UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
DEPARTAMENTO DE ESTATÍSTICA E INFORMÁTICA
PROGRAMA DE PÓS–GRADUAÇÃO EM INFORMÁTICA APLICADA

JOÃO HENRIQUE DE MEDEIROS DELGADO

NEURAL NETWORKS WITH ADAPTIVE AND TRAINABLE ACTIVATION FUNCTIONS: A PERFORMANCE ANALYSIS

RECIFE – PE

2024

JOÃO HENRIQUE DE MEDEIROS DELGADO

NEURAL NETWORKS WITH ADAPTIVE AND TRAINABLE ACTIVATION FUNCTIONS: A PERFORMANCE ANALYSIS

Dissertation submitted to the Coordenação do Programa de Pós-Graduação em Informática Aplicada of the Departamento de Estatística e Informática - DEINFO - Universidade Federal Rural de Pernambuco, as part of the requirements necessary to obtain a Master's degree.

ADVISOR: Tiago Alessandro Espínola Ferreira

RECIFE – PE

2024

Dados Internacionais de Catalogação na Publicação
Sistema Integrado de Bibliotecas da UFRPE
Bibliotecário(a): Auxiliadora Cunha – CRB-4 1134

D352n Delgado, João Henrique de Medeiros.
Neural networks with adaptive and trainable activation functions: a performance analysis / João Henrique de Medeiros Delgado. – Recife, 2024.
147 f.; il.

Orientador(a): Tiago Alessandro Espínola Ferreira.

Dissertação (Mestrado) – Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, BR-PE, 2025.

Inclui referências e apêndice(s).

1. Função de Ativação. 2. Redes Neurais. 3. Função de ativação treinável. 4. Neurônio Global-Local 5. Recursos locais e globais. I. Ferreira, Tiago Alessandro Espínola, orient. II. Título

CDD 004

JOÃO HENRIQUE DE MEDEIROS DELGADO

NEURAL NETWORKS WITH ADAPTIVE AND TRAINABLE ACTIVATION FUNCTIONS: A PERFORMANCE ANALYSIS

Dissertation submitted to the Coordenação do Programa de Pós-Graduação em Informática Aplicada of the Departamento de Estatística e Informática - DEINFO - Universidade Federal Rural de Pernambuco, as part of the requirements necessary to obtain a Master's degree.

Approved on: 16 de dezembro de 2024.

EXAMINATION BOARD

Tiago Alessandro Espínola Ferreira (Orientador)
Universidade Federal Rural de Pernambuco (UFRPE)
Departamento de Estatística e Informática (DEINFO)

Kerolly Kedma Felix do Nascimento
Universidade Regional do Cariri (URCA)
Departamento de Estatística e Informática (DEINFO/UFRPE)

Paulo Salgado Gomes de Mattos Neto
Universidade Federal de Pernambuco (UFPE)
Centro de Informática (CIn)

To my father (in memory), to my son, wife and mother.

Acknowledgements

To my parents for the values and good education they passed on to me, shaping me to be the person I am today.

To my son, the greatest gift of my life, for all the happy moments he gave me.

To my wife for all the support and help during this difficult period.

To my uncles Izabel and Lothar, for having contributed significantly to my education throughout my life.

To my advisor Prof. Tiago Ferreira for all the teaching, understanding and support.

To my friend Paulo Oliveira for his constant help and companionship.

To my master's degree colleagues and everyone in the research group I am part of, IFROG, for all the cooperation and learning, especially Henrique Torres, also for all the help at the beginning of my work.

To the Physics Department of UFRPE and the National Laboratory for Scientific Computing (LNCC) for providing the computational infrastructure necessary to carry out this work.

"Joy lies in the fight, in the attempt, in the suffering involved, not in the victory itself"

(Mahatma Gandhi)

Abstract

The activation function plays an important role in an artificial neural network by performing a non-linear transformation of the input data, which enables the network to learn complex patterns. Despite being a well-established field of study, there is still much interest in discovering new activation functions that can accelerate training and increase the performance of neural networks. Recent studies have demonstrated the potential of activation functions that have weights and are able to adapt to the input data during the training process. Also, there has been growing interest in periodic functions that are able to learn global patterns, improving the generalization of models. Following these steps, this work aims to evaluate the performance of the activation function called Global-Local Neuron (GLN) in order to investigate its performance in untested types of problems, such as image compression and classification and multiclass classification. Recently proposed, this function differs from most trainable functions, which only adjust their shape or combine functions with similar characteristics. This one combines two distinct and complementary activation functions, one with local properties and the other with global properties. It also proposes and evaluates other variations of the GLN function using activation functions such as the ReLU and Mish functions as alternatives for the local component and the recently proposed Growing Cosine Unit as an alternative to the global component of the GLN. Datasets with varied characteristics and widely used as benchmarks by the scientific community were used. The network training used the k-fold cross-validation method to ensure a robust evaluation and, for each activation function evaluated, four models were created with two different values for the batch size and two methods of initializing the weights so as not to restrict the study to specific parameters that may benefit a specific function. At least 30 training rounds were also performed for each model in order to perform a statistical significance analysis using the KS-Test method and verify whether the differences in the results found are relevant. The results demonstrated that the GLN is capable of adapting and can find, with the functions employed in its composition, an efficient combination, being, in general, statistically similar to the best results of the individual functions and in some cases superior. The results brought further evidence to the fact that there is no ideal activation function for all types of problems and situations, and that GLN, and its variations, can be very interesting alternatives as a function in different scenarios.

Keywords: Activation Function, Neural Networks, Trainable Activation Function, Global-Local Neuron, Local and Global Features

Resumo

A função de ativação desempenha um importante papel em uma rede neural artificial ao fazer uma transformação não linear dos dados de entrada, o que possibilita a rede ser capaz de aprender padrões complexos. Apesar de ser um campo de estudo bem consolidado, ainda existe muito interesse em descobrir novas funções de ativação que possam acelerar o treinamento e aumentar a performance das redes neurais. Estudos recentes tem demonstrado os potenciais das funções de ativação que possuem pesos e são capazes de se adaptarem a partir dos dados de entrada durante o processo de trenamento. Também, tem havido crescente interesse nas funções periódicas que são capazes de aprender padrões globais, melhorando a generalização dos modelos. Seguindo esses passos, este trabalho tem o objetivo de avaliar o desempenho da função de ativação chamada Global-Local Neuron (GLN) a fim de investigar o seu desempenho em tipos de problemas não testados, como de compactação e classificação de imagens e classificação multiclasse. Recentemente proposta, essa função difere da maioria das funções treináveis, que apenas ajustam sua forma ou combinam funções com características semelhantes. Esta, combina duas funções de ativação distintas e que se complementam, uma com propriedade local e outra com propriedade global. Também, propõe, e avalia, outras variações da função GLN utilizando em sua composição funções de ativação como as funções ReLU e Mish como alternativas para a componente local e a função recém proposta cosseno crescente como alternativa à componente global do GLN. Foram empregados datasets com características variadas e muito utilizados como benchmarking pela comunidade científica. O treinamento da rede usou o método de validação cruzada k-fold para garantir uma avaliação robusta e, para cada função de ativação avaliada, foram criados quatro modelos com dois valores diferentes para o tamanho do batch e dois métodos de inicialização dos pesos de modo a não restringir o estudo a parâmetros específicos que podem beneficiar alguma função específica. Também foram realizadas pelo menos 30 rodadas de treinamento para cada modelo, de modo a realizar uma análise da significância estatística pelo método KS-Teste e comprovar se as diferenças nos resultados encontrados são relevantes. Os resultados demostraram que o GLN é capaz de se adaptar e consegue encontrar, com as funções empregadas em sua composição, uma combinação eficiente, sendo, de modo geral, estatisticamente similares aos melhores resultados das funções individuais e em alguns casos superior. Os resultados trouxeram mais evidências para o fato que não há uma

função de ativação ideal para todos os tipos de problemas e situações, e que o GLN, e suas variações, podem ser alternativas muito interessantes como função em diversos cenários.

Palavras-chave: Função de Ativação, Redes Neurais, Função de ativação treinável, Neurônio Global-Local, Recursos locais e globais

List of Figures

List of Tables

List of symbols

AE	*AutoEncoders*
ANN	*Artificial Neuron Network*
CIFAR	*Canadian Institute For Advanced Research*
CNN	*Convolutional neural Network*
GCU	*Growing Cosine Unit*
GLN	*Global-local neuron*
KS	*Kolmogorov-Smirnov*
LSTM	*Long Short-Term Memory*
MLP	*Multilayer Perceptron*
MNIST	*Modified National Institute of Standards and Technology*
RMSProp	*Root Mean Square Propagation*
RBMs	*Restricted Boltzmann Machines*
ReLU	*Rectified Linear Unit*
RNN	*Recurrent neural networks*
SGD	*Stochastic Gradient Descent*
SVM	*Support Vector Machine*
VGG	*Visual Geometry Group*
WRN	*Wide Residual Networks*

Contents

1 Introduction

1.1 Justification

We cannot talk about neural networks without talking about activation functions, as they play a fundamental role in these networks' capacity. They bring the fundamental nonlinearity so that these networks learn complex patterns from input data. Therefore, notable advances in neural networks are also intrinsically linked to the history of the development and innovations of activation functions.

Initially, the perceptron algorithm, developed in the 1950s, a precursor to neural networks, applied the Heaveside Step function as the activation function. The Heaveside Stepe function is a binary activation function that produces the output 1 or 0 depending on the threshold reached (JAGTAP; KARNIADAKIS, 2022).

At the end of the 1980s, with the return of interest in neural networks, due to the great advancement brought by the backpropagation algorithm, and the creation of the first MLP neural network, the sigmoidal function gained popularity. This non-linear function returns a continuous output between 0 and 1. The sigmoidal function employment was a significant advance for neural networks. It allowed the neural networks to model probabilities another other things (NWANKPA et al., 2018), solving problems beyond simple binary classifications.

Another function that became very common as an activation function in neural networks was the hyperbolic tangent, replacing the sigmoid function, as it provided better performance (GOODFELLOW et al., 2016). It is a scaled sigmoid function, but centered on zero, which made the output distribution more symmetric and was therefore often preferred (BUDUMA; LOCASCIO, 2017). These activation functions, despite being widely used, are saturated (XU et al., 2016), leading to problems of vanishing gradient and gradient explosion, especially in very deep networks, thus making it difficult for the network to converge during training.

Research has also studied the use of alternative activation functions such as the use of periodic functions (MCCAUGHAN, 1997; SOPENA et al., 1999), such as the sine trigonometric function, which is non-monotonic, different from the traditional sigmoid function and Tanh, monotone. These studies began to demonstrate the potential of these

oscillatory functions by demonstrating that they could outperform traditional networks in several aspects in different tasks.

In 2010, a new activation function was introduced in the article "Rectified Linear Units Improve Restricted Boltzmann Machines" (NAIR; HINTON, 2010). This article details an approach to improve the performance of Restricted Boltzmann Machines (RBMs) by replacing binary hidden units with rectified linear units (ReLUs). Since then, ReLU has become the most widely used activation function (GOODFELLOW et al., 2016).

Despite the wide use of the ReLus functions, mainly in Convolutional Neural Networks, many variants were created to solve its original limitations. Like the "Dead Neuron" (when the neuron can no longer learn due to the negative value as input), non-zero-centeredness, and sensitivity to exploding gradients. Variants such as Leaky ReLU (XU et al., 2015), Prametric ReLU (PReLU) (HE et al., 2015a), and Exponential Linear Unit (ELU) (CLEVERT et al., 2015) were developed by introducing slopes or other adaptations to avoid returning a zero value in the negative part of the function.

The academic community's interest in activation functions has returned very strongly since then, largely trying to resolve the limitations and disadvantages of classical activation functions and the importance that the activation function has on network performance. New proposals have emerged proposing that the combination of existing activation functions can help solve these problems, such as Sigmoid and Relu (MASTROMICHALAKIS, 2021), and also these with ELU (KILIçARSLAN; CELIK, 2021).

Another aspect that emerged was that of trainable activation functions. Although it is not a new idea, as of 2018, there has been a growing interest in these activation functions that have the ability to learn through the network (APICELLA et al., 2021). These functions have adjustable parameters, as well as the weights and biases of the network, enabling dynamic adaptation depending on the input data. This capability has brought excellent results in terms of performance, convergence, and generalization of models (LIAO, 2020; APICELLA et al., 2019; SUBRAMANIAN et al., ; HE et al., 2015a).

Generally, the same activation function is employed for all neurons in the same layer of the network. However, proposals have emerged to allow that, in addition to having a trainable activation function, each neuron also has its own activation function adjusted from the input set (DUSHKOFF; PTUCHA, 2016; ERTUğRUL, 2018), allowing a more

subtle response of each neuron.

After a long period, non-monotonic functions regained interest in the literature. Since traditional functions cannot adequately model periodic signals, especially outside the training data range (ZIYIN et al., 2020; ROBIN et al., 2022), the use of periodic functions becomes useful in that it is possible to better extrapolate data. Furthermore, they are ideal for capturing repetitive patterns in data.

Despite these benefits, they are much more complex to train (PARASCANDOLO et al., 2017; WU, 2022), due to the presence of multiple local minima and because they are very sensitive to network initialization conditions (RAMASINGHE; LUCEY, 2022).

Still, several recent articles have explored these oscillatory functions: (MALLICK et al., 2020) presents a study with a periodic perceptron with the results showing that it outperforms the multilayer perceptron in the tested tasks (XOR and parity); (CHAN et al., 2018) presents the SinP[N] activation function that brought faster convergence and accuracy similar to those obtained in tests using the ReLU and Swish functions; (MEHTA et al., 2021) details a new approach in the functional representation of signals using neural networks with periodic activations and a modulation structure; (SITZMANN et al., 2020) explores periodic activation functions in neural networks for implicit neural representations; (NOEL et al., 2021) introduces a new oscillatory activation function called the Growing Cosine Unit that allows a single neuron to learn the XOR function.

Building on the ideas of trainable and adaptive functions and the benefits of periodic functions, a new activation function called the Global Local Neuron (GLN) (FERREIRA et al., 2021) emerged in 2021. This activation function combines local and global learning properties into one single neuron in addition to enabling dynamic adjustment of weights between these different properties, allowing for a finer adaptation of its characteristics during the training process.

The GLN was extensively tested in two types of problems, regression and solving differential equations, and generally had a superior performance or statistically similar to those obtained by the functions that compose it, sinusoidal function or hyperbolic tangent function.

This excellent initial performance, in the scope of these problems, indicates that GLN can be efficient in capturing patterns with local and global characteristics that are properties of the most diverse types of problems.

1.2 Research Problem

Does the use of trainable activation functions, which combine two activation functions that are different in their characteristics, one global and the other local, bring benefits for better accuracy, lower loss values, and speed of convergence in artificial neural networks?

1.3 Objectives

1.3.1 General Objectives

The general objective of this research is to evaluate the performance of artificial neural networks that use trainable activation functions with two components (fixed activation functions), distinct in their characteristics, specifically the GLN function and its variations, compared to the use of these functions alone. In solving multiclass and image classification problems.

1.3.2 Specific objectives

- Select datasets that are relevant to the problems presented; have a variation of contexts, sizes, and levels of complexity; that are accessible; and used by the academic community.
- Define and implement trainable activation functions that are a variation of GLN
- Create neural network models for each type of problem with identical configurations for each type of neural network.
- Train neural network models for each activation function evaluated using the same data set and partitioning.
- Define evaluation metrics that allow a detailed analysis of the performance of networks in terms of generalization capacity and precision.
- Perform statistical analyses to validate whether the observed differences are significant.

1.4 Chapter Preview

This work is divided into the following sections:

- Chapter 2 presents the fundamental concepts relating to the development of work. Gives a brief introduction to artificial neural networks and the main networks used in this work. It also presents the concept of activation functions and discusses the activation functions used in this work.
- Chapter 3 presents the methodology carried out including the research project, selection of samples, how data was collected, and finally the analysis of this data.
- Chapter 4 presents the results obtained in each of the two phases of this work.
- Finally, in Chapter 5, the conclusions of this study and suggestions for possible future work are presented

2 Literature review

This chapter will present the fundamental concepts related to the work. A brief introduction to the emergence of artificial neural networks and the initial developments that led to the creation of these powerful tools will be provided. Subsequently, the main networks used in this study will be discussed. Finally, the activation functions, which are crucial for the operation of these networks and the main focus of this work, will be addressed.

2.1 Artificial neural networks

Artificial neural networks are computational models inspired by the functioning of the human brain, aiming to simulate the interaction between neurons. One of the first works that laid the foundation for the advancement of these networks was by McCulloch and Walter Pitts (MCCULLOCH; PITTS, 2021) in 1943, where they proposed a simplified model of an artificial neuron using electronic logic gates to represent a neuron. They demonstrated that propositional or Boolean logic could address brain nerve activity. Fifteen years later, Frank Rosenblatt developed the concept of the perceptron (ROSENBLATT, 1958), a very simple algorithm capable of learning to classify patterns, which it was a precursor to artificial neural networks.

After a long period with few advancements, artificial neural networks regained popularity in the 1980s and 1990s with the emergence of the backpropagation training algorithm (RUMELHART et al., 1986). This algorithm, which uses error backpropagation techniques to adjust the weights of the connections between neurons, enabled the development of the Multi-layer Perceptron (MLP), an architecture with multiple layers of neurons capable of learning to classify patterns in non-linearly separable sets. Various types of problems were addressed using this new architecture. Since then, several others have been developed, such as Recurrent Neural Network (RNN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), AutoEncoders (AE), and many others.

2.1.1 Multi-Layer Perceptron (MLP)

Also known as a universal approximator (DU; SWAMY, 2013), Multi-Layer Perceptron (MLP) networks are formed by a simple structure composed of three main types of layers: the input layer, the hidden layers (at least one), and the output layer, as shown in Figure 1. Each layer is a grouping of artificial neurons that are not connected but are connected to all the neurons of the previous layer (HAYKIN, 2000). These neurons have the role of receiving the input signal or information, performing processing, and passing it forward as an output value.

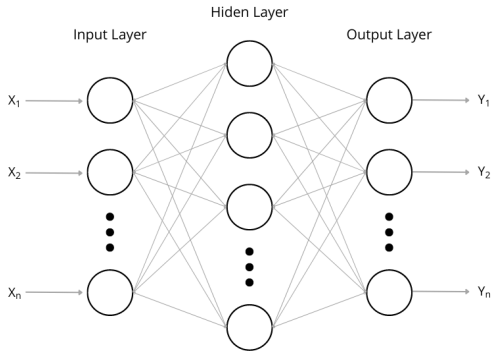


Figure 1 – Schematic representation of an Multi-Layer Perceptron (MLP).

The functioning of this neural network architecture is intrinsically based on the backpropagation method for updating weights. Unlike the perceptron, where weight updates occur simply and directly because there is only one layer,

$$w_{\text{new}} = w_{\text{old}} + \eta \times (y - \hat{y}) \times x, \tag{2.1}$$

here, w_{new} and w_{old} are the new and old values of the weight, respectively. η is the learning rate. y is the true class value. $\hat{y}$ is the predicted class. x is the input.

In the MLP the update of weights is more complex due to the presence of one or more hidden layers,

$$w_{\text{new}} = w_{\text{old}} - \eta \times \frac{\partial E}{\partial w}, \tag{2.2}$$

E is the cost or loss function. $\frac{\partial E}{\partial w}$ is the gradient of the cost function with respect to the weight.

One of the fundamental parameters for updating weights is the learning rate, η (Equation (2.1)). This rate determines the size of the step taken in the direction opposite to the gradient, directly affecting the speed and efficiency with which the model learns (GOODFELLOW et al., 2016). Lower rate values make the convergence smoother and more stable but slow down the process and make it more likely to get stuck in local minima. On the contrary, higher rates make the model more unstable, possibly preventing convergence to a global minimum, despite increasing the learning speed.

Another very important parameter is the initial value of the weights, w. There are different weight initialization techniques, and the choice of technique affects the final performance of the trained model and directly influences the speed at which the network converges to a minimum of the loss function (NARKHEDE et al., 2021).

Initially, the importance of weight initialization was not understood, and weights were randomly initialized. However, more modern techniques such as Glorot/Xavier Initialization (GLOROT; BENGIO, 2010), which maintains the variance of gradients across layers depending on the number of inputs and outputs of each layer, and He Initialization (HE et al., 2015b), with an initialization technique focused on the RELU function that maintains the variance depending on the number of inputs in the layer, have emerged.

2.1.2 Autoencoders

An Autoencoder is a type of unsupervised learning algorithm that attempts to learn the identity function by reproducing the same input data at the output of the neural network (NG, 2010). At first, this may not seem interesting or useful, but the goal of this type of network is ultimately to learn a compressed representation of the input data, used to create a copy at the output with minimal information loss (GOODFELLOW et al., 2016; HINTON; SALAKHUTDINOV, 2006). To achieve this, it features a structure where the intermediate layers have fewer neurons than the first and last layers.

Figure 2 shows a generic architecture, similar to a horizontal hourglass, where there is an initial layer with X neurons, a final layer with the same number of neurons as

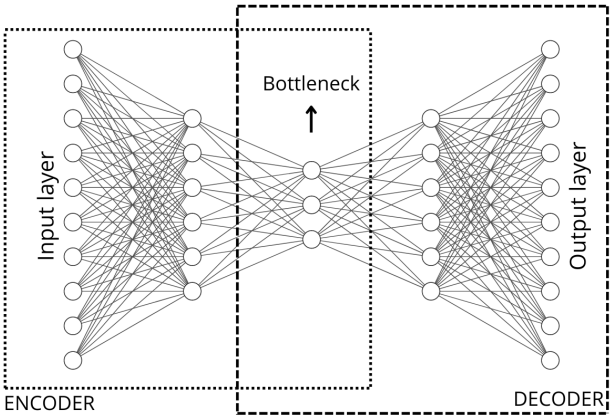


Figure 2 – Schematic representation of an Autoencoder, highlighting the encoder and decoder submodels.

the input layer, and between these layers, one or more hidden intermediate layers with a central layer known as the bottleneck, which has the fewest neurons. An Autoencoder model comprises two submodels, the encoder and the decoder (GOODFELLOW et al., 2016). The encoder submodel, located between the initial layer and the bottleneck layer, is responsible for creating the compressed representation of the input data from the correlations existing among these data (NG, 2010). Between the bottleneck layer and the final layer, the decoder submodel is responsible for reconstructing this representation in the output data.

