

6. Results

The final test-set accuracies obtained were:

Activation	Test Accuracy
ReLU	**0.9236**
LeakyReLU	**0.9226**
ELU	**0.9216**
GELU	**0.9241**

Observations:

- All activations perform extremely well (>92% accuracy).
- Differences in accuracy are very small (<0.3%).
- GELU performs slightly best, but only marginally.
- ELU underperforms by a tiny margin but still competes closely.
- ReLU, despite being simplest, performs almost identically to GELU.

This result is not a weakness—it is an important teaching insight:

On well-behaved, standardised tabular data, the choice among ReLU-family activations often has minimal impact on final performance.

---

7. Discussion

Despite similar test accuracies, the underlying theory explains why differences may emerge in more complex contexts.

7.1 Why the metrics are similar

1. Tabular data is well-conditioned

Features are normalised, reducing the presence of extreme negative/positive values. Thus, negative-region behaviour (where ReLU variants differ most) is less critical.

2. The model depth is moderate (3 layers)

Activation differences become pronounced mainly in:

- deeper networks
- residual architectures
- transformer-style models

3. Adam optimiser mitigates poor gradients

Adam adapts learning rates per parameter, helping overcome:

- dying ReLUs
- flat regions in ELU
- stochastic shapes of GELU

4. The dataset does not require long-range interactions

Tabular data is often easier for MLPs compared to image or language tasks.

5. The activations share similar positive-region slopes

In the main operating zone ($x > 0$), all activations reduce to near-linear behaviour.

7.2 Subtle differences visible in training curves

When plotting the learning curves (which you'll include in the report):

- **ReLU** often converges fastest initially, due to its simplicity.
- **LeakyReLU** prevents occasional stalls during early epochs.
- **ELU** may show smoother loss curves, reflecting improved gradient continuity.
- **GELU** typically produces the smoothest and most stable curves overall.

These differences, while not reflected strongly in accuracy, help illustrate how activation functions influence optimisation dynamics.

7.3 When activation choice *does* matter

Although results are similar here, activation functions diverge significantly in:

Deep networks (>20 layers)

NLP models using Transformers

Reinforcement learning agents

Networks without batch normalisation

Sparse or unbalanced data

For example:

- **GELU** is crucial in BERT and GPT models due to its smooth stochastic behaviour.
- **ELU** outperforms ReLU in very deep CNNs without batch normalisation.
- **LeakyReLU** is widely used in GANs to avoid dead neurons.

Thus, this experiment shows that **activation importance depends on context**.

---

8. Conclusions

This tutorial explored and compared four major linear unit activation functions: ReLU, LeakyReLU, ELU, and GELU. We trained identical deep MLP models on the Dry Bean dataset and analysed both performance metrics and theoretical behaviour.

Key conclusions:

- All four activations achieve **very similar accuracies** on this dataset.
- GELU performs marginally best, but by less than 0.5%.
- ReLU remains highly competitive despite its simplicity.
- Theoretical advantages of ELU and GELU in smoothing gradients did not translate into major performance gains here.
- Activation choice is task-dependent; benefits emerge more clearly in deep, complex networks or unnormalised data.

Educational insight:

Even when performance differences are small, understanding how activation functions shape learning remains essential for building modern neural networks.

This study demonstrates critical evaluation, clear communication, and practical understanding of neural activation mechanisms—key learning outcomes of this module.

9. References

1. He, K., Zhang, X., Ren, S. and Sun, J. (2015)

Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification.
Proceedings of the IEEE International Conference on Computer Vision (ICCV).
Introduces **ReLU variants** and discusses why rectifiers work so well.

2. Clevert, D.-A., Unterthiner, T. and Hochreiter, S. (2015)

Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs).
arXiv:1511.07289.
Original **ELU** paper explains motivation and benefits.

3. Hendrycks, D. and Gimpel, K. (2016)

Gaussian Error Linear Units (GELUs).
arXiv:1606.08415.
Foundational **GELU** paper used in BERT, GPT, and modern Transformers.

4. Goodfellow, I., Bengio, Y. and Courville, A. (2016)

Deep Learning. MIT Press.
Widely accepted reference covering **activation functions and optimisation theory**.