# Research Paper Analysis: Gaussian Error Linear Units (GELUs)

Alex Shrestha

CS545 Machine Learning

## 1 Introduction

For this research paper analysis, I decided to focus on activation functions due to their substantial importance in neural networks. Activations functions introduce the much needed non-linearity and thus are fundamental to neural networks. Without non-linear activations, a very deep layered neural network could be represented by a single layer. The paper I will analyze is Gaussian Error Linear Units (GELUs) [1] by Dan Hendrycks and Kevin Gimpel. The paper's goal is to introduce a new activation function called the Gaussian Error Linear Unit or GELU for short. The paper emprically evalutates how the GELU performs against standard non-linear activations ReLU and ELU by running experiments on the MNIST dataset, Twitter POS tagging dataset, TIMIT dataset, and the CIFAR-10/100 dataset. One of the core reasons that led the authors to developing a new activation function is that they wanted to address the existing limitations of existing activations function like the sigmoid and ReLU. The sigmoid activations were useful in the early days of networks but as networks became deeper, the performance of the sigmoid diminished. Thus ReLU came about and it's "non-smooth gating" provided faster and better convergence. One issue with ReLU was its inability to output negative values and its less probabilistic nature. Thus the GELU was born, incorporating a probabilistic approach which will be explored next.

# 2 Analysis

The GELU activation function is $x\Phi(x)$ where $\Phi(x)$ is the standard Gaussian cumulative distribution function. The GELU nonlinearity weights inputs by their value, rather than gating inputs by their sign as ReLUs do. The goal is still to have a form of dropout involved. It is important to note that ReLU and dropout accomplish similar things (yielding a neuron's output) but in different ways: ReLU does this deterministically using the sign of an input whereas dropout does this randomly.

GELU multiplies the input by zero or one randomly by multiplying the neuron input $x$ by $m \sim \text{Bernoulli}(\Phi(x)) = P(X \leq x),\ X \sim \mathcal{N}(0,1)$ . We can notice that as $x$ decreases, the probability of the input being dropped inversely increases. However, the goal is to have a "deterministic decision". What this means is that an input should not have multiple different outputs, thus a deterministic decision leads to a more stable output since slight changes in the input will not cause a drastic change. To achieve this goal, the expected value of the transformation on an input $x$ is taken: we have $\Phi(x) \times 1x + (1 - \Phi(x)) \times 0x = x\Phi(x)$. This equation can be approximated leading to:

$$\text{GELU}(x) = xP(X \leq x) = x\Phi(x) = x \cdot \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$$

where the error function is used to compute the cumulative distribution function of the Gaussian.

The authors evaluated GELU against ReLU and ELU on MNIST classification, MNIST autoencoding, Twitter part-of-speech tagging, TIMIT frame recoginition, and CIFAR-10/100 datasets. On the MNIST classification dataset, they were able to show that classifiers using GELU are more robust to noised inputs which they showed by adding uniform noise to each input data. They were also able to show that taking the median result over multiple runs, GELU tends to have the lowest median training log loss both with and without dropout. When training a deep autoencoder on MNIST they saw

similar results: mean squared loss was lowest using the GELU compared to the other non-linear activation functions. Similar results were shown in both the TIMIT dataset and the Twitter POS dataset as well. When working with more intricate architecture such as shallow and deep convolutional neural networks, the GELU activation function still outperformed the other non-linear functions. The authors used the CIFAR-10/100 dataset and still saw GELU obtaining the lowest median error rate compared to ReLU and ELU.

# 3 Conclusion

In the discussion section, the authors note that even though GELU performed better, it still had some similarities to the ReLU and ELU. They note that the GELU can be viewed as a way to smooth ReLU similar to how the sigmoid smoothed the binary threshold activations. That being said GELU also has important differentiations from previous non-linear functions. GELU is a non-convex, non-monotonic, and non-linear in the positive domain and it exhibits curvature at all points. This may allow the GELU to better approximate more complicated functions than the ReLU. Also due to GELU being an expected value, it also has a probabilistic interpretation that the ReLU does not. The authors also make a point to note that the ReLUs gate the input depending on the sign of the input whereas GELU weights its input "depending upon how much greater it is than other inputs." What this means is that the $\Phi(x)$ term weights the input by the probability that a standard random variable is less than or equal to $x$: the $\Phi(x)$ is adjusting how much of $x$ should be passed through based on its relative magnitude in the normal distribution. Based on the experiments shown in this paper, it has been shown that GELU is a viable alternative to other non-linearities due to its consistent performance.

# 4 Questions

Some questions I have for the authors:

1. What in your prior research or experience led you to believe that the current set of non-linear activation functions could be improved upon thus leading to the creation of the GELU?

2. What inspired the mathematical formulation of the GELU: how did you come up with GELU specifically?

3. Do you believe that the experiments you performed were diverse enough to choose GELU as an alternative activation function in any situation that requires a neural network architecture?

4. In what settings does the GELU not perform as well as the other functions?

5. What are some weaknesses of the GELU that can be improved upon?

# References

[1] Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016.