2.1.3 Convolutional Neural Networks (CNN)

Convolutional neural networks (CNN) are a type of deep neural networks mainly aimed at the area of computer vision. They are used very successfully in facial recognition and object detection applications, important in robot and autonomous car technologies, as well as in other areas such as natural language understanding and speech recognition (LECUN et al., 2015). The main characteristic of CNNs networks is the use of a mathematical operation called convolution.

This convolution operation is a specialized type of linear operation, used in at least one of the layers, replacing general matrix multiplication (GOODFELLOW et al., 2016). It works by sliding trainable filters, also known as kernels, over the input image, see Figure 3, and at each position, performing a pointwise product between the filter and the corresponding part of the image. The results are summed to produce a single value on the feature map. This process is repeated for each filter, thus creating a complete set of

feature maps.

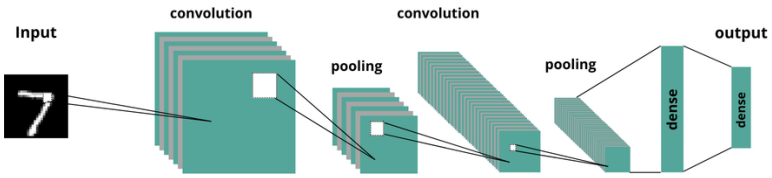


Figure 3 – *Schematic representation of an Convolutional Neural Networks (CNN).*

This convolution process brings one of the great advantages of CNNs: the number of parameters is drastically reduced since each neuron is connected only to a small region of the input data (GOODFELLOW et al., 2016). Furthermore, the filters, or weights, are applied to different parts of the input set and thus learn features that are location-independent. This reduces the complexity of the model and generalizes well to new samples outside the training set.

2.2 Activation Functions

As one of the fundamental elements that constitute a neuron in an artificial neural network (HAYKIN, 2000), the activation functions are responsible for determining how the weighted sum of input weights, along with bias, is transformed to define the output value of the neuron.

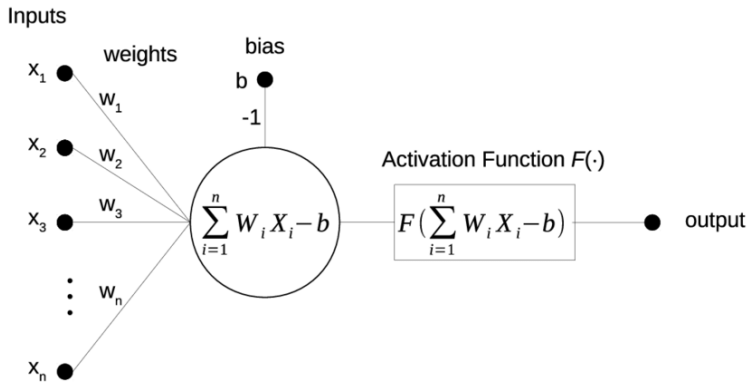


Figure 4 – *Artificial Neuron* scheme.

A conventional neuron, see Figure 4, operates by multiplying a trainable weight (*weights*), to each input value (*inputs*), summing all these products, and finally adding a bias *bias* to this sum to apply the result as an argument for an activation function:

$$output = f\left(\sum_{i=1}^{n} W_i \times X_i - b\right). \tag{2.3}$$

Without activation functions in the neurons of an artificial neural network, the output signals would merely be a linear function of the input data. Thus, the network would lack the ability to learn complex patterns. By using nonlinear activation functions, the network can learn complex patterns.

Activation functions can be divided into two categories: Fixed-shape activation functions and trainable activation functions (APICELLA et al., 2021). The difference lies in whether or not they have parameters that can be modified during network training. Fixed functions do not have parameters, while trainable functions have parameters that alter the function according to the data. Next, examples of activation functions from these two classes used in this work will be presented.

2.2.1 Fixed activation functions

These are predefined activation functions that do not change their shape as the network is trained (APICELLA et al., 2021).

2.2.1.1 Tanh

The hyperbolic tangent function, or tanh, is a scaled version of the sigmoid function (GOODFELLOW et al., 2016), shaped like an S, see Figure 5, smooth, and continuously differentiable. Historically, it became preferred over the sigmoid function because it performed better. Nowadays, it is still widely used in artificial neural networks, particularly in feedforward and recurrent neural networks.

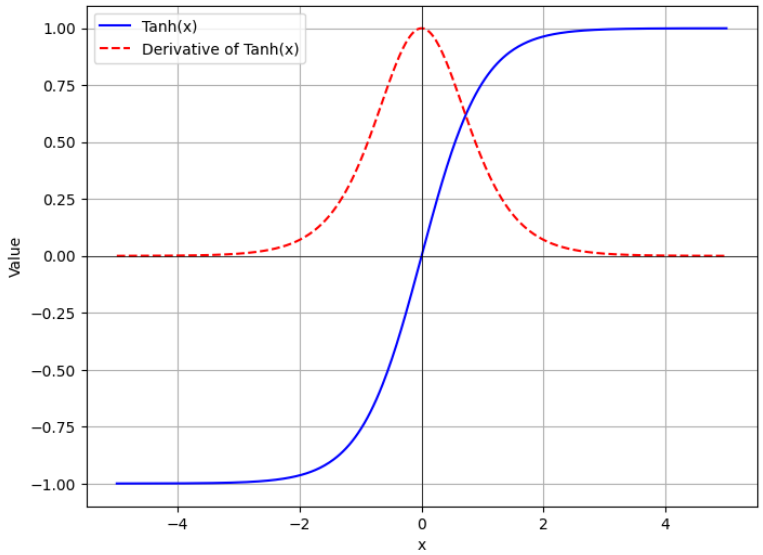


Figure 5 – *Graphical representation of the hyperbolic tangent function in blue and its derivative in red*

This function returns values ranging from -1 to 1, providing a zero-centered output distribution, which results in better convergence during the training phase. Mathematically, it is defined as:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{2.4}$$

This can also be expressed as:

$$\tanh(x) = 2\sigma(2x) - 1, \tag{2.5}$$

where (σ) is the sigmoid function 2.6,

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.6}$$

2.2.1.2 ReLU

Being one of the activation functions with the lowest computational cost and addressing one of the biggest problems in deep neural networks, the vanishing gradient, the ReLU (Rectified Linear Unit) function has stood out since its proposal in 2010 by

Nair and Hinton when they used it in Restricted Boltzmann Machines (RBMs) (NAIR; HINTON, 2010).

The ReLU function is defined as:

$$ReLU(x) = max(0, x), \tag{2.7}$$

where x is the input to the activation function. The graph of the Relu function is presented in Figure 18.

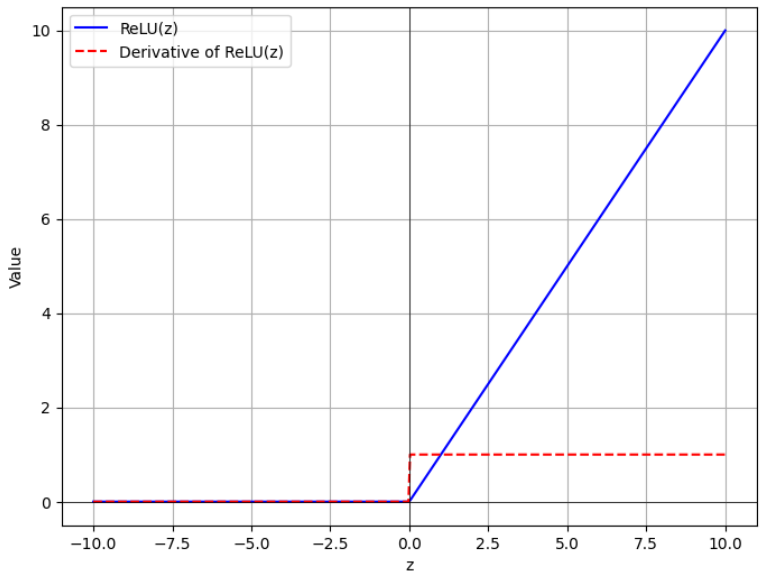


Figure 6 – *Graphical representation of the ReLU function in blue and its derivative in red*

The neuron is activated only if the input is positive; if it is negative, the output is zero. This leads to the problem of the ”dead neuron”, which occurs when a neuron consistently produces a zero output, rendering it incapable of learning during the training period. Several activation functions have been derived from ReLU with the primary intent of addressing this issue.

One of the main activation functions derived from Relu was the Leaky ReLU function (XU et al., 2015). This brings a component α,

$$LeakyReLU(x) = max(\alpha x, x) \tag{2.8}$$

a generally very small positive coefficient (*e.g.*, 0.01), when x is negative. Ensuring that the gradient is never completely zero.

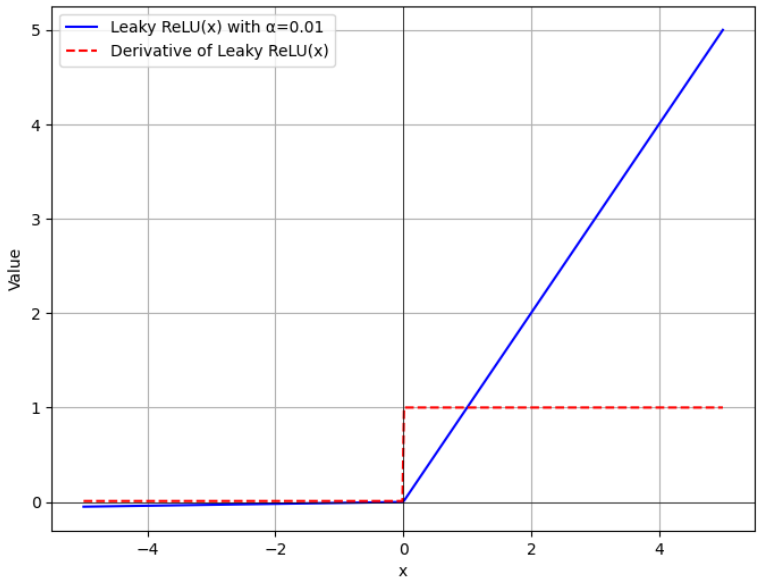


Figure 7 – *Leaky ReLU*

2.2.1.3 Mish

The Mish activation function is one of the functions created with the aim of solving ReLU problems. It has the properties of non-monoticity, smoothness, continuity, and self-regularization (MISRA, 2020; GUSTINELI, 2022). It is mathematically defined by:

$$f(x) = x \cdot \tanh(\text{softplus}(x)) = x \cdot \tanh(\ln(1 + e^x)), \tag{2.9}$$

where softplus(x) is a smooth approximation of the ReLU function, given by:

$$\text{softplus}(x) = \ln(1 + e^x). \tag{2.10}$$

Very similar to ReLU in that it has no upper limit and, for most negative input values, has a derivative of 0, it differs from it mainly in the smooth and continuous transition of its derivative, and in the ability to generate negative values, as can be seen in Figure 8.

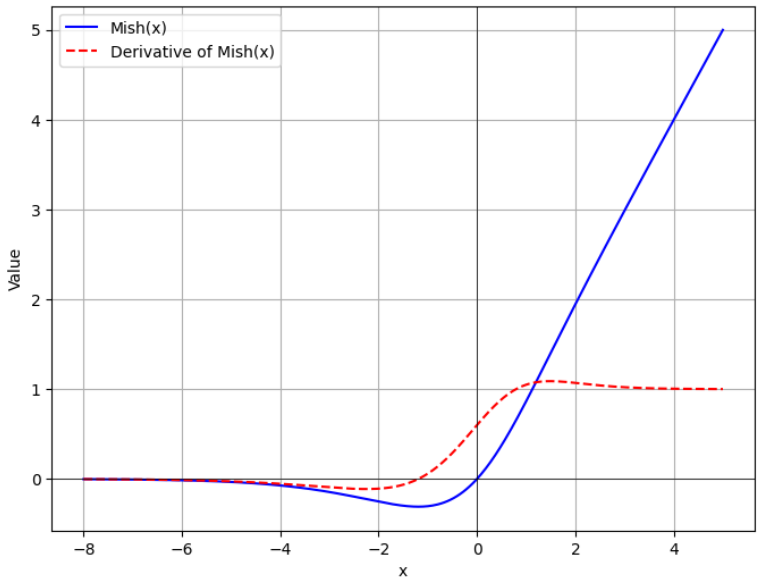


Figure 8 – *Graphical representation of the Mish function in blue and its derivative in red*

Despite having a higher computation cost, it has demonstrated superior performance among the main activation functions for image classification problems (MISRA, 2020; GUSTINELI, 2022).

2.2.1.4 Softmax

Softmax is an activation function mainly used in the last layer of neural networks, especially to deal with multi-class classification problems. It is often used in conjunction with the cross-entropy loss function, which quantifies the difference between predicted probabilities and the true distribution (GOODFELLOW et al., 2016).

Unlike other activation functions that are applied to a single neuron, Softmax is applied to an output vector, converting this vector of real values into a vector of probabilities. The probabilities of each value are proportional to the relative scale of each value in the vector and all output values sum to 1 (DU; SWAMY, 2013).

The mathematical formula for the Softmax function for a given vector (Z) of real numbers is:

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Here, e represents a mathematical constant approximately equal to 2.71828, z_i is

the input value for the current class being normalized, and K is the number of classes in the multiclass classifier.

The denominator $\sum_{j=1}^{K} e^{z_j}$ is the sum of the exponential transformations of all input values in the vector.

2.2.1.5 Sin

The sine activation function is a periodic function derived from the mathematical trigonometric function. It is not an activation function that is normally used in traditional neural networks and receives little attention from the scientific community (PARASCANDOLO et al., 2017), although it has shown promise in several applications, such as signal representation (MEHTA et al., 2021).

The low popularity of this function, and other similar sinusoidal functions, is because training becomes more complex and with great difficulty in converging to a global minimum because of its periodic nature, with the presence of infinite local minima (PARASCANDOLO et al., 2017).

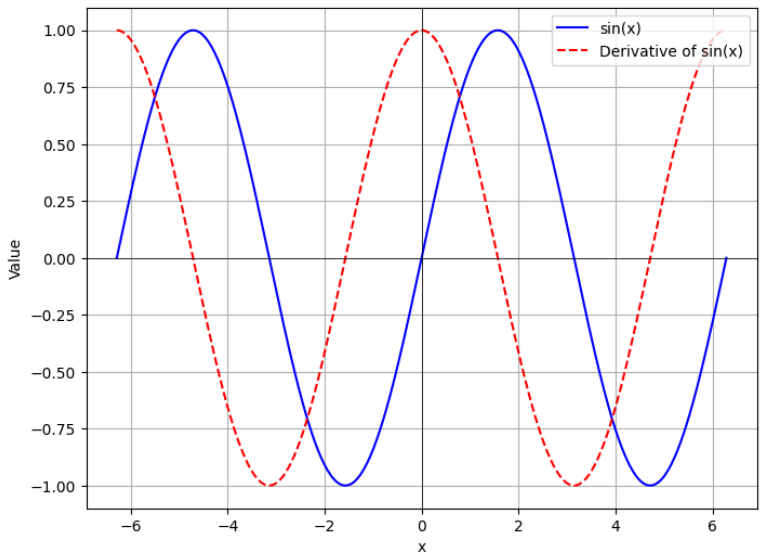


Figure 9 – *Graphical representation of the sin function in blue and its derivative in red*

2.2.1.6 Growing Cosine Unit (GCU)

The Growing Cosine Unit (GCU) activation function is also a periodic function based on the trigonometric cosine function (NOEL et al., 2021). It is defined mathematically

by:

$$C(x) = x \cdot \cos(x). \tag{2.11}$$

Except for isolated points, the derivative is non-zero throughout its domain, as can be seen in Figure 10. This fact minimizes the vanishing gradient problem (NOEL et al., 2021; GUSTINELI, 2022)

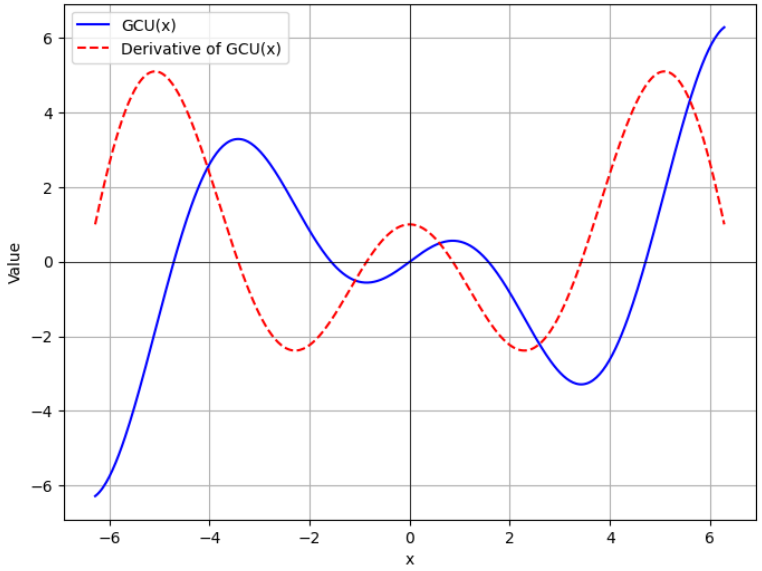


Figure 10 – *Graphical representation of the Growing Cosine Unit (GCU) in blue and its derivative in red*

In its experimental results, it obtained good results compared to already recognized activation functions such as ReLU and Mish for evaluated image classification problems.

2.2.2 Trainable activation functions

These are activation functions that have adjustable parameters during the training process (APICELLA et al., 2021). This allows these functions to change their own form as they adapt to the data set, aiming to optimize the network's performance.

These activation functions can be classified into two categories: parameterized standard functions, which modify the classic activation functions by adding parameters but maintaining their form, and functions based on ensemble methods, which combine

several base (non-trainable) activation functions in a linear fashion (APICELLA et al., 2021).

The idea of creating these functions that learn from the network began in the 1990s, involving the parameterization of common activation functions, such as the sigmoid and the hyperbolic tangent, with trainable parameters.

Following the trend of rectified activation functions to solve the limitations of ReLU, Parametric ReLU (PReLU) (HE et al., 2015a) emerged in 2015, introducing a trainable parameter for the negative slope.

Also motivated to overcome the problems of ReLU, Swish (RAMACHANDRAN et al., 2017) emerged in 2017, a non-monotone and smooth function, defined as:

$$f(x) = x \cdot \sigma(x), \tag{2.12}$$

and which can be reparameterized to include a trainable parameter, β,

$$f(x;\beta) = \beta x \cdot \sigma(x) \tag{2.13}$$

When β is adjusted during training, it allows the activation function to behave similarly to ReLU when β is large, or as a linear function when β is zero.

Motivated by the idea that fixed activation functions may not be ideal for all tasks or data sets, (LIAO, 2020) proposes two approaches, one as an activation function as a Fourier series, where the parameters of the series (amplitude, frequency and coefficients) are trainable, and another as a linear combination of simple activation functions.

Reviving the hyperbolic tangent function, TanhSoft (BISWAS et al., 2021) emerged in 2021, using variations of it with trainable parameters to control the shape of the curve.

Recent research such as the Learnable Extended Activation Function (LEAF) (BODYANSKIY; KOSTIUK, 2023), 2023, which combines the properties of squashing functions and rectifier units, APALU (Adaptive Piecewise Approximated Activation Linear Unit) (SUBRAMANIAN et al.,), which combines characteristics of linear and nonlinear functions and ErfReLU (RAJANAND; SINGH, 2024) which combines the advantages of ReLU (Rectified Linear Unit) and the error function (erf), still as an alternative to solve problems such as "dying ReLU", continue to explore this theme.

2.2.2.1 Global-local neuron (GLN)

The GLN (Global-Local Neuron) activation function is an adaptive and trainable function composed of two mathematical functions that complement each other, one global and one local (FERREIRA et al., 2021), given by:

$$F(x) = (\alpha.global(x) + (1 - \alpha).local(x)) - Bias, \tag{2.14}$$

where α and $Bias$ are trainable weights and $0 \leq \alpha \leq 1$. At the implementation level,

$$\alpha = sig(z),\ z \in \mathbb{R} \tag{2.15}$$

where $sig(\cdot)$ is the logistic sigmoid function 2.16, given by:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.16}$$

The GLN differs from the traditional model as it has two distinct activation functions in its characteristics. These functions take as an argument the output produced by the linear combination of weights, input data, and bias in a neuron and apply them to a trainable weight α of complementary form as we see in Figure 11)

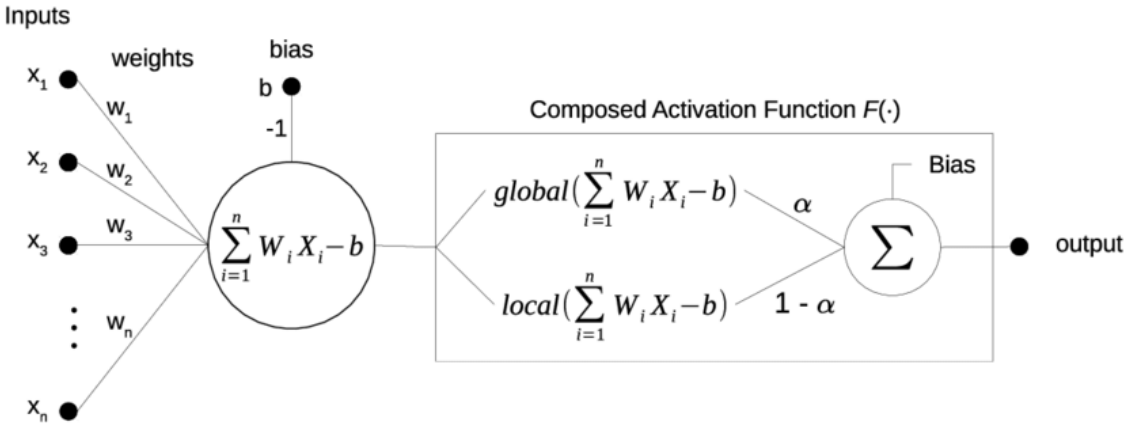


Figure 11 – *Global-Local Neuron*(GLN) scheme (FERREIRA et al., 2021).

In this way, the network itself, during the training process, adjusts the relative importance of each of these components (the weight α) for the final definition of the activation function. Specifically, the same GLN activation function is used at each layer

of the network. Therefore, since α and $Bias$ are components of the activation function, these parameters are the same for each layer.

By using activation functions with unique characteristics in the composition of the GLN, such as the sine trigonometric function, and using the hyperbolic tangent function, it is possible to identify two different characteristics, global (or general) and local (or specific) respectively, or the combination of both with different degrees of importance.

3 Methodology

3.1 Introduction

This research aims to evaluate the impacts of trainable and adaptable activation functions, especially the Global-Local Neuron (GLN) and its variations, on the performance of artificial neural networks in new types of problems. As already demonstrated in other works, for regression and differential equation problems, by combining two activation functions with distinct features, global and local, the robustness of the models in generalizing and recognizing complex patterns is improved. We hypothesize that this same approach also brings the same benefits to multiclass and image classification problems.

GLN was originally designed using the combination of the sine and hyperbolic tangent functions, which capture the general and specific features, respectively. In this work, while maintaining the characteristics that define GLN, we develop four new variations with different compositions of activation functions in its structure: GLN-ReLU, GLN-Mish, GLN-Growing Cosine Unit (GLN-GCU) and GLN-Growing Cosine Unit ReLU (GLN-GCU-R).

The methodology was divided into two phases. In the first phase, we investigated the performance of the original GLN in comparison with the standard hyperbolic sine and tangent functions, training a Multi-layer Perceptron (MLP) Autoencoder neural network to generate compressed images using the CIFAR10 dataset. These compressed images, with reduced dimensionality, were used as input data for two classifiers, Wide Residual Networks (WRN) (ZAGORUYKO; KOMODAKIS, 2016) and Support Vector Machines (SVMs) (AWAD; KHANNA, 2015), verifying the accuracy in the classification of these compressed images, and thus evaluating their ability to extract features by the autoencoder networks (GOODFELLOW et al., 2016).

In the second phase we had access to the Santos Dumont supercomputer at LNCC, greatly increasing our computational resources. We then expanded the study to include 4 variations of GLN, along with their constituent activation functions, ReLU, Mish, GCU, Sine, Tanh. They were evaluated on a wide range of different datasets, Breast Cancer, Iris, Wine, and Yeast for multiclass classification problems, and MNIST, CIFAR-10 and Lung Colon Cancer for image classification problems.

For further investigation, we varied the batch size and weight initialization hyperparameters in both models used, MLP and CNN, to solve the respective problems under analysis.

These neural network architectures employed were chosen because they are the classic models, widely recognized and used for each type of specific problem addressed in this research, without aiming to achieve the best absolute performance.

3.2 Research design

This research adopts an experimental quantitative approach to evaluate the impact of GLN-type activation functions on the performance of artificial neural networks. This design, which requires variable control and the use of meaningful metrics, was chosen because it allows obtaining precise numerical data such as accuracy, loss, and convergence speed, commonly used in research in this area, and allows a rigorous statistical comparison between the activation functions in different configurations of neural network models.

To this end, several data sets were selected, which will be presented in the following section, based on the types of problems addressed, image compression, multiclass classification, and image classification. The preprocessing step of these data involved the Min-Max normalization technique and the K-fold cross-validation method to divide the data set into training, validation, and test sets.

After preprocessing, we designed and implemented specific artificial neural network models to solve the proposed problems. Including Multi-layer Perceptrons (MLP), Convolutional Neural Networks (CNN), and Wide Residual Networks (WRN). Then, to obtain the data for analysis, several rounds of training and testing were performed by changing the activation functions, and in the second phase of the experiment, the batch size and weight initialization hyperparameters.

Given the stochastic nature of artificial neural networks, several experiments were repeated to ensure reliable results. Finally, the performance of the neural networks, measured through accuracy, loss value, and convergence speed, was compared between the different configurations of the models and their activation functions using the Kolmogorov-Smirnov statistical technique (KS-Test).

3.3 Sample selection

This research used several public datasets, presented in the following subsections, widely used in the scientific community and considered benchmarks in machine learning and neural network research. These databases have a wide variety of domains and different levels of complexity.

For multiclass classification problems, the Breast Cancer, Iris, Wine and Yeast databases found in the UC Irvine machine learning repository were used. These datasets offer a wide variety of samples, bringing several challenges in terms of separability, class balance, and attribute correlation.

For image classification problems, the MNIST, CIFAR-10, and Lung Colon Cancer image databases were used, which provided different levels of complexity in terms of image resolution, class distribution, and domain-specific features.

The cross-validation method was employed on all datasets using the sklearn library (PEDREGOSA et al., 2011). We applied the method with K = 5, that is, we divided each dataset into five subgroups or folds. One of the folds was used as the test set, while the other four were used as the training and validation sets. This process was repeated 5 times, with each fold being used as the test set once. Performance metrics were calculated by averaging the results of each process, providing more reliable estimates of the model's generalization ability.

3.3.0.1 Breast Cancer Wisconsin

The Breast Cancer Wisconsin database, created by Wolberg in 1992, is a dataset currently comprising 569 samples representing cases of benign and malignant breast tumor diagnoses, therefore with two classes. Each sample has 30 numerical attributes calculated from digitized images of the breast tissue, including the mean tumor radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension.(WOLBERG WILLIAM; STREET, 1995)

3.3.0.2 Iris

The Iris database, created by Fisher in 1936, is a dataset consisting of 150 samples from three distinct classes, each with the same quantity as the others, referring to a type of Iris plant: Iris setosa, Iris virginica, and Iris versicolor. Each sample has 4 numerical attributes: sepal length, sepal width, petal length, and petal width.

A characteristic feature is that one of the 3 classes is linearly separable from the other two, but these are not linearly separable from each other. It is one of the oldest databases used in the literature on multi-class classification methods and is widely used primarily in the fields of statistics and machine learning (FISHER, 1988).

3.3.0.3 Yeast

The Yeast database is a dataset composed of 1484 samples, with 8 attributes each that represent the subcellular localization of proteins in Gram-negative bacteria. They are divided into 4 classes based on the subcellular localization of the proteins: in the cytoplasm, in the inner membrane (cytoplasmic), in the periplasm or in the outer membrane (NAKAI, 1996; HORTON; NAKAI, 1996). This dataset presents a superior challenge due to the imbalance and overlap between classes (NAKAI; KANEHISA, 1991).

3.3.0.4 Wine

The Wine database is a dataset consisting of 178 samples of wines grown in the same region of Italy but derived from three different cultivars. It has unequal proportions of samples for the 3 varieties, and each sample has the following chemical attributes: Alcohol, Malic Acid, Ash, Alkalinity of ash, Magnesium, Total phenols, Flavonoids, Non-flavonoid phenols, Proanthocyanidins, Color intensity, Hue, OD280/OD315 of diluted wines, and Proline (AEBERHARD; FORINA, 1991).

3.3.0.5 MNIST

The MNIST, Modified National Institute of Standards and Technology database, is a collection of images of handwritten digits from 0 to 9, consisting of a training set of

60,000 images and a test set of 10,000 images. All the images have been standardized, centered, and formatted into a square shape with a resolution of 28 x 28 pixels and are in grayscale.

It is one of the oldest and most used image databases for image classification, but it does not have a very high complexity. As a result, it is relatively simple to achieve low error rates with current classification models.

Figure 12 – Sample images from the MNIST database used for evaluating the impact of GLN-type activation functions on image classification accuracy.

3.3.0.6 CIFAR-10

The CIFAR-10 (KRIZHEVSKY; HINTON, 2009) image bank was chosen, which has 60 thousand color images in its repository classified into 10 different categories, automobile, bird, plane, cat, etc. These images have a resolution of 32x32 pixels with 3 RGB layers and are divided into 50 thousand training images and 10 thousand for testing. Of these training courses, 5,000 were taken for validation. These images, originally represented by a three-dimensional vector, were resized to become a linear array with 3072 input data to feed the neural network models.

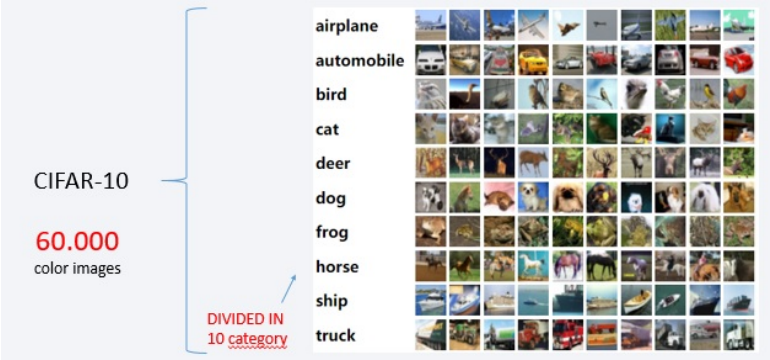


Figure 13 – Sample images from the CIFAR-1 database used for evaluating the impact of GLN-type activation functions on image classification accuracy.

3.3.0.7 LC25000

The Lung and Colon Cancer Histopathological Image database is a collection of histological entities composed of five classes of images: colon adenocarcinoma, benign colonic tissue, lung adenocarcinoma, lung squamous cell carcinoma, and benign lung tissue. It includes a total of 25,000 color images with a resolution of 768x768 pixels.

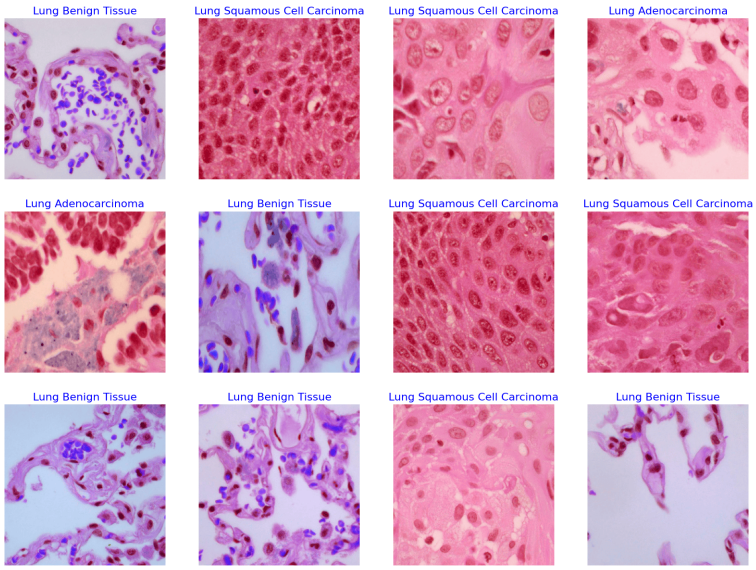


Figure 14 – Sample images from the Lung and Colon Cancer Histopathological Image database used for evaluating the impact of GLN-type activation functions on image classification accuracy.

3.4 Data collect

The objective of this phase is to collect data on the values of accuracy, loss, and convergence speed (in terms of the number of training epochs) for the test set in the process of evaluating the models. For this purpose, three specific neural network models were created and trained on different problems to evaluate the performance of the activation functions based on the generated data. In the first phase of the research, an MLP Autoencoder Network was used for the image compression problem, and in the second phase, an MLP Network for the multi-class classification problem and a CNN for the image classification problem.

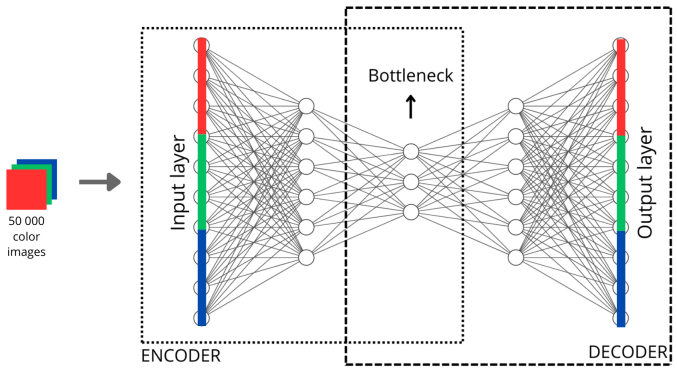


Figure 15 – Illustration of the transformation process of the Cifar10 dataset into a one-dimensional vector for training in an autoencoder network

In the first phase, using the CIFAR10 image dataset, five models with a Multi-layer Perceptron (MLP) Autoencoder neural network architecture were created, each with a different bottleneck layer size, for generating compressed images. See Figure 15 which illustrates this phase. Considering the resolution of CIFAR10 images, which is 32x32 pixels, the following five variations of this layer size with their respective compression rates were chosen: 24x24 (25%), 20x20 (38%), 16x16 (50%), 12x12 (63%), and 8x8 (75%). The sizes of the second and fourth layers of this model were scaled to have a uniform rate of compression from the input layer to the bottleneck layer, and decompression from the bottleneck layer to the output layer. Thus, since the images in 'one-dimensional format' have 3072 values, 32x32x3, the final architectures were:

- 1º model: $3072 \rightarrow 2400 \rightarrow 1728 \rightarrow 2400 \rightarrow 3072$
- 2º model: $3072 \rightarrow 2136 \rightarrow 1200 \rightarrow 2136 \rightarrow 3072$

- 3º model: 3072 → 1920 → 768 → 1920 → 3072
- 4º model: 3072 → 1752 → 432 → 1752 → 3072
- 5º model: 3072 → 1632 → 192 → 1632 → 3072

All these models were trained using the Adam optimization algorithm (KINGMA; BA, 2014) with a learning rate of 0.0001 and the mean squared error (MSE) function as the metric and loss function, one of the most common loss functions for training autoencoders since the purpose is to reconstruct the input data. Training was performed for up to 1000 epochs and a stopping criterion was defined to interrupt training if there was no improvement of at least 0.00001 in the loss value of the validation set during 100 consecutive epochs.

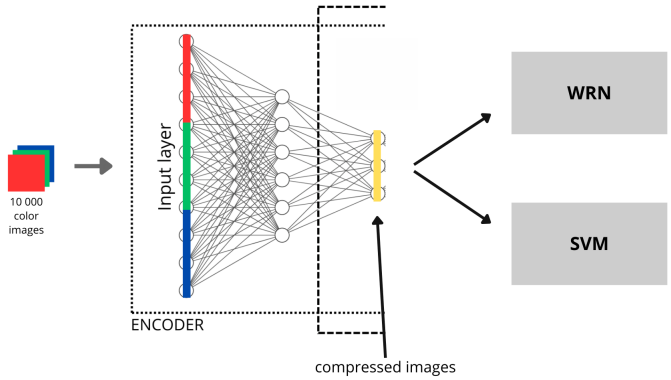


Figure 16 – Illustration of the transformation process of the Cifar10 dataset with 10 thousand test images to a one-dimensional vector for generating compressed images in the ENCODER for classification in the WRN and SVM models

After training, compressed images were created using the Encoder submodel of the Autoencoder network model in the 5 bottleneck variations and for the 3 activation functions evaluated in this phase. Thus, a new dataset based on CIFAR10 with reduced dimensionality was created. This new dataset was classified using a Width Residual Neural Network (WRN) with network depth set to 40 and broadening factor set to 4. The experiments were trained for 200 epochs using the SGD optimizer with Nesterov momentum (BENGIO et al., 2013) and smoothed cross-entropy as the loss function. The initial learning rate was set to 0.1, with momentum of 0.9, weight decay of 0.0005, and minibatch size of 1024. The learning rate was reduced by 0.2 at epochs 60, 120, and 160.

Additionally, we also classified the same compressed images using the Support Vector Machine (SVM) classification algorithm, comparing the classification efficiency of

both methods between the different activation functions and image compression levels. See Figure 16 which illustrates this phase.

In the second phase, for multiclass classification problems using the Breast Cancer, Iris, Wine and Yeast databases, we used MLP neural network models with 3 hidden layers, with 10 neurons in the first layer and 5 in the intermediate layer. In the last layer, the number of neurons varied according to the number of classes in each database. Similarly, for the Yeast database, since it is more complex, we used the same type of MLP neural network with 4 hidden layers with 100 neurons each, except for the last one with 10. The softmax activation function was used in all output layers.

Between the dense layers of these models, a Batch Normalization layer and a Dropout layer with a value of 0.3 were included. The learning rate was set at 0.001. The loss function used and the evaluation metric were respectively "Categorical Crossentropy" and "Categorical Accuracy" for the Breast Cancer, Iris, Wine and Yeast databases.

All MLP models were trained for up to ten thousand epochs. A stopping criterion was defined to interrupt training if there was no improvement of at least 0.0001 in the validation set loss value during 100 consecutive epochs for all models. A learning rate reduction by a factor of 0.1 was also established if no improvement occurred in at least 500 epochs in the validation set loss value.

For image classification tasks, a Convolutional Neural Network (CNN) based on the VGG19 architecture was created for training and obtaining data from all image datasets, MNIST, CIFAR10 and Lung Colon Cancer. The loss function used and the evaluation metric were respectively "Categorical Crossentropy" and "Accuracy". After initial performance tests with all activation functions, the optimization method chosen was Stochastic Gradient Descent (SGD) because it obtained better convergence in all activation functions. The other optimizers evaluated were: ADAM and RMSProp.

A maximum number of 500 epochs was established for training and an early stopping criterion was defined if there was no improvement of at least 0.0001 in the loss value of the validation set during 30 consecutive epochs. A reduction in the learning rate by a factor of 0.1 was also established with the following conditions:

- After more than 35 epochs for the first reduction;
- After more than 55 epochs for the second reduction;
- After more than 70 epochs for the third and final reduction.

All parameters of the neural network were kept constant except the activation functions used for evaluation in this research. However, in the second phase of the experiment, the batch size value and the weight initialization methods were varied in order to analyze how these factors can impact the performance of the activation functions, ensuring a robust evaluation in different model configurations. The batch size values applied were 8 and 32 for the multiclass classification problems and 32 and 128 for the image classification problems. The initialization methods used were: Glorot normal (GLOROT; BENGIO, 2010), He normal (HE et al., 2015a).

An important aspect to consider when working with neural networks is their stochastic nature (BISHOP, 1994). This means that the data obtained can vary with each execution, despite using the same dataset and the same initial parameters. This is due to several reasons, among the main ones, the randomness in the initialization of weights and also the stochastic nature of optimizers like SGD, ADAM, and RMSProp.

Therefore, to ensure more reliable measures of the models' performance, several rounds of training and testing were carried out for each activation function and for each variation of the model parameters. In the first phase of the experiment with the autoencoder networks, for each model configuration and for each activation function, 48 rounds were conducted. In the second phase, for all multi-class classification problems, the number of rounds was 50, since the execution of the models requires much less time. However, a smaller number of samples were performed for the image classification problems since the training time is quite long. In this case, the number of rounds defined was 30, the minimum amount necessary to ensure that this sample distribution is approximately normal, according to the Central Limit Theorem (KWAK; KIM, 2017).

The objective of these experimental configurations and the choice of neural network models used was not to achieve the best results for classifying these data nor to be the state-of-the-art for these problems, but rather to use already established models and basic configurations sufficient to provide consistent and comparative insights into the effects of the different activation functions studied.

3.5 Data analysis

The analysis focused on comparing the performance of the networks using different activation functions, measured by the values of accuracy, loss, and convergence speed (in terms of the number of training epochs). The KS test (Kolmogorov-Smirnov test) was used to compare independent samples and verify if they were drawn from populations with identical distributions. This test is useful insofar as we can determine whether the probability distributions obtained from the experiment results, for each activation function, can be considered from different distributions, and thereby the possibility of identifying a better performance or not. The analysis was conducted using Python software, with libraries such as NumPy, Pandas, SciPy, and Scikit-learn.

4 Results

In this chapter, we will show the results after training all models with the different activation functions. Initially, we will present the results from the first phase, which were presented and published at the 8th IEEE Latin American Conference on Computational Intelligence (IEEE LA-CCI 2022) (DELGADO; FERREIRA, 2022). Then, we will present the results from the second phase of the research. We will divide the results into two groups of problems: multiclass classification and image classification. The confusion matrix data are presented in the appendix.

4.1 First Phase Results

4.1.1 First step - Autoencoder training

Table 1 presents the results of tests performed for each variation in the number of neurons in the bottleneck layer, always for 48 experimental repetitions. The average loss value (MSE) for the image data of the training (Tr), validation (Val) , test (Te) and Epoch sets is presented. Complementing the information in Table 1, we present the data in boxplot format in Figure 17.

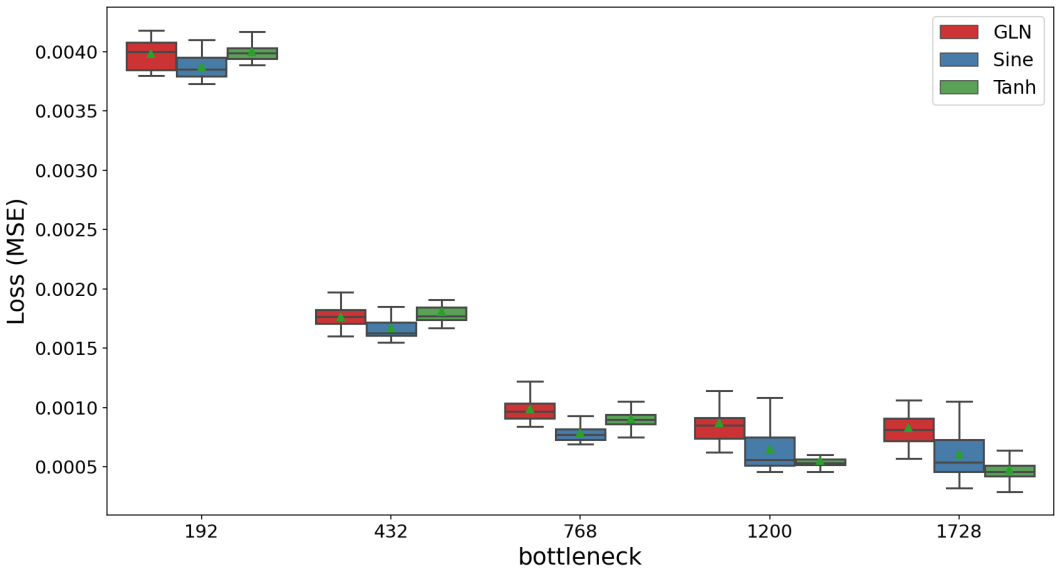


Figure 17 – *Box-Plot* of the loss for the bottleneck layer for each number of neurons experimented (48 experiments), outliers have been removed for better visualization

Table 1 – Mean values of the metrics for each bottleneck layer evaluated in all models, GLN and Tanh and Sine— results of the training (Tr), validation (Val) and Test (Te) sets. The best results are highlighted in bold-face.

Bottleneck	***Metric***	***GLN***	***Tanh***	***Sine***
1728	Tr_loss	0.000621	**0.000330**	0.000486
	Val_loss	0.000719	**0.000366**	0.000515
	Te_loss	0.000827	**0.000474**	0.000608
	Epoch	**192**	333	204
1200	Tr_loss	0.000645	**0.000426**	0.000519
	Val_loss	0.000737	**0.000476**	0.000549
	Te_loss	0.000873	**0.000554**	0.000647
	Epoch	**184**	396	211
768	Tr_loss	0.000819	0.000756	**0.000695**
	Val_loss	0.000909	0.000824	**0.000737**
	Te_loss	0.000987	0.000900	**0.000788**
	Epoch	**218**	361	341
432	Tr_loss	**0.001547**	0.001627	0.001549
	Val_loss	0.001672	0.001699	**0.001612**
	Te_loss	0.001765	0.001806	**0.001670**
	Epoch	**260**	335	420
192	Tr_loss	**0.003640**	0.003823	0.003734
	Val_loss	0.003866	0.003908	**0.003804**
	Te_loss	0.003982	0.003999	**0.003873**
	Epoch	**252**	431	309

As seen in Table 1, which shows the average loss values for each bottleneck value, it can be seen that the model that used Tanh's function has the lowest loss values for not very tight bottlenecks. From 50% of compression, the model with the periodic sine function starts to have the lowest loss values along with the GLN in the highest compressions for the training set. We will see later that this fact did not reflect on the performance of the GLN in the classification test. On the other hand, in all variations of the bottleneck, the model with GLN presents the best performance in terms of speed of convergence. Reducing the convergence time in relation to the second best result between 6% and 36% approximately.

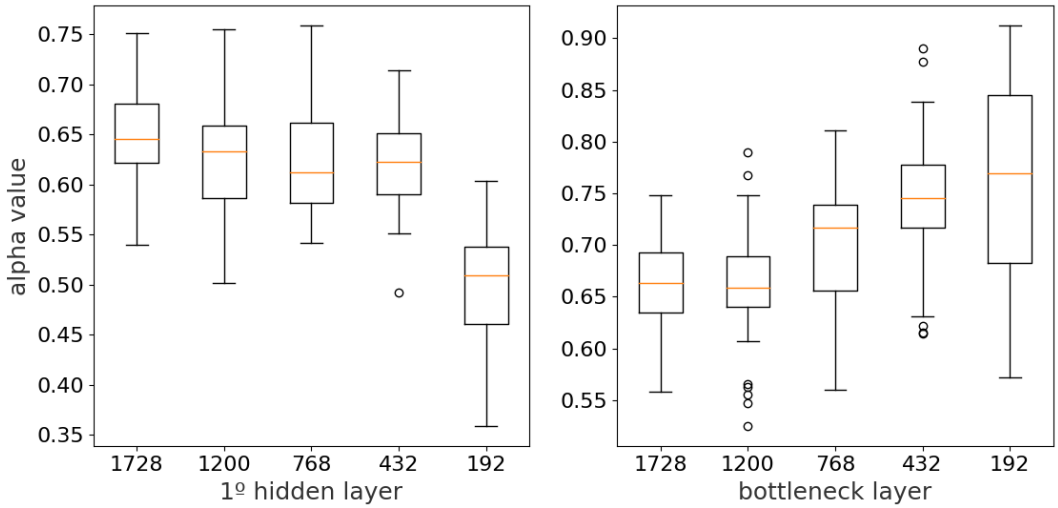


Figure 18 – *Box-Plot* of the α weight values for the first hidden layer and the bottleneck layer for each number of neurons experimented (48 experiments).

The value of the α weight was also evaluated. This weight shows the importance of each component of the activation function in the GLN. If equal to 1, it means that the neuron has the characteristic totally global (Sin function), and if equal to 0, totally local (Tanh function). For intermediate values, a mixture of these characteristics. As can be seen in Figure 18, the greater the compression, the greater the importance given to the global characteristic learned in the bottleneck layer. This shows that when the information compression is strong (few neurons in the bottleneck layer), the behavior is adapted for the characterization of the information through the global function. As the compression decreases (the number of neurons in the bottleneck layer increases), the network starts to increase the importance of the local component of the activation function, representing the total information with greater relevance to the details and resulting in

better auto-associative performance. In this sense, it is possible to prove the adaptation learned by the activation function during the training of the neural network.

4.1.2 Second step - Classification of the images compressed by the Autoencoder

Table 2 – Mean values of the accuracy for each bottleneck layer evaluated in all models, GLN and Tanh and Sine. The best results are highlighted in bold-face.

Bottleneck	WRN			SVM		
	Tanh	GLN	Sine	Tanh	GLN	Sine
1728	36.45%	**39.09%**	32.96%	45.22%	42.49%	**45.70%**
1200	35.11%	**38.24%**	35.46%	45.33%	43.51%	**45.83%**
768	**35.25%**	35.00%	35.17%	45.63%	45.66%	**46.85%**
432	**34.80%**	33.24%	33.32%	45.80%	**47.24%**	46.84%
192	**38.53%**	35.86%	33.60%	47.63%	48.25%	**48.89%**

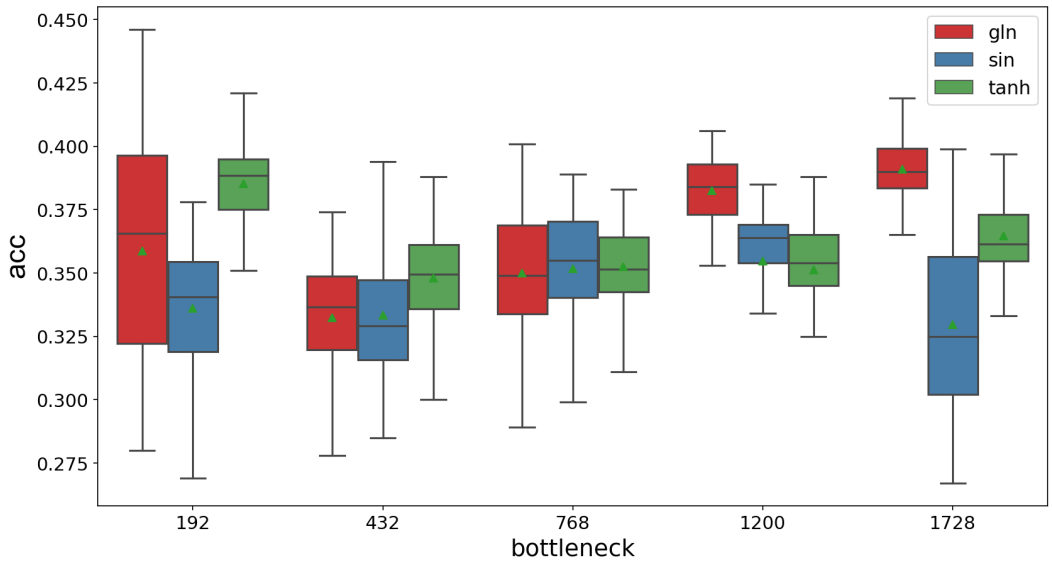


Figure 19 – *Box-Plot* of the accuracy for the bottleneck layer for each number of neurons experimented (48 experiments) using the WRN neural network for classification

As a basis for comparison, we evaluated the accuracy of the WRN used, with the

same set of 10,000 CIFAR10 test images, for training, validation, and testing. Obtaining an average accuracy of 71.7%. Remember that the second phase used these same 10 thousand images, but compressed.

The performance of GLN in the compressed layers of 1728 and 1200 stands out, as we can see in Table 2 and Figure 19. It obtained a performance of 7.2% superior to the second place when the bottleneck layer had 1728 neurons and 7.8% superior with 1200 neurons. The model with the Tanh activation function performed better in the classification in the most compacted layer with 192 neurons followed by the GLN, where, despite the natural tendency, to increase the weight of the global component for greater compactions, as we can see in Figure 18, GLN increased the importance of the first hidden layer to the local component and thus achieved a classification performance close to that obtained in the Tanh model. Thus demonstrating the ability to adapt and take advantage of the best features of both in different situations. In the other bottleneck layers, despite the different average values presented, it is not possible to say that they are of different distributions by the KS test performed, as will be shown next.

Table 3 – KS test results for *loss function* (MSE) and accuracy values

bottleneck	GLN-Tanh				GLN-Sin				Tanh-Sin			
	loss		accuracy		loss		accuracy		loss		accuracy	
	Statistically Similar	p-value	Statistically Similar	p-value	Statistically Similar	p-value	Statistically Similar	p-value	Statistically Similar	p-value	Statistically Similar	p-value
1728	No	2.88E-19	No	7.43E-12	No	2.05E-10	No	2.88E-19	No	1.00E-04	No	2.66E-06
1200	No	1.89E-20	No	7.44E-11	No	2.05E-10	No	2.68E-08	No	0.017902	No	0.036121
768	No	0.000955	Yes	0.250293	No	3.70E-18	Yes	0.960171	No	7.43E-12	Yes	0.692660
432	Yes	0.522069	Yes	0.058763	No	8.28E-07	Yes	0.692660	No	4.25E-09	No	0.002134
192	No	0.004544	No	0.000165	No	0.000955	No	0.000407	No	4.25E-09	No	4.02E-16

The bi-sample Kolmogorov–Smirnov (KS) test was performed in order to assess whether the three architectures (the architecture with GLN and the architecture with Tanh and Sine) are statistically similar for each of the metric values, evaluated in relation to the bottleneck size. The bisample KS test is a nonparametric test used to compare two-sample empirical distribution functions (GIBBONS; CHAKRABORTI, 2014). The null hypothesis is that the two population distributions are identical, in other words, two samples from the same population. The alternative hypothesis is that the two distributions sampled come from different population distributions. We apply this test to the *loss function* values obtained in the first phase with the test images and for the accuracy value obtained in the

second phase in the classification of compressed images and in the different arrangements of the activation functions studied.

According to Table 4, where we applied the KS test with GLN and Tanh, GLN and Sine, and Tanh and Sine, the null hypothesis was not rejected for the loss values of the models with GLN and Tanh in the bottleneck layer, 432, and for the accuracy values and in layers 768 and 432. Indicating that they may be one of the same probability distribution. For all other experiments, the null hypothesis was rejected at the 5% significance level. This test validates the results presented above.

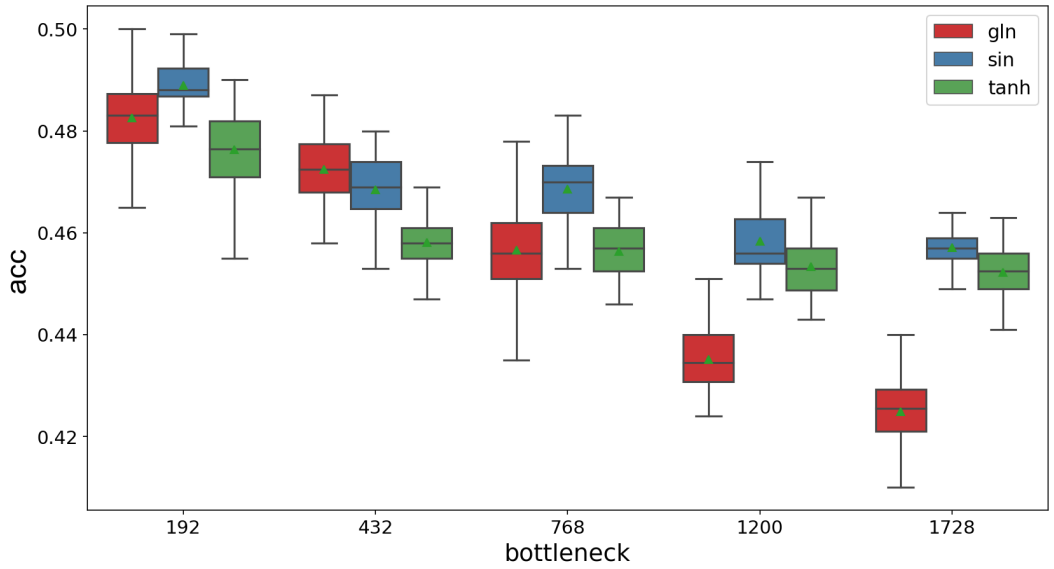


Figure 20 – *Box-Plot* of the accuracy for the bottleneck layer for each number of neurons experimented (48 experiments) using the SVM method for classification.

However, when SVM was used for classification, GLN was better when the bottleneck layer was more compacted, Table 2 and Figure 20. The accuracy value in the SVM with the complete set of the CIFAR10 image bank was 46%.

We also present in the Table 3 the classification report obtained with the confusion matrix, normally used to evaluate the performance of classifiers in data sets. These data confirm the superior performance of GLN for compression values up to 50%.

Table 4 – Classification Report Confusion Matrix

act	metric	1728			1200			768			432			192		
		acc	m avg	w avg	acc	m avg	w avg	acc	m avg	w avg	acc	m avg	w avg	acc	m avg	w avg
GLN	precision		**39.2%**	**39.7%**		**38.3%**	**38.6%**		35.1%	35.6%		33.3%	33.8%		35.9%	37.1%
	recall		**38.8%**	**39.1%**		**38.1%**	**38.2%**		35.0%	35.0%		33.1%	33.2%		35.5%	35.9%
	f1-score	**39.1%**	**38.9%**	**39.3%**	**38.2%**	**38.2%**	**38.4%**	35.0%	34.9%	35.2%	33.2%	33.1%	33.4%	35.9%	35.5%	36.3%
Tanh	precision		36.5%	36.9%		35.7%	36.0%		**35.3%**	**35.7%**		**34.9%**	**35.6%**		**38.5%**	**39.3%**
	recall		36.4%	36.4%		35.5%	35.6%		**35.3%**	**35.3%**		**34.7%**	**34.8%**		**38.1%**	**38.5%**
	f1-score	36.4%	36.3%	36.6%	35.6%	35.5%	35.7%	**35.3%**	**35.2%**	**35.4%**	**34.8%**	**34.6%**	**35.0%**	**38.5%**	**38.2%**	**38.8%**
Sin	precision		36.5%	36.7%		36.4%	36.9%		33.0%	33.5%		34.5%	34.9%		31.5%	32.4%
	recall		36.4%	36.4%		36.0%	36.3%		32.7%	32.9%		34.4%	34.4%		31.2%	31.4%
	f1-score	36.4%	36.3%	36.4%	36.3%	36.1%	36.5%	32.9%	32.7%	33.1%	34.4%	34.3%	34.5%	31.4%	31.1%	31.7%

4.2 Second Phase Results

4.2.1 Multiclass Classification Problems

4.2.1.1 Breast Cancer

Tables 5, 6, 7, 8 and 9 present the results, for the Breast Cancer dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine functions; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function; GLN-GCU and the functions that compose it, the Tanh function and GCU; GLN-GCU-R and the functions that compose it, the ReLU and GCU function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. These results were grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summary view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 21, 23, 25, 27, 29. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The α values per layer of the neural network are also presented in bloxpot format in the Figures 22, 24, 26, 28 and 30.

Table 5 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Breast Cancer - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	8	GL	97.35	1.01	100.00	95.61	0.08	0.03	**179.06**	31.61	97.38	97.35	97.35
Sin			96.49	1.20	99.12	94.74	0.10	0.02	239.38	74.28	96.61	96.49	96.51
Tanh			**97.68**	1.00	100.00	95.61	**0.08**	0.03	186.72	37.04	**97.7**	**97.68**	**97.68**
GLN	8	HE	97.22	1.02	99.12	94.74	0.09	0.03	183.00	27.83	97.24	97.22	97.23
Sin			96.93	0.97	99.12	93.86	0.09	0.02	231.06	50.18	97.0	96.92	96.94
Tanh			97.54	0.87	99.12	95.61	0.09	0.03	181.34	37.82	97.55	97.54	97.54
GLN	64	GL	97.33	0.92	99.12	95.61	0.09	0.02	201.98	23.25	97.35	97.33	97.33
Sin			97.01	1.11	100.00	92.98	0.09	0.03	231.90	67.84	97.05	97.01	97.02
Tanh			97.33	0.87	99.12	94.74	0.08	0.02	206.38	36.90	97.35	97.33	97.33
GLN	64	HE	97.08	1.03	99.12	94.69	0.09	0.02	214.98	42.68	97.1	97.08	97.09
Sin			96.54	1.14	98.25	92.04	0.10	0.03	291.08	71.38	96.61	96.54	96.55
Tanh			97.24	1.02	99.12	94.74	0.08	0.02	199.54	35.60	97.26	97.24	97.25

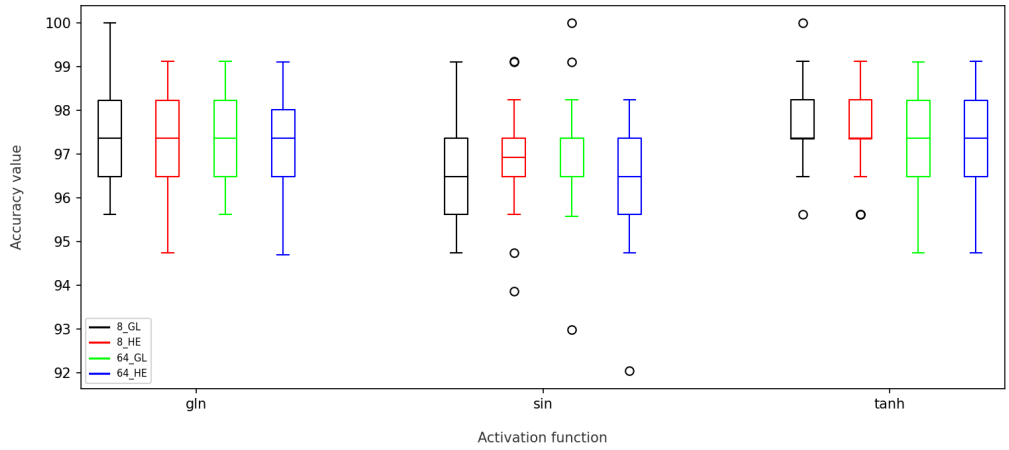


Figure 21 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Breast Cancer - GLN/Tanh/Sine)

As can be seen in Table 5 and Figure 21, the hyperbolic tangent function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 97.68, between the GLN and Sine functions. This performance was also reflected in the precision, recall and f1-score metrics. However, as we see in Table 10, the model with the GLN function has a statistically similar distribution to the model with the Tanh function in all variations. It is also possible to notice that the GLN function performs slightly better using the GL initialization method regardless of the batch size of this experiment. While the Tanh function shows slightly better performance at batch size 8 regardless of the initialization method. For the sine function, it is not possible to perceive any preference here.

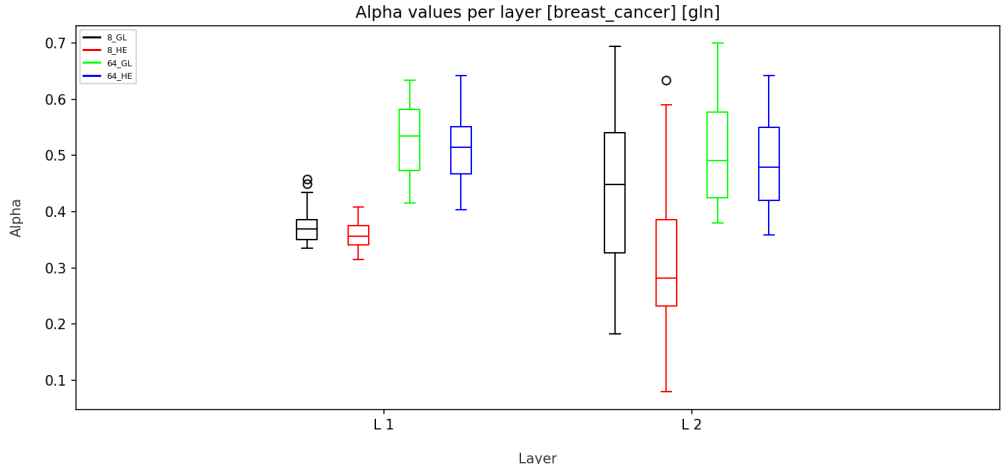


Figure 22 – *Box-Plot* of weight values α per layer (50 experiments) [Breast Cancer - GLN]

It is also observed that all activation functions managed to reach the maximum value of 100% for this data set. The Tanh and GLN functions arrived at this value with the parameters 8 GL and the sine function, 64 GL. The GLN function stands out in terms of convergence speed. It obtained the lowest convergence value, 179 epochs and lowest standard deviation, 31, for variation 8 GL, where the best results were obtained for the accuracy of GLN and Tanh. Indicating greater stability.

The distribution of α values can be seen in Figure 22. It can be seen that the values of α tend to move towards the activation function with the best performance. These values tend to be below 0.5 for batch value 8, where the tanh function has better performance, and close to 0.5, for batch value 64, where the sine function has performance closer to that obtained by the tanh function.

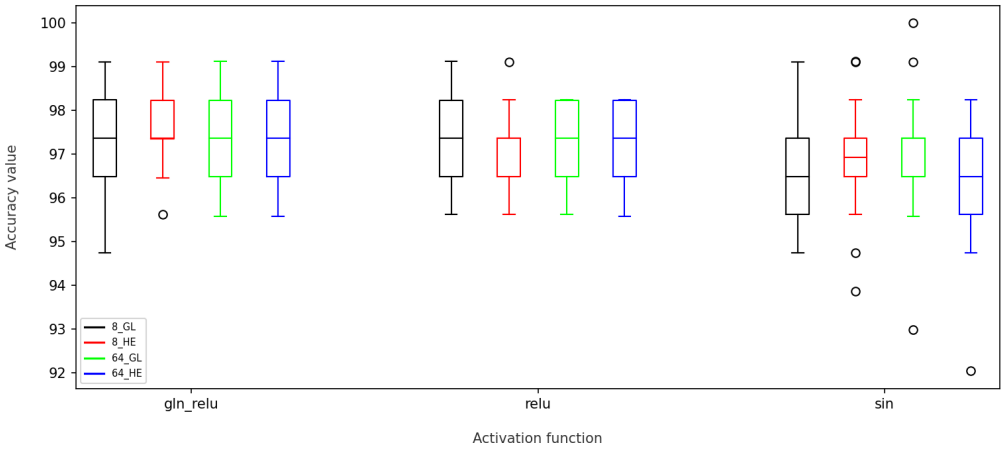


Figure 23 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Breast Cancer - GLN-ReLU/ReLU/Sine)

Table 6 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Breast Cancer - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	8	GL	97.42	0.89	99.12	94.74	0.08	0.02	**177.40**	39.94	97.44	97.42	97.42
ReLU			97.33	0.83	99.12	95.61	0.09	0.02	230.48	77.13	97.37	97.33	97.34
Sin			96.49	1.20	99.12	94.74	0.10	0.02	239.38	74.28	96.61	96.49	96.51
GLN-ReLU	8	HE	**97.45**	0.71	99.12	95.61	**0.08**	0.02	179.66	40.17	**97.47**	**97.45**	**97.46**
ReLU			97.22	0.78	99.12	95.61	0.09	0.02	257.54	60.24	97.26	97.22	97.23
Sin			96.93	0.97	99.12	93.86	0.09	0.02	231.06	50.18	97.0	96.92	96.94
GLN-ReLU	64	GL	97.26	0.95	99.12	95.58	0.08	0.02	206.96	35.42	97.29	97.26	97.27
ReLU			97.17	0.85	98.25	95.61	0.09	0.02	238.96	41.48	97.21	97.17	97.18
Sin			97.01	1.11	100.00	92.98	0.09	0.03	231.90	67.84	97.05	97.01	97.02
GLN-ReLU	64	HE	97.26	0.95	99.12	95.58	0.09	0.03	211.44	38.94	97.28	97.26	97.26
ReLU			97.22	0.80	98.25	95.58	0.08	0.02	257.72	45.06	97.25	97.22	97.23
Sin			96.54	1.14	98.25	92.04	0.10	0.03	291.08	71.38	96.61	96.54	96.55

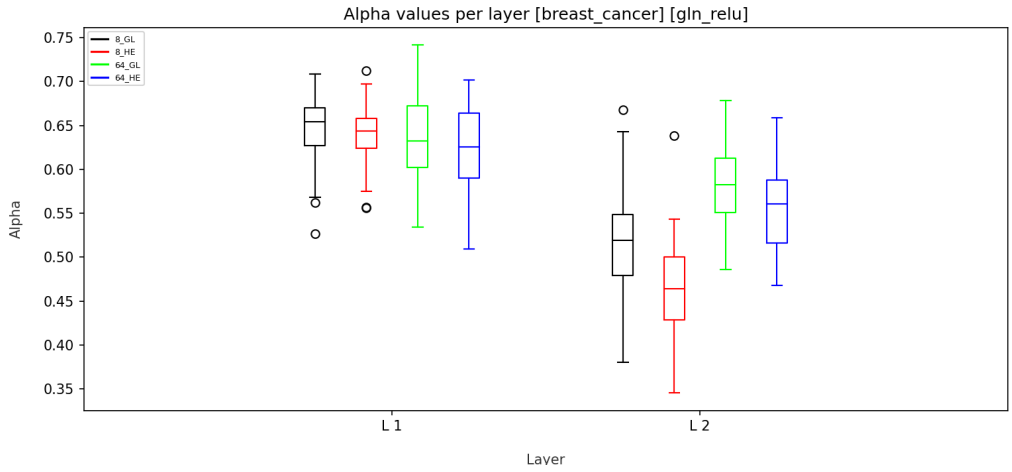


Figure 24 – *Box-Plot* of weight values α per layer (50 experiments) [Breast Cancer - GLN-ReLU]

In Table 6, where the results for the GLN-ReLU and ReLU and Sine functions are presented, the GLN-ReLU function stands out, obtaining the highest accuracy value, 97.45. This performance was also reflected in the precision, recall and f1-score metrics. At the 5% significance level, the results of the KS test, presented in Table 10, demonstrate that the GLN-ReLU and ReLU Functions are similar in their distributions in all tested variations. No significant difference is noted from the results presented in Table 6 and Figure 23 for the different batch size variations and for the initialization method, although the Normal HE initialization method is known to be more effective for ReLU (HE et al., 2015a) functions. The sine function was the only one that presented the highest maximum value of 100%, in the batch variation equal to 64 and initialization of the Glorot Normal

weights.

The lowest epoch value, 177, was obtained by the GLN-ReLU function as shown in Table 6. This result was almost 25% better than that obtained by the second best convergence time obtained by the ReLU function, which required 230 epochs. GLN-ReLU was the one that presented the best stability in the convergence time, as can be seen in the standard deviation values for the distribution of values for the period in the same table.

The distribution of α values is shown in Figure 24. In the first layer of the model, the values of α approached the global sine function, without much difference in the distributions for the 4 variations. While for the second layer, there was a slight difference according to the batch size value and the initialization method. For batch value 8 and He Normal method, α tends to slightly choose the sine function.

Table 7 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Breast Cancer - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	8	GL	**97.79**	0.89	99.12	95.61	0.08	0.02	**175.42**	40.88	**97.81**	**97.79**	**97.79**
Mish			97.44	0.83	99.12	95.61	0.08	0.02	238.60	53.28	97.48	97.43	97.44
Sin			96.49	1.20	99.12	94.74	0.10	0.02	239.38	74.28	96.61	96.49	96.51
GLN-Mish	8	HE	97.61	0.99	100.00	95.61	0.08	0.02	181.44	39.85	97.62	97.61	97.61
Mish			97.54	0.79	99.12	96.49	0.08	0.02	244.54	54.39	97.58	97.54	97.55
Sin			96.93	0.97	99.12	93.86	0.09	0.02	231.06	50.18	97.0	96.92	96.94
GLN-Mish	64	GL	97.21	1.10	98.25	94.74	0.08	0.02	203.10	35.97	97.23	97.21	97.21
Mish			97.33	0.99	100.00	95.61	**0.08**	0.02	250.58	49.85	97.37	97.33	97.34
Sin			97.01	1.11	100.00	92.98	0.09	0.03	231.90	67.84	97.05	97.01	97.02
GLN-Mish	64	HE	97.15	1.09	99.12	93.86	0.09	0.02	225.64	51.46	97.18	97.15	97.16
Mish			97.28	0.91	99.12	95.61	0.08	0.02	251.54	48.13	97.32	97.28	97.28
Sin			96.54	1.14	98.25	92.04	0.10	0.03	291.08	71.38	96.61	96.54	96.55

The GLN-Mish function obtained the highest accuracy value compared to the functions that compose it, Mish and Sine, as can be seen in Table 7. It obtained an average accuracy of 97.79 This better performance was also identified in the precision, recall and f1-score metrics. At the 5% significance level, the results of the KS test, presented in Table 10, demonstrate that the GLN-Mish and Mish Functions are statistically similar in all tested variations. As was identified for the GLN-Mish and Mish functions, the different batch sizes and initialization methods used did not significantly affect the results, as can be seen in Table 7 and Figure 25. All 3 activation functions reached the maximum result of 100.

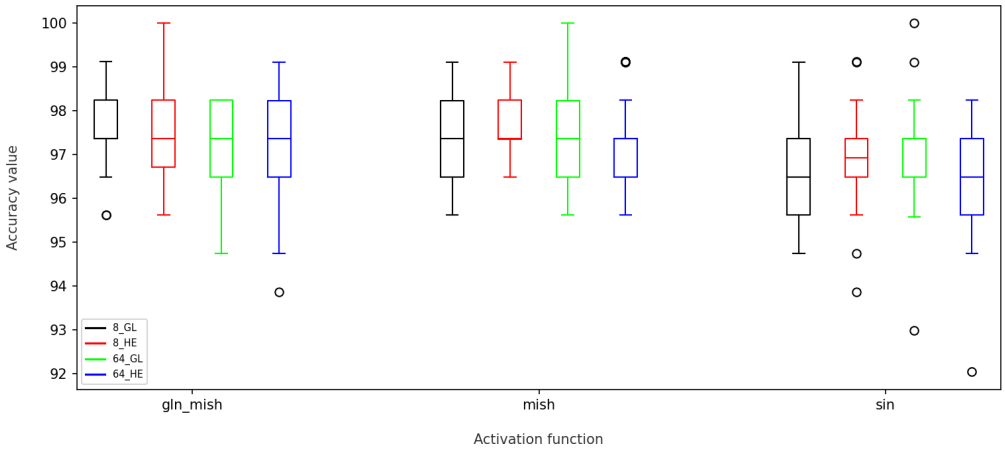


Figure 25 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Breast Cancer - GLN-Mish/Mish/Sine)

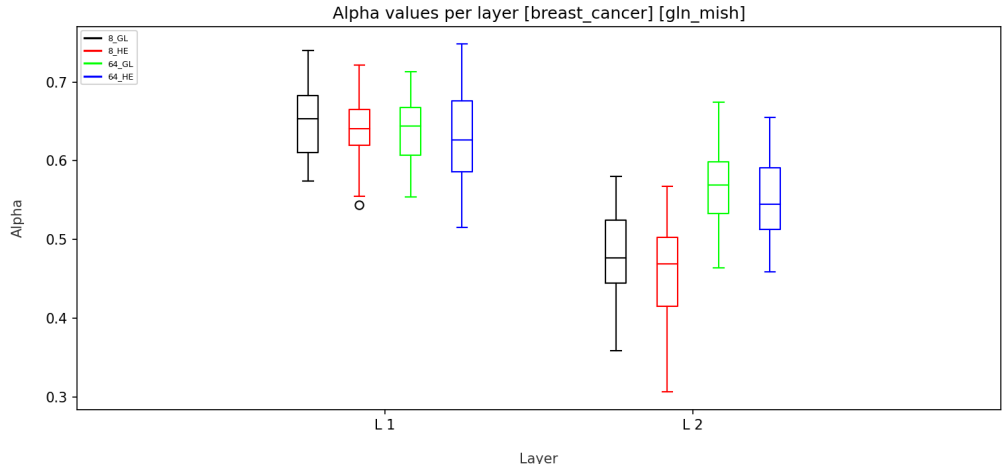


Figure 26 – *Box-Plot* of weight values α per layer (50 experiments) [Breast Cancer - GLN-Mish]

The lowest average value for the convergence time, 175, was obtained by the GLN-Mish function as shown in Table 7. This result was almost 25% better than that obtained by the second best convergence time obtained by the Sine function, which presented an average value of 231 epochs. GLN-Mish also presented the best stability in the convergence time, as can be seen in the standard deviation values for the distribution of values for the time in the same table.

The distribution of α values is presented in Figure 26 and presents a very similar result to that obtained by the α values for the GLN-ReLU function. In the first layer of the model, the values of α approached the global sine function, without much difference in the distributions for the 4 variations. While for the second layer, there was a slight difference depending on the batch size value. For batch value 8, α tends to slightly choose

the mish function, while α tends to choose the sine function for value 64.

Table 8 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Breast Cancer - GLN-GCU/Tanh/GCU)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	95.08	1.93	98.25	90.35	0.16	0.06	333.08	111.83	95.23	95.08	95.11
GLN-GCU			96.70	2.58	99.12	80.70	0.10	0.05	325.40	91.78	96.75	96.7	96.71
Tanh			**97.68**	1.00	100.00	95.61	**0.08**	0.03	186.72	37.04	**97.7**	**97.68**	**97.68**
GCU	8	HE	94.61	3.37	99.12	74.56	0.18	0.07	431.64	144.39	94.8	94.6	94.64
GLN-GCU			96.77	1.29	99.12	92.98	0.09	0.03	372.42	109.04	96.82	96.77	96.78
Tanh			97.54	0.87	99.12	95.61	0.09	0.03	**181.34**	37.82	97.55	97.54	97.54
GCU	64	GL	94.01	2.12	98.23	86.84	0.18	0.08	496.18	164.08	94.27	94.01	94.06
GLN-GCU			94.94	2.99	98.25	78.07	0.14	0.07	395.74	131.21	95.07	94.94	94.97
Tanh			97.33	0.87	99.12	94.74	0.08	0.02	206.38	36.90	97.35	97.33	97.33
GCU	64	HE	87.95	7.74	96.49	67.54	0.30	0.14	578.26	217.79	89.29	87.94	88.22
GLN-GCU			94.06	2.58	98.23	86.84	0.17	0.06	530.44	149.73	94.26	94.06	94.1
Tanh			97.24	1.02	99.12	94.74	0.08	0.02	199.54	35.60	97.26	97.24	97.25

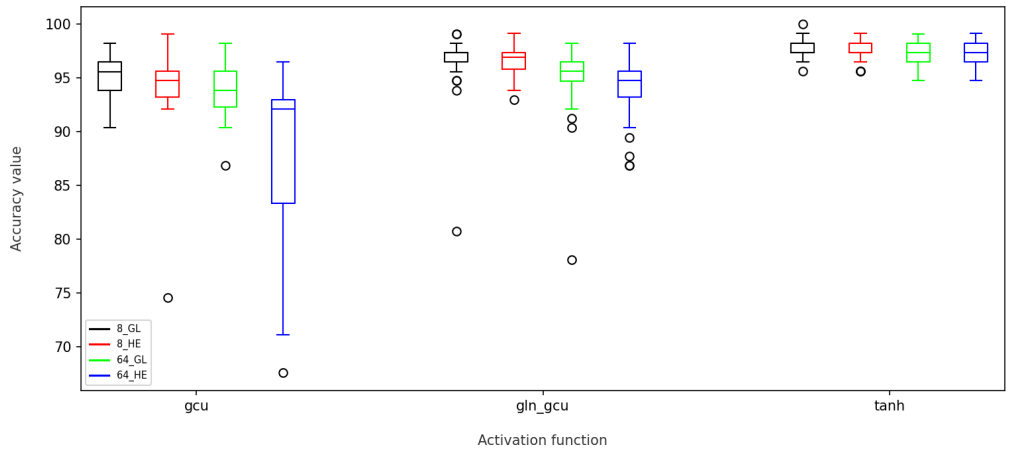


Figure 27 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Breast Cancer - GCU/GLN-GCU/Tanh)

As shown in Table 8 and Figure 27, the Tanh function performed best in relation to the GLN-GCU and GCU activation functions. Except for the 8 HE variation, the KS test presented in Table 10 shows that the Tanh and GLN-GCU functions are not statistically similar. The model with the GCU function presented the worst performance mainly for the larger batch size and for the normal HE initialization method. This parameter configuration led to a very accentuated dispersion, negatively influencing the GLN-GCU function. Although this managed to present greater stability compared to the GCU function. It is also noted that there was a large difference in the rate of convergence

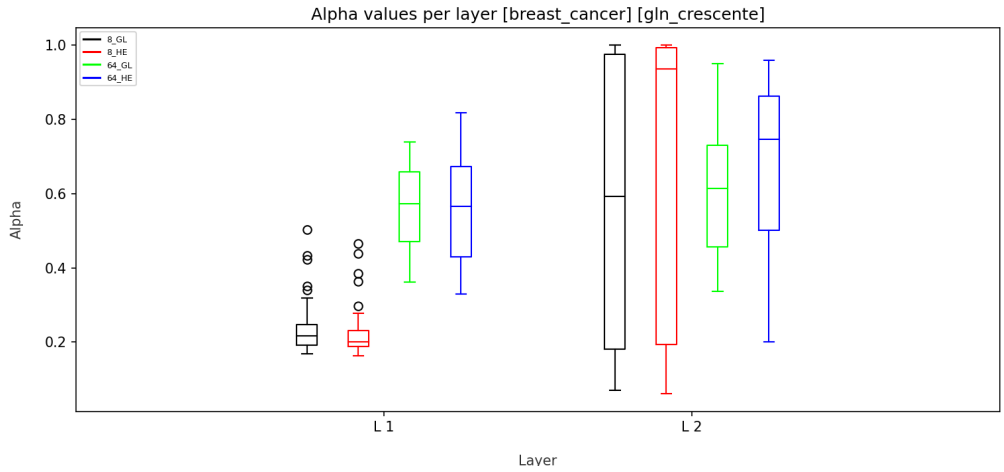


Figure 28 – *Box-Plot* of weight values α per layer (50 experiments) [Breast Cancer - GLN-GCU]

between the Tanh functions and the others. Tanh is the most efficient in this aspect as well.

We can see from Figure 28 that the values of α in the first layer for the batch size equal to 8 are very close to 0.2, indicating a greater contribution to the hyperbolic tangent function. While for size 64 it is close to 0.6, indicating a slight prevalence for the global Increasing Cosine function. In the second layer of the model, the value of α presents a very large dispersion, indicating a certain instability in the choice of the predominant function, especially for batch size equal to 8.

Table 9 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score.(Breast Cancer - GLN-GCU-ReLU/ReLU/GCU)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	95.08	1.93	98.25	90.35	0.16	0.06	333.08	111.83	95.23	95.08	95.11
GLN-GCU-ReLU			96.87	1.19	99.12	92.11	0.09	0.03	286.30	70.54	96.93	96.87	96.88
ReLU			**97.33**	0.83	99.12	95.61	0.09	0.02	**230.48**	77.13	**97.37**	**97.33**	**97.34**
GCU	8	HE	94.61	3.37	99.12	74.56	0.18	0.07	431.64	144.39	94.8	94.6	94.64
GLN-GCU-ReLU			96.86	1.30	99.12	93.86	0.10	0.04	334.20	107.32	96.91	96.85	96.87
ReLU			97.22	0.78	99.12	95.61	0.09	0.02	257.54	60.24	97.26	97.22	97.23
GCU	64	GL	94.01	2.12	98.23	86.84	0.18	0.08	496.18	164.08	94.27	94.01	94.06
GLN-GCU-ReLU			94.91	2.23	98.25	85.09	0.15	0.06	403.28	125.33	95.02	94.9	94.93
ReLU			97.17	0.85	98.25	95.61	0.09	0.02	238.96	41.48	97.21	97.17	97.18
GCU	64	HE	87.95	7.74	96.49	67.54	0.30	0.14	578.26	217.79	89.29	87.94	88.22
GLN-GCU-ReLU			94.34	5.09	98.25	62.28	0.16	0.11	520.36	189.66	94.5	94.34	94.38
ReLU			97.22	0.80	98.25	95.58	**0.08**	0.02	257.72	45.06	97.25	97.22	97.23

As can be seen in Table 9 and Figure 29, the Tanh function was the one that

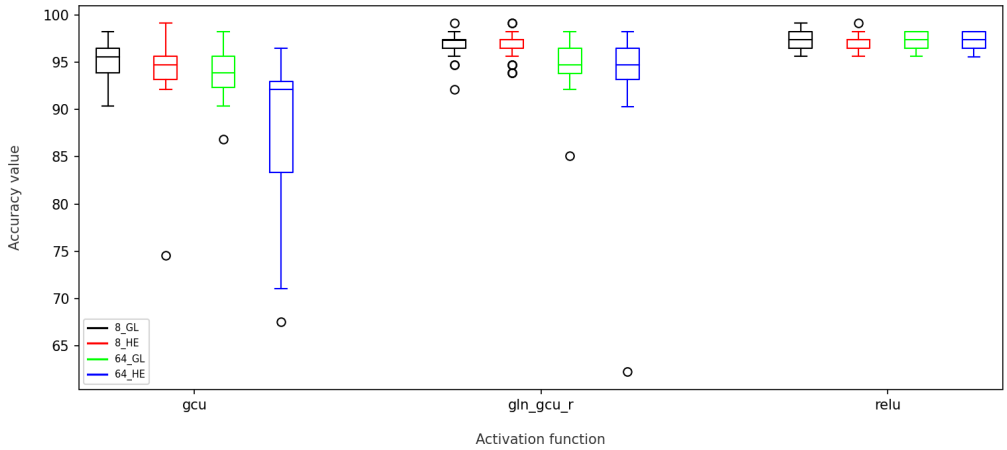


Figure 29 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Breast Cancer - GCU/GLN-GCU-R/ReLU)

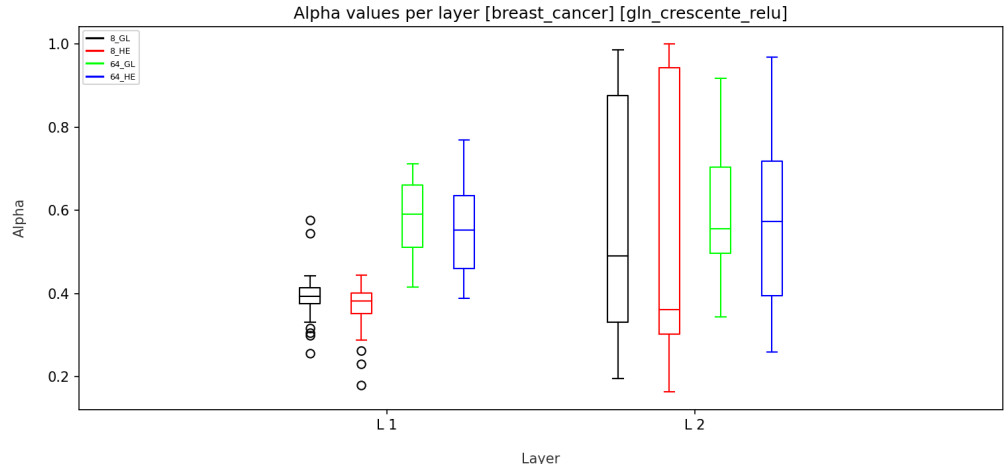


Figure 30 – *Box-Plot* of weight values α per layer (50 experiments) [Breast Cancer - GLN-GCU-R]

presented the best performance in relation to the GLN-GCU-R and ReLU activation functions. The KS test presented in Table 10 shows that the ReLU and GLN-GCU-R functions are statistically similar for batch size 8. The model with the GLN-GCU-R activation function, whose local component is given by the ReLU function, presents greater stability compared to the GLN-GCU activation function, whose local component is given by the Tanh function, when using the parameters with the worst performance of the GCU function, batch 64 and normal HE initialization method.

We can see from Figure 30 that the values of α show similar trends compared to the GLN-GCU activation function. In the first layer for batch size equal to 8 it approaches 0.4, indicating a greater contribution to the hyperbolic tangent function. While for size 64 it is close to 0.6, indicating a slight prevalence for the global Increasing Cosine function.

In the second layer of the model, the value of α also presents a very large dispersion, indicating a certain instability in the choice of the predominant function, especially for the batch size equal to 8. However, unlike the GLN-GCU function, this distribution tends more towards the local ReLU function.

Table 10 – KS test results for *accuracy* - Breast Cancer

batch size →	8								64							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	No	5.84e-03	Yes	0.549	Yes	0.179	Yes	0.272	Yes	0.869	Yes	0.998	No	1.15e-02	Yes	0.869
sin	-	-	No	3.80e-05		-	No	2.83e-03		-	Yes	0.396		-	No	1.31e-03
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	Yes	0.967	No	1.31e-03	Yes	0.549	No	2.83e-03	Yes	0.998	Yes	0.869	Yes	1.0	No	2.83e-03
relu		-	No	5.84e-03		-	Yes	0.272		-	Yes	0.967		-	No	1.15e-02
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	Yes	0.396	No	1.45e-07	Yes	0.869	No	2.83e-03	Yes	0.717	Yes	0.272	Yes	0.869	No	5.84e-03
mish		-	No	5.82e-04		-	No	5.84e-03		-	Yes	0.869		-	No	5.84e-03
act	gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh	
gcu	No	1.39e-05	No	8.76e-13	No	1.58e-06	No	8.76e-13	No	1.15e-02	No	3.77e-17	No	1.58e-06	No	1.49e-21
gln_gcu		-	Yes	0.112		-	No	1.15e-02		-	No	2.67e-11		-	No	8.76e-13
act	gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu	
gcu	No	1.45e-07	No	8.76e-13	No	4.93e-07	No	8.76e-13	Yes	0.112	No	2.81e-15	No	1.58e-06	No	3.21e-24
gln_gcu_r		-	Yes	0.549		-	Yes	0.396		-	No	1.06e-08		-	No	4.05e-08

4.2.1.2 Iris

Tables 11, 12, 13, 14 and 15 present the results, for the Iris dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine functions; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function; GLN-GCU and the functions that compose it, the Tanh function and GCU; GLN-GCU-R and the functions that compose it, the ReLU and GCU function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. The data was grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summary view of the data distribution and complement the information

in these tables, we present the accuracy data in bloxpot format in the Figures 31, 33, 35, 37, 39. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 32, 34, 36, 38 and 40.

Table 11 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Iris - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	8	GL	96.93	3.15	100.0	90.0	0.07	0.04	823.50	219.66	96.94	96.93	96.93
Sin			96.53	3.50	100.0	90.0	0.08	0.04	**787.60**	214.23	96.58	96.53	96.53
Tanh			**97.67**	2.95	100.0	90.0	0.10	0.04	848.88	220.14	**97.68**	**97.67**	**97.67**
GLN	8	HE	96.33	3.52	100.0	90.0	0.09	0.05	833.10	248.96	96.34	96.33	96.33
Sin			97.07	3.34	100.0	90.0	0.08	0.05	893.66	195.05	97.08	97.07	97.07
Tanh			97.07	3.41	100.0	90.0	0.10	0.05	896.40	224.05	97.07	97.07	97.06
GLN	64	GL	97.60	2.94	100.0	90.0	**0.07**	0.05	941.88	221.66	97.61	97.6	97.6
Sin			97.27	2.83	100.0	90.0	0.07	0.05	996.56	197.48	97.27	97.27	97.27
Tanh			97.13	3.16	100.0	90.0	0.07	0.05	966.24	164.35	97.13	97.13	97.13
GLN	64	HE	96.87	3.39	100.0	90.0	0.07	0.05	1059.94	294.80	96.87	96.87	96.87
Sin			96.60	2.82	100.0	90.0	0.08	0.05	1132.56	247.26	96.6	96.6	96.6
Tanh			97.07	3.48	100.0	90.0	0.08	0.06	1067.60	216.81	97.07	97.07	97.07

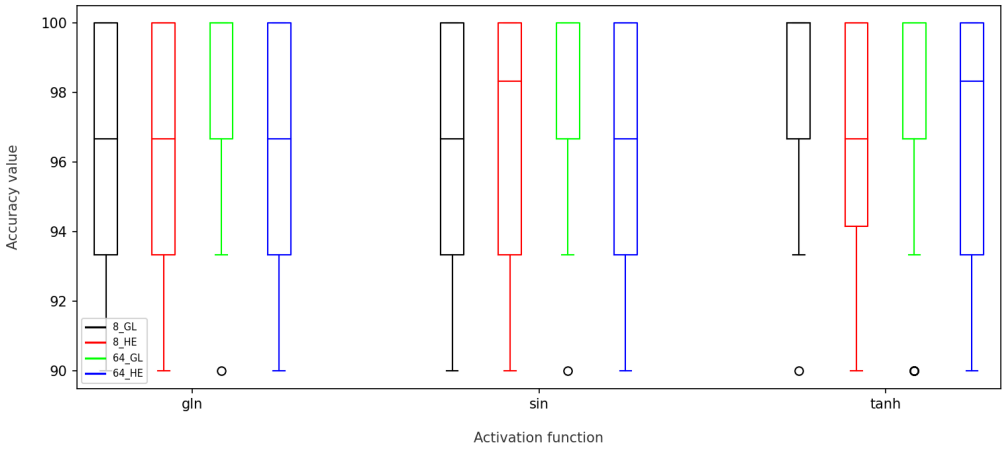


Figure 31 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (IRIS - GLN/Tanh/Sine)

As can be seen in Table 11 and Figure 31, the hyperbolic tangent function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 97.67, between the GLN and Sine functions. This

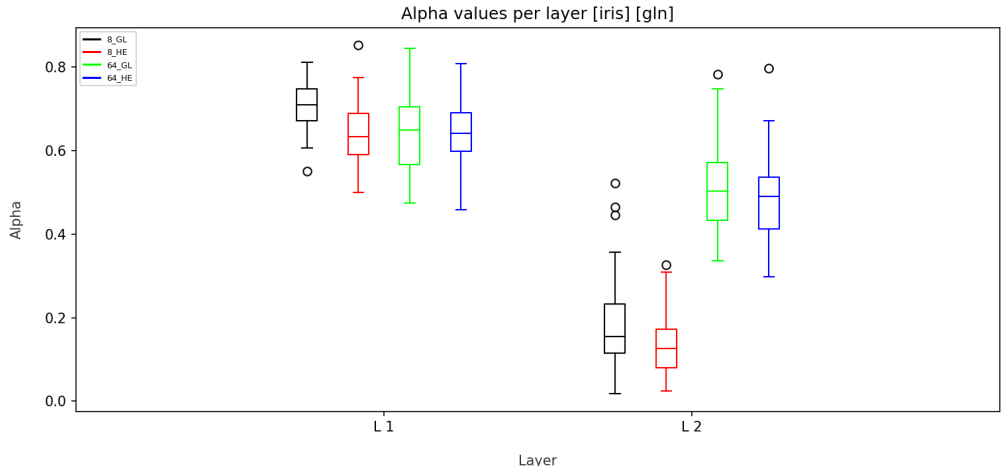


Figure 32 – *Box-Plot* of weight values α per layer (50 experiments) [IRIS - GLN]

performance was also reflected in the precision, recall and f1-score metrics. However, as we see in Table 16, all models present a statistically similar distribution in all variations. All functions reached the maximum accuracy value of 100% and the same minimum value of 90%. The model with the sine function was the one with the shortest convergence time, 787 epochs. No significant difference is noted in the results across the different parameters used.

The distribution of α values can be seen in Figure 32. In the first layer, regardless of the parameter variation, the values were above 0.5, indicating a choice for the global function. However, in the second layer, there was a clear difference in the values of α in relation to the batch size. α tended towards the local Tanh function in models with the batch size parameter at 8, while it was close to 0.5 for size 64.

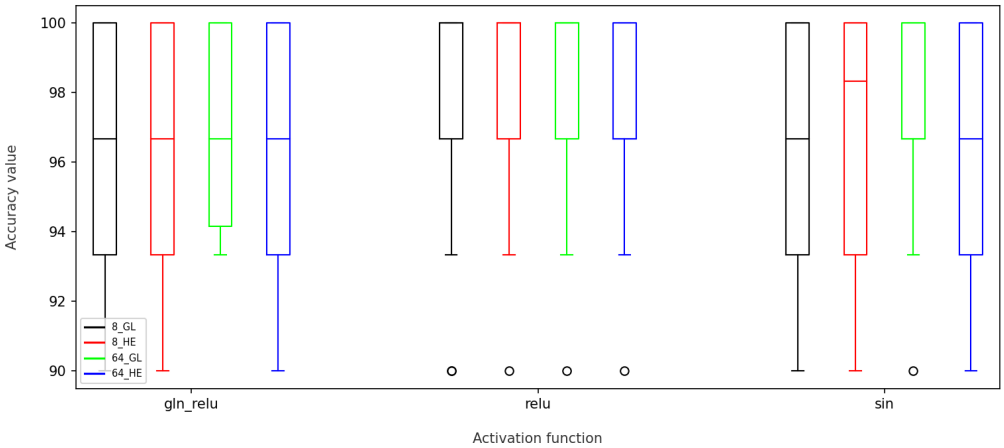


Figure 33 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (IRIS - GLN-ReLU/ReLU/Sine)

Table 12 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Iris - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	8	GL	96.67	3.37	100.0	90.0	0.08	0.05	787.66	219.77	96.69	96.67	96.67
ReLU			**97.80**	3.06	100.0	90.0	0.09	0.06	**598.72**	164.19	**97.8**	**97.8**	**97.8**
Sin			96.53	3.50	100.0	90.0	0.08	0.04	787.60	214.23	96.58	96.53	96.53
GLN-ReLU	8	HE	96.60	3.47	100.0	90.0	0.09	0.05	780.54	221.40	96.61	96.6	96.6
ReLU			97.53	2.92	100.0	90.0	0.09	0.05	693.84	221.92	97.55	97.53	97.53
Sin			97.07	3.34	100.0	90.0	0.08	0.05	893.66	195.05	97.08	97.07	97.07
GLN-ReLU	64	GL	97.13	2.69	100.0	93.33	0.07	0.05	867.28	196.98	97.14	97.13	97.13
ReLU			97.80	2.91	100.0	90.00	0.07	0.06	830.80	253.45	97.8	97.8	97.8
Sin			97.27	2.83	100.0	90.00	**0.07**	0.05	996.56	197.48	97.27	97.27	97.27
GLN-ReLU	64	HE	96.80	3.36	100.0	90.0	0.07	0.06	985.40	229.90	96.8	96.8	96.8
ReLU			97.47	2.74	100.0	90.0	0.07	0.05	942.88	229.27	97.47	97.47	97.47
Sin			96.60	2.82	100.0	90.0	0.08	0.05	1132.56	247.26	96.6	96.6	96.6

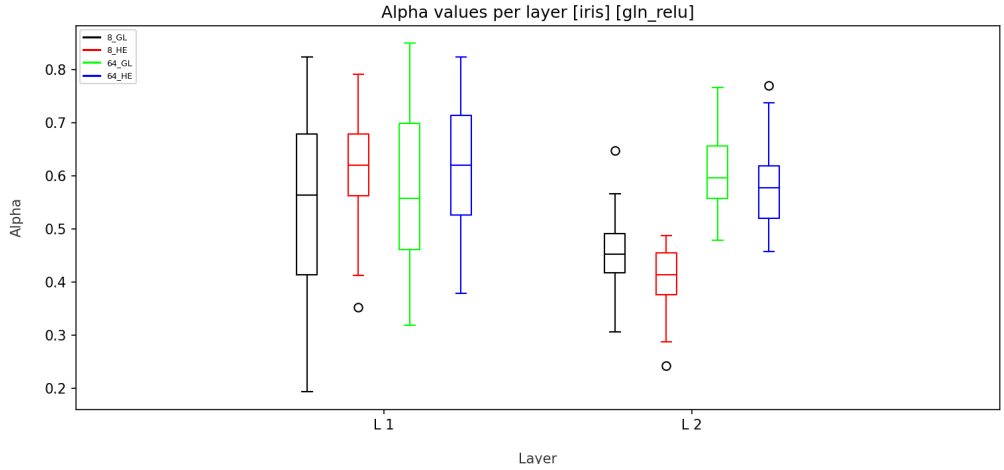


Figure 34 – *Box-Plot* of weight values α per layer (50 experiments) [IRIS - GLN-ReLU]

As seen in Table 12 and Figure 33, the ReLU function was the one that obtained the highest average accuracy value, 97.80, between the GLN-ReLU and Seno functions, both in the batch variation equal to 8 and 64, and in the GL initialization method. This performance was also reflected in the precision, recall and f1-score metrics. However, as we see in Table 16, the KS test demonstrates that the null hypothesis cannot be discarded, indicating that all models presented a statistically similar distribution in all variations. All functions reached the maximum value of 100% accuracy and the minimum value of at least 90%. The GLN-ReLU function was the one with the greatest stability for the accuracy value in variation 64, batch size and GL weights initialization method. And the model with the ReLU function was the one with the shortest convergence time, 598 epochs.

No significant difference was noted in the results in the different parameters used.

The distribution of α values can be seen in Figure 34. In the first layer, the values tended to be slightly above 0.5, indicating a slight choice for the global function. For the 8 GL variation, the distribution of α values became more dispersed. Reaching both values close to 0.2 and 0.8. However, in the second layer, there was also a clear distinction in the values of α for the two batch values. The value of α tended more towards the local ReLU function in models with the batch size parameter at 8, while it was close to 0.5 for size 64, indicating a balance in the weights between the sine and tanh functions.

Table 13 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Iris - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	8	GL	97.07	3.20	100.0	90.00	0.08	0.04	823.92	223.24	97.08	97.07	97.07
Mish			97.73	2.81	100.0	93.33	0.09	0.05	820.32	282.24	97.74	97.73	97.73
Sin			96.53	3.50	100.0	90.00	0.08	0.04	787.60	214.23	96.58	96.53	96.53
GLN-Mish	8	HE	96.53	3.50	100.0	90.00	0.09	0.05	812.62	211.21	96.54	96.53	96.53
Mish			97.87	2.76	100.0	93.33	0.10	0.04	**773.26**	224.81	97.87	97.87	97.87
Sin			97.07	3.34	100.0	90.00	0.08	0.05	893.66	195.05	97.08	97.07	97.07
GLN-Mish	64	GL	97.40	2.88	100.0	93.33	0.06	0.05	896.24	190.63	97.41	97.4	97.4
Mish			**98.13**	2.71	100.0	93.33	**0.06**	0.05	1020.32	225.91	**98.13**	**98.13**	**98.13**
Sin			97.27	2.83	100.0	90.00	0.07	0.05	996.56	197.48	97.27	97.27	97.27
GLN-Mish	64	HE	97.00	3.18	100.0	90.0	0.07	0.05	994.78	245.48	97.0	97.0	97.0
Mish			97.67	2.88	100.0	90.0	0.06	0.05	1132.88	285.80	97.67	97.67	97.67
Sin			96.60	2.82	100.0	90.0	0.08	0.05	1132.56	247.26	96.6	96.6	96.6

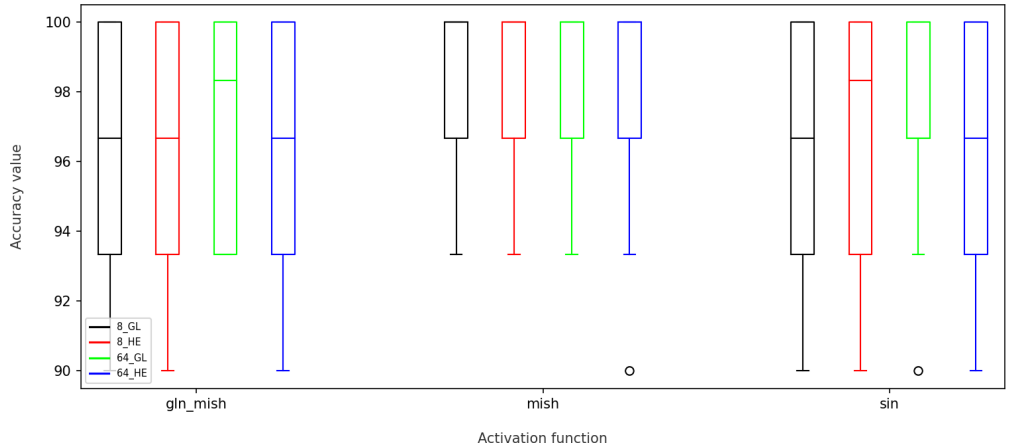


Figure 35 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (IRIS - GLN-Mish/Mish/Sine)

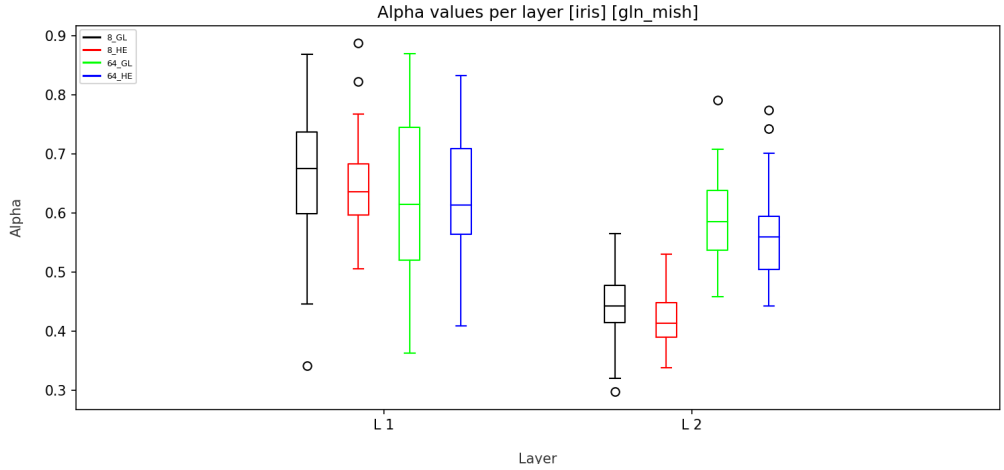


Figure 36 – *Box-Plot* of weight values α per layer (50 experiments) [IRIS - GLN-Mish]

As demonstrated in Table 13 and Figure 35, the Mish function achieved the highest average accuracy value, with 98.13, surpassing the GLN-Mish and Sine functions, in both batch variations of 8 and 64, and with the GL initialization method. This superior result was also observed in the precision, recall and f1-score metrics. However, as indicated by Table 16, the KS test did not allow rejecting the null hypothesis, suggesting that the model distributions are statistically similar in all tested configurations. All functions achieved a maximum accuracy of 100% and a minimum accuracy of at least 90%. The Mish function showed greater stability in accuracy in the batch configuration of 64 and GL initialization method. The model using the Mish function presented the shortest convergence time, with 773 epochs. There were no significant differences in the results between the different parameters used.

The distribution of α values is illustrated in Figure 36. The α values were predominantly above 0.5, in the first layer, indicating a preference for the global function, but with a high dispersion in 3 of the 4 variations tested. On the other hand, in the second layer, the value of α tended more towards the local Mish function in the models with a batch of 8, while it had a trend above 0.5 for the batch of 64.

As can be seen in Table 14 and Figure 37, the hyperbolic tangent function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 97.67, between the GLN-GCU and GCU functions. This performance was also reflected in the precision, recall and f1-score metrics. However, as we see in Table 16, the KS test indicates that the models with batch 8 and batch 64 variations with the GL initialization method presented a statistically similar distribution.

Table 14 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Iris - GLN-GCU/Tanh/GCU)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	95.60	6.51	100.0	56.67	0.14	0.14	946.86	357.46	95.6	95.6	95.6
GLN-GCU			96.93	3.42	100.0	90.00	0.08	0.05	**812.92**	220.39	96.94	96.93	96.93
Tanh			**97.67**	2.95	100.0	90.00	0.10	0.04	848.88	220.14	**97.68**	**97.67**	**97.67**
GCU	8	HE	90.07	9.72	100.0	56.67	0.34	0.20	841.52	476.50	90.24	90.07	90.05
GLN-GCU			93.73	9.61	100.0	36.67	0.19	0.18	997.30	366.00	93.73	93.73	93.72
Tanh			97.07	3.41	100.0	90.00	0.10	0.05	896.40	224.05	97.07	97.07	97.06
GCU	64	GL	94.07	9.41	100.0	50.0	0.15	0.19	1124.22	400.08	94.09	94.07	94.06
GLN-GCU			97.27	3.21	100.0	90.0	0.08	0.05	1139.10	281.16	97.27	97.27	97.27
Tanh			97.13	3.16	100.0	90.0	**0.07**	0.05	966.24	164.35	97.13	97.13	97.13
GCU	64	HE	88.47	12.55	100.0	40.00	0.32	0.24	1079.18	496.64	88.86	88.47	88.49
GLN-GCU			87.13	19.37	100.0	13.33	0.28	0.33	1318.20	623.86	87.08	87.13	87.09
Tanh			97.07	3.48	100.0	90.00	0.08	0.06	1067.60	216.81	97.07	97.07	97.07

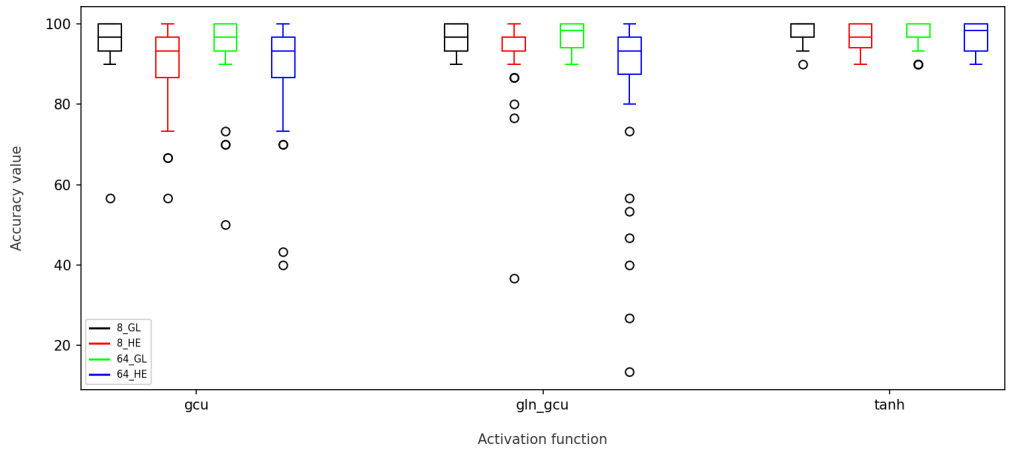


Figure 37 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (IRIS - GCU/GLN-GCU/Tanh)

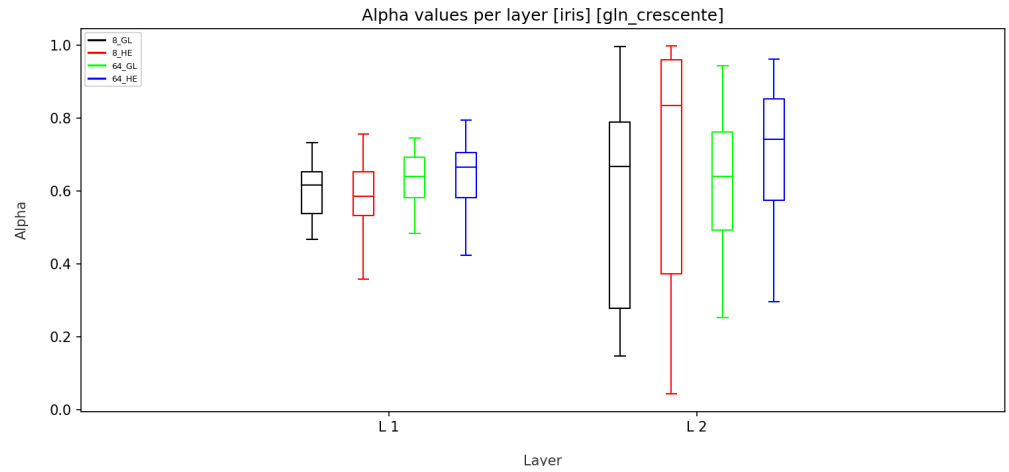


Figure 38 – *Box-Plot* of weight values α per layer (50 experiments) [IRIS - GLN-GCU]

All functions reached the maximum value of 100% accuracy. However, there was a lot of variation in the results for the models with GCU and GLN-GCU with the normal HE initialization method, which negatively affected these models, making convergence difficult in many samples. The model with the GLN-GCU function was the one with the shortest convergence time, 812 epochs, in the 8 GL variation.

The distribution of α values can be seen in Figure 38. In the first layer, regardless of the parameter variation, the values were above 0.5, indicating a choice for the global function. Also in the second layer, the value of α showed a tendency to choose the sine function, with a higher dispersion mainly for the batch size of 8.

Table 15 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Iris - GLN-GCU-ReLU/ReLU/GCU))

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	95.60	6.51	100.0	56.67	0.14	0.14	946.86	357.46	95.6	95.6	95.6
GLN-GCU-ReLU			96.47	3.33	100.0	90.00	0.08	0.05	765.56	213.37	96.48	96.47	96.46
ReLU			**97.80**	3.06	100.0	90.00	0.09	0.06	**598.72**	164.19	**97.8**	**97.8**	**97.8**
GCU	8	HE	90.07	9.72	100.0	56.67	0.34	0.20	841.52	476.50	90.24	90.07	90.05
GLN-GCU-ReLU			94.13	8.37	100.0	50.00	0.17	0.16	959.06	430.64	94.14	94.13	94.14
ReLU			97.53	2.92	100.0	90.00	0.09	0.05	693.84	221.92	97.55	97.53	97.53
GCU	64	GL	94.07	9.41	100.0	50.00	0.15	0.19	1124.22	400.08	94.09	94.07	94.06
GLN-GCU-ReLU			95.47	13.21	100.0	6.67	0.10	0.17	1062.96	328.03	95.49	95.47	95.48
ReLU			97.80	2.91	100.0	90.00	**0.07**	0.06	830.80	253.45	97.8	97.8	97.8
GCU	64	HE	88.47	12.55	100.0	40.00	0.32	0.24	1079.18	496.64	88.86	88.47	88.49
GLN-GCU-ReLU			89.93	15.14	100.0	36.67	0.27	0.31	1233.90	552.29	90.09	89.93	89.97
ReLU			97.47	2.74	100.0	90.00	0.07	0.05	942.88	229.27	97.47	97.47	97.47

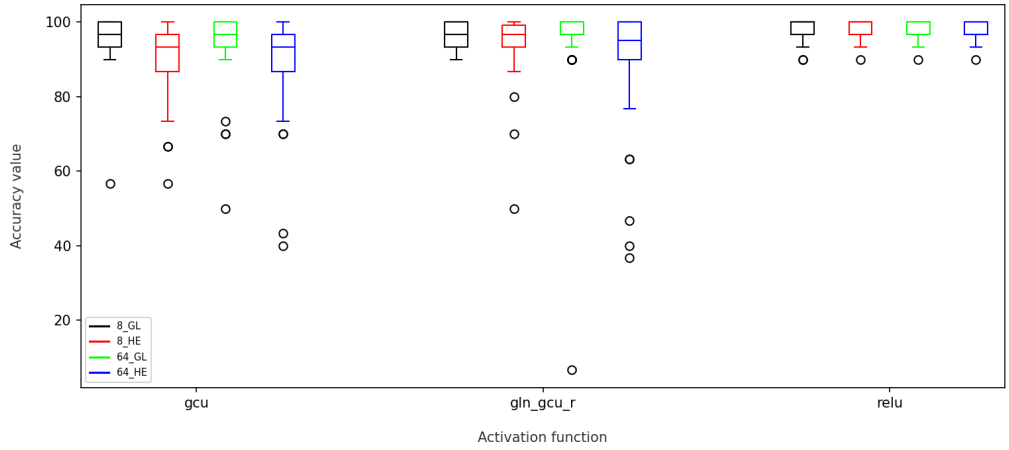


Figure 39 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (IRIS - GCU/GLN-GCU-R/ReLU)

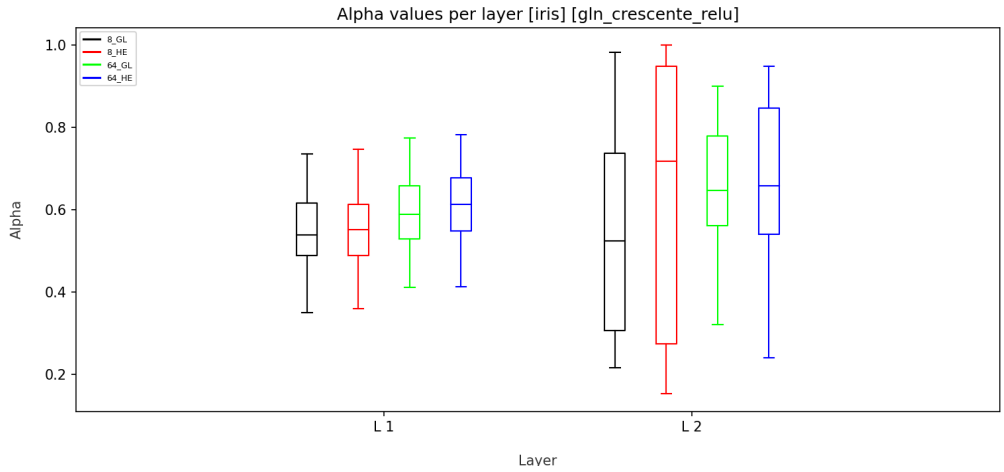


Figure 40 – *Box-Plot* of weight values α per layer (50 experiments) [IRIS - GLN-GCU-R]

Table 16 – KS test results for *accuracy* - Iris

batch size →	8								64							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	Yes	1.0	Yes	0.869	Yes	0.869	Yes	0.967	Yes	0.967	Yes	0.967	Yes	0.549	Yes	1.0
sin		-	Yes	0.717		-	Yes	1.0		-	Yes	1.0		-	Yes	0.272
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	Yes	0.396	Yes	1.0	Yes	0.967	Yes	0.998	Yes	0.396	Yes	1.0	Yes	0.869	Yes	0.717
relu		-	Yes	0.396		-	Yes	0.998		-	Yes	0.717		-	Yes	0.549
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	Yes	0.998	Yes	1.0	Yes	0.549	Yes	0.998	Yes	0.717	Yes	1.0	Yes	0.967	Yes	0.717
mish		-	Yes	0.717		-	Yes	0.967		-	Yes	0.272		-	Yes	0.112
act	gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh	
gcu	Yes	0.717	Yes	0.179	Yes	0.068	No	1.31e-03	Yes	0.549	Yes	0.272	Yes	0.869	No	1.31e-03
gln_gcu		-	Yes	0.998		-	Yes	0.112		-	Yes	1.0		-	No	5.84e-03
act	gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu	
gcu	Yes	1.0	Yes	0.068	Yes	0.112	No	5.82e-04	Yes	0.396	Yes	0.272	Yes	0.396	No	3.80e-05
gln_gcu_r		-	Yes	0.112		-	Yes	0.068		-	Yes	0.967		-	No	2.17e-02

As can be seen in Table 15 and Figure 39, the ReLU function was the one that obtained the highest average accuracy value, 97.80, between the GLN-ReLU and Seno functions, both in the same batch variation at 8 and 64, and in the GL initialization method. This performance was also reflected in the precision, recall and f1-score metrics. However, as we see in Table 16, the KS test indicates that the models with batch 8 and batch 64 variations with the GL initialization method presented a statistically similar distribution. All functions reached the maximum value of 100% accuracy. There was also a lot of variation in the results for the models with GCU and GLN-GCU-R with the normal HE initialization method, making convergence difficult in many samples. The model with the ReLU function was the one with the shortest convergence time and greatest stability,

598 epochs with a standard deviation of 164.

The distribution of α values can be seen in Figure 40. In the first layer, the values tended to be slightly above 0.5, indicating a slight choice for the global function. In the second layer, the values were much more dispersed, especially for the batch 8 variation, reaching values close to 1 and 0, indicating that at times this layer predominantly chose the global function, at times it preferred the local function.

4.2.1.3 Wine

Tables 17, 18, 19, 20 and 21 present the results, for the Wine dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine functions; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function; GLN-GCU and the functions that compose it, the Tanh function and GCU; GLN-GCU-R and the functions that compose it, the ReLU and GCU function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. The data was grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summary view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 41, 43, 45, 47 and 49. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 42, 44, 46, 48 and 50.

According to Table 17 and Figure 41, the hyperbolic tangent function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 97.30, among the GLN and Sine functions, also having better performance in the precision, recall and f1-score metrics. Despite this higher average, according to Table 22, all models present a statistically similar distribution in all variations.

Table 17 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Wine - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	8	GL	96.91	2.80	100.0	88.89	0.08	0.05	590.72	221.40	97.02	96.91	96.92
Sin			96.86	3.34	100.0	86.11	0.09	0.05	**547.54**	147.67	96.91	96.85	96.86
Tanh			**97.30**	2.61	100.0	91.43	**0.08**	0.05	549.18	232.91	**97.37**	**97.3**	**97.31**
GLN	8	HE	96.97	3.47	100.0	80.56	0.09	0.07	584.70	185.45	97.06	96.97	96.98
Sin			95.68	3.06	100.0	86.11	0.12	0.09	622.68	171.58	95.84	95.67	95.7
Tanh			96.86	2.63	100.0	91.43	0.09	0.06	561.10	236.94	97.0	96.85	96.87
GLN	64	GL	96.28	2.53	100.0	88.57	0.11	0.08	923.58	378.47	96.34	96.29	96.3
Sin			96.67	2.27	100.0	91.43	0.09	0.06	870.76	309.39	96.74	96.69	96.69
Tanh			96.63	1.97	100.0	91.43	0.11	0.06	929.16	410.38	96.67	96.63	96.64
GLN	64	HE	95.90	3.28	100.0	85.71	0.10	0.08	997.52	371.50	95.95	95.9	95.9
Sin			96.12	3.68	100.0	85.71	0.11	0.10	1026.00	335.89	96.22	96.12	96.14
Tanh			96.69	2.32	100.0	91.43	0.11	0.07	1020.06	396.44	96.74	96.69	96.69

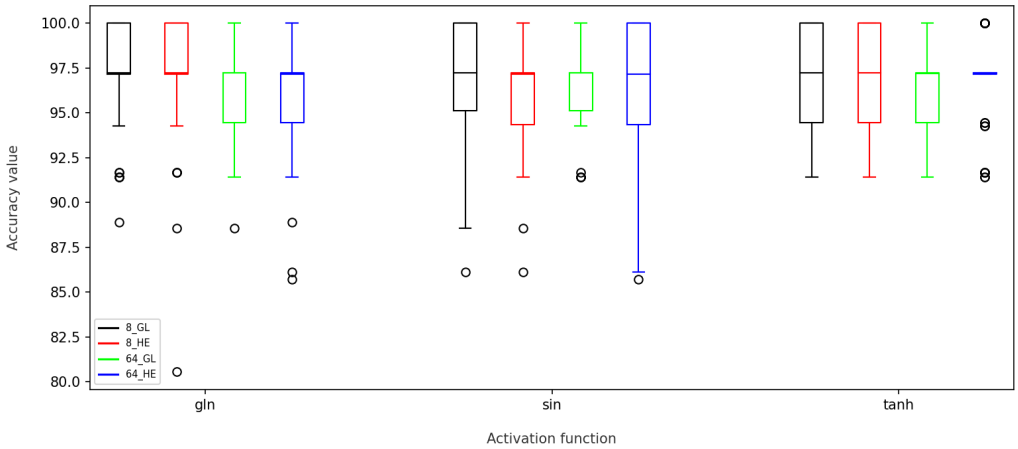


Figure 41 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Wine - GLN/Tanh/Sine)

All functions managed to reach the maximum value of 100% accuracy. The model with the sine function was the one that presented the shortest convergence time, 547 epochs. No significant difference is noted in the results in the different parameters used except for the significant difference in the convergence speed when the batch size varies.

The distribution of α values can be seen in Figure 42. In the first layer, regardless of the variation of the parameters, the values were slightly above 0.5, indicating a slight bias towards the global function. However, in the second layer, there was a clear difference in the α values in relation to the batch size. α tended strongly towards the local Tanh function in the models with the batch size parameter at 8, while it was slightly below 0.5 for the batch size parameter of 64.

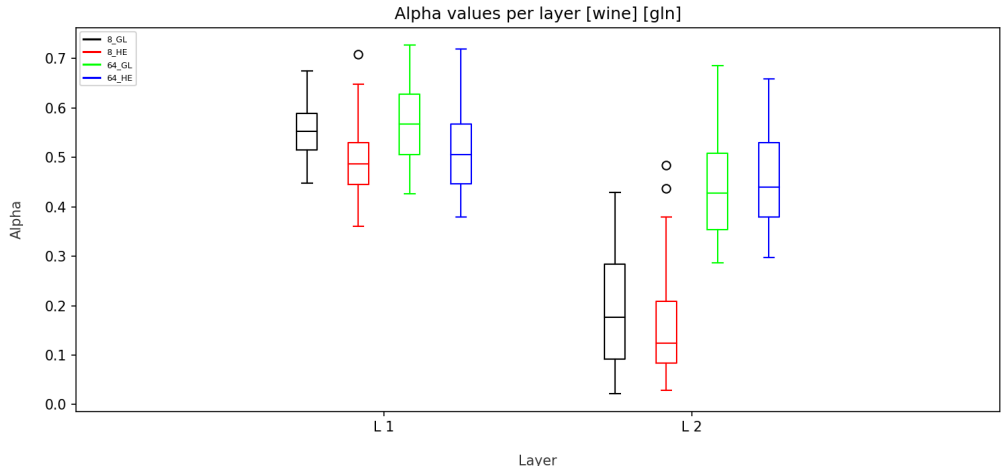


Figure 42 – *Box-Plot* of weight values α per layer (50 experiments) [Wine - GLN]

Table 18 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Wine - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	8	GL	96.23	3.09	100.0	88.89	0.09	0.06	**524.34**	203.88	96.39	96.24	96.26
ReLU			96.36	2.73	100.0	88.89	0.09	0.06	704.74	260.29	96.49	96.35	96.37
Sin			**96.86**	3.34	100.0	86.11	0.09	0.05	547.54	147.67	**96.91**	**96.85**	**96.86**
GLN-ReLU	8	HE	96.70	2.74	100.0	88.89	0.09	0.05	602.64	179.93	96.81	96.69	96.7
ReLU			96.01	3.06	100.0	88.57	0.10	0.06	680.74	200.42	96.09	96.01	96.03
Sin			95.68	3.06	100.0	86.11	0.12	0.09	622.68	171.58	95.84	95.67	95.7
GLN-ReLU	64	GL	96.68	2.61	100.0	88.57	0.11	0.08	936.12	388.61	96.78	96.69	96.69
ReLU			96.57	2.92	100.0	86.11	0.10	0.07	1047.76	391.75	96.64	96.57	96.59
Sin			96.67	2.27	100.0	91.43	**0.09**	0.06	870.76	309.39	96.74	96.69	96.69
GLN-ReLU	64	HE	96.75	2.28	100.0	91.67	0.10	0.07	947.5	347.10	96.8	96.74	96.75
ReLU			96.80	2.53	100.0	88.89	0.10	0.09	1239.9	420.35	96.82	96.8	96.8
Sin			96.12	3.68	100.0	85.71	0.11	0.10	1026.0	335.89	96.22	96.12	96.14

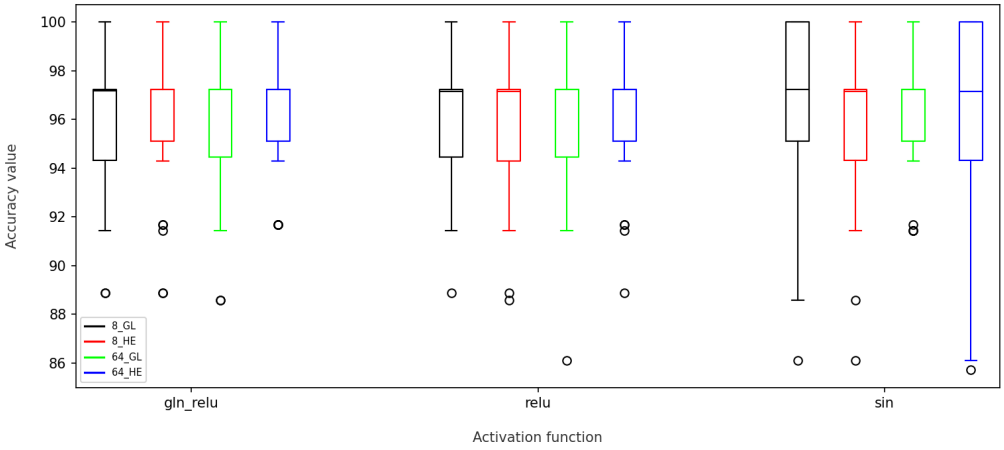


Figure 43 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Wine - GLN-ReLU/ReLU/Sine)

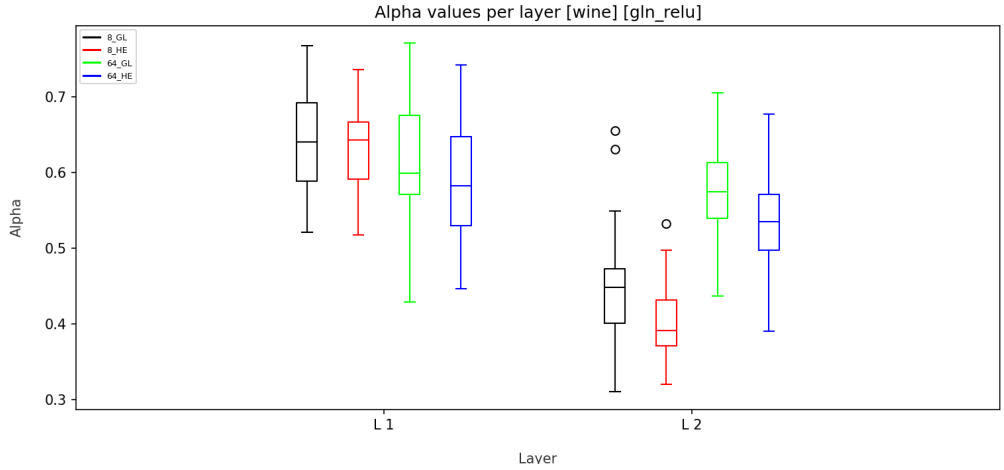


Figure 44 – *Box-Plot* of weight values α per layer (50 experiments) [Wine - GLN-ReLU]

According to Table 18 and Figure 43, the Sine function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 96.86, among the GLN-ReLU and ReLU functions, also having better performance in the precision, recall and f1-score metrics. According to Table 22, all models however present a statistically similar distribution in all variations. All functions also managed to reach the maximum value of 100% accuracy. The model with the GLN-ReLU function was the one that presented the shortest convergence time, 524 epochs. A significant difference in the results is only noted in the convergence speed when the batch size varies.

The distribution of α values can be seen in Figure 44. The values in the first layer were above 0.5, regardless of the variation of the parameters, indicating a choice for the global function. In contrast, in the second layer, there was a clear difference in the values of α in relation to the batch size. α tended towards the local ReLU function in the models with the batch size parameter at 8, while it was slightly above 0.5 for the size 64.

As seen in Table 19 and Figure 45, the Mish function, in the batch variation equal to 64 and GL initialization method, was the one that obtained the highest average accuracy value, 97.64, among the GLN-Mish and Sin functions, also having better performance in the precision, recall and f1-score metrics. Even so, all models present a statistically similar distribution in all variations, according to Table 22. All functions also managed to reach the maximum value of 100% accuracy. With 542 epochs, the model with the GLN-Mish function was the one that presented the shortest convergence time. Here, a significant difference in the results is also noted only in the convergence speed when the batch size is changed.

Table 19 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Wine - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	8	GL	96.91	2.29	100.0	91.43	0.09	0.05	**542.46**	188.51	97.0	96.91	96.92
Mish			96.86	2.91	100.0	88.89	0.09	0.05	557.36	138.09	96.94	96.85	96.87
Sin			96.86	3.34	100.0	86.11	0.09	0.05	547.54	147.67	96.91	96.85	96.86
GLN-Mish	8	HE	96.35	3.01	100.0	88.57	0.09	0.06	569.60	162.17	96.44	96.35	96.36
Mish			96.58	2.71	100.0	86.11	0.10	0.06	641.10	196.59	96.68	96.57	96.58
Sin			95.68	3.06	100.0	86.11	0.12	0.09	622.68	171.58	95.84	95.67	95.7
GLN-Mish	64	GL	96.57	2.38	100.0	88.57	0.10	0.06	862.64	371.25	96.64	96.57	96.58
Mish			**97.64**	2.16	100.0	91.43	**0.07**	0.06	1068.98	381.16	**97.69**	**97.64**	**97.64**
Sin			96.67	2.27	100.0	91.43	0.09	0.06	870.76	309.39	96.74	96.69	96.69
GLN-Mish	64	HE	96.29	2.57	100.0	88.57	0.12	0.10	945.92	384.34	96.41	96.29	96.31
Mish			97.13	1.94	100.0	91.43	0.08	0.06	1183.00	440.30	97.19	97.13	97.14
Sin			96.12	3.68	100.0	85.71	0.11	0.10	1026.00	335.89	96.22	96.12	96.14

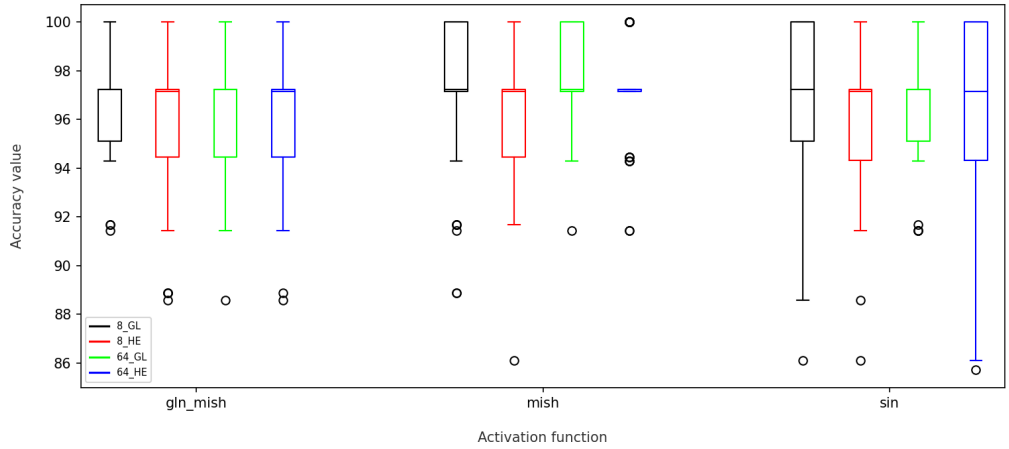


Figure 45 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Wine - GLN-Mish/Mish/Sine)

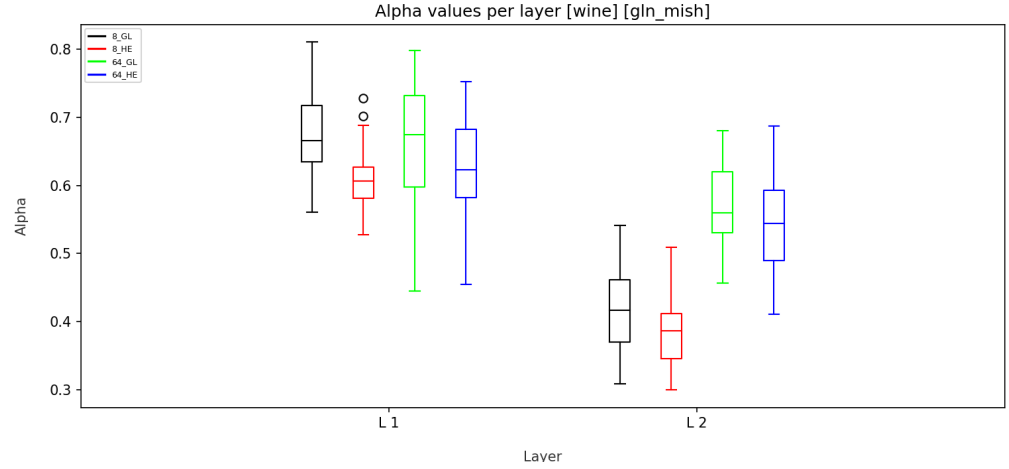


Figure 46 – *Box-Plot* of weight values α per layer (50 experiments) [Wine - GLN-Mish]

The distribution of α values can be seen in Figure 46. In all parameter variations, the values in the first layer were above 0.5, which indicates a choice for the global function in this first layer. While in the second layer, the batch size leads to different choices in the values of α. This tended towards the local function Mish in the models with the batch size parameter at 8, while it was slightly above 0.5 for the size 64.

Table 20 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Wine - GLN-GCU/Tanh/GCU)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	90.97	7.92	100.0	63.89	0.25	0.16	786.98	260.88	90.96	90.96	90.96
GLN-GCU			95.11	3.04	100.0	88.89	0.14	0.06	691.70	233.24	95.18	95.11	95.13
Tanh			**97.30**	2.61	100.0	91.43	**0.08**	0.05	**549.18**	232.91	**97.37**	**97.3**	**97.31**
GCU	8	HE	80.24	13.38	100.0	30.56	0.52	0.30	903.54	388.75	80.25	80.17	80.15
GLN-GCU			88.29	12.88	100.0	41.67	0.29	0.26	939.74	355.69	88.27	88.26	88.26
Tanh			96.86	2.63	100.0	91.43	0.09	0.06	561.10	236.94	97.0	96.85	96.87
GCU	64	GL	81.65	15.75	100.0	27.78	0.44	0.28	1072.10	494.27	81.71	81.63	81.65
GLN-GCU			90.87	8.41	100.0	48.57	0.22	0.17	1070.26	462.60	90.99	90.9	90.91
Tanh			96.63	1.97	100.0	91.43	0.11	0.06	929.16	410.38	96.67	96.63	96.64
GCU	64	HE	61.40	15.98	91.43	17.14	0.87	0.28	823.06	518.41	62.14	61.4	61.54
GLN-GCU			61.70	20.40	97.14	27.78	0.81	0.36	858.42	586.08	62.23	61.63	61.74
Tanh			96.69	2.32	100.00	91.43	0.11	0.07	1020.06	396.44	96.74	96.69	96.69

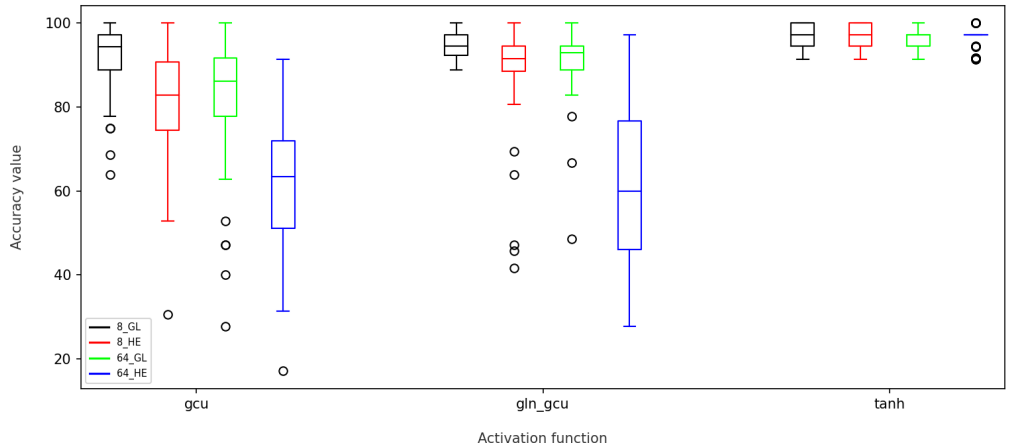


Figure 47 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Wine - GCU/GLN-GCU/Tanh)

As shown in Table 20 and Figure 47, the hyperbolic tangent function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 97.30, among the GLN-GCU and GCU functions and the shortest convergence time in this same parameter variation, 549. This performance was

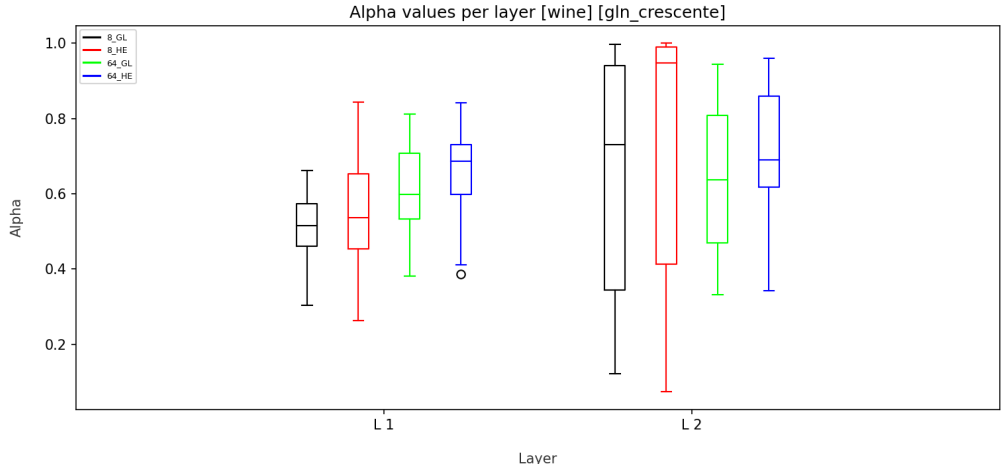


Figure 48 – *Box-Plot* of weight values α per layer (50 experiments) [Wine - GLN-GCU]

also reflected in the precision, recall and f1-score metrics. There was no similarity between the models with GLN-CGU and the Tanh function, as shown in the results presented in the KS test that we can see in Table 22. Only the GLN-GCU and GCU models did not reach the maximum value of 100% accuracy when the batch had the value 64 and the initialization method was HE, with a wide variation in the results of the samples in these parameter values.

The distribution of the α values can be seen in Figure 48. In the first layer, regardless of the variation of the parameters, the values were above 0.5, indicating a choice for the global function. Also in the second layer the value of α showed a tendency to choose the GCU function, with a higher dispersion mainly for the batch size of 8.

Table 21 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Wine - GLN-GCU-ReLU/ReLU/GCU))

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	90.97	7.92	100.0	63.89	0.25	0.16	786.98	260.88	90.96	90.96	90.96
GLN-GCU-ReLU			94.94	3.02	100.0	88.57	0.14	0.08	**667.96**	202.08	95.0	94.94	94.95
ReLU			96.36	2.73	100.0	88.89	**0.09**	0.06	704.74	260.29	96.49	96.35	96.37
GCU	8	HE	80.24	13.38	100.0	30.56	0.52	0.30	903.54	388.75	80.25	80.17	80.15
GLN-GCU-ReLU			88.54	16.89	100.0	25.71	0.28	0.31	949.66	406.17	88.58	88.54	88.54
ReLU			96.01	3.06	100.0	88.57	0.10	0.06	680.74	200.42	96.09	96.01	96.03
GCU	64	GL	81.65	15.75	100.0	27.78	0.44	0.28	1072.10	494.27	81.71	81.63	81.65
GLN-GCU-ReLU			90.92	8.44	100.0	63.89	0.24	0.20	1147.74	295.73	90.99	90.9	90.92
ReLU			96.57	2.92	100.0	86.11	0.10	0.07	1047.76	391.75	96.64	96.57	96.59
GCU	64	HE	61.40	15.98	91.43	17.14	0.87	0.28	823.06	518.41	62.14	61.4	61.54
GLN-GCU-ReLU			70.87	23.05	97.22	25.71	0.62	0.40	907.26	581.10	71.22	70.84	70.92
ReLU			**96.80**	2.53	100.00	88.89	0.10	0.09	1239.90	420.35	**96.82**	**96.8**	**96.8**

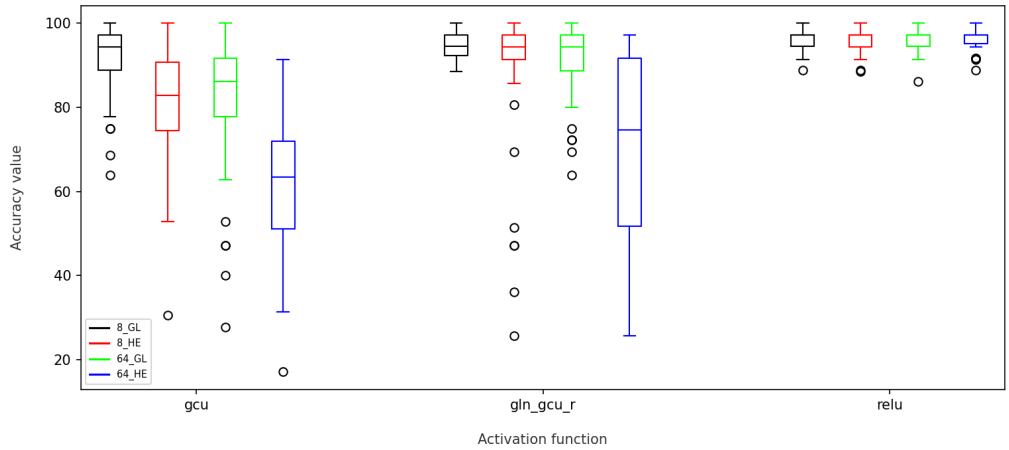


Figure 49 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Wine - GCU/GLN-GCU-R/ReLU)

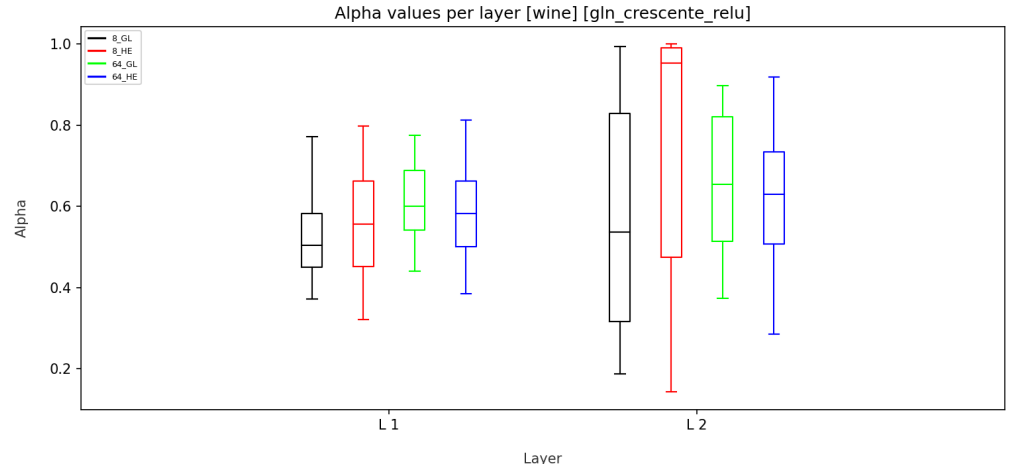


Figure 50 – *Box-Plot* of weight values α per layer (50 experiments) [Wine - GLN-GCU-R]

As we can see in Table 21 and Figure 49, the ReLU function, in the batch variation equal to 64 and HE initialization method, was the one that obtained the highest average accuracy value, 96.80, among the GLN-GCU-R and GCU functions. This performance was also reflected in the precision, recall and f1-score metrics. There was no similarity between the models with GLN-CGU and the ReLU function in the same parameter variations as seen in Table 22. However, although the KS test was not performed between distributions with different parameters, we can see in Figure 49 great similarity of the distributions of the ReLU models with the GLN-GCU-R model in the 8 GL variation, indicating that they probably have the same distribution. In this experiment, only the GLN-GCU and GCU models did not reach the maximum accuracy value of 100% when the batch size was 64 and the initialization method was HE, with a wide variation in the results of the samples

at these parameter values.

The distribution of the α values can be seen in Figure 50. In the first layer, regardless of the variation in the parameters, the values were above 0.5, indicating a choice for the global function and relatively similar in all parameter variations. Also, the value of α, in the second layer, tended to choose the GCU function, with a higher dispersion mainly for the batch size of 8.

Table 22 – KS test results for *accuracy* - Wine

batch size →	8								64							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	Yes	1.0	Yes	0.967	Yes	0.112	Yes	0.549	Yes	1.0	Yes	0.998	Yes	0.717	Yes	0.717
sin		-	Yes	1.0		-	Yes	0.396		-	Yes	1.0		-	Yes	0.396
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	Yes	0.998	Yes	0.717	Yes	0.717	Yes	0.717	Yes	0.967	Yes	1.0	Yes	1.0	Yes	0.717
relu		-	Yes	0.396		-	Yes	0.998		-	Yes	0.998		-	Yes	0.869
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	Yes	1.0	Yes	0.869	Yes	0.998	Yes	0.967	Yes	0.272	Yes	1.0	Yes	0.179	Yes	0.869
mish		-	Yes	1.0		-	Yes	0.869		-	Yes	0.272		-	Yes	0.179
act	gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh	
gcu	No	3.92e-02	No	3.80e-05	No	1.58e-06	No	2.08e-14	No	2.83e-03		1.41e-13	Yes	0.549	No	1.98e-27
gln_gcu		-	No	1.31e-03		-	No	4.93e-07		-		1.58e-06		-	No	7.77e-23
act	gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu	
gcu	No	3.92e-02	No	5.82e-04	No	1.45e-07	No	8.76e-13	No	5.84e-03		2.67e-11	No	1.31e-03	No	9.81e-26
gln_gcu_r		-	Yes	0.179		-	Yes	0.068		-		2.83e-03		-	No	5.02e-12

4.2.1.4 Yeast

Tables 23, 24, 25, 26 and 27 present the results, for the Yeast dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine functions; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function; GLN-GCU and the functions that compose it, the Tanh function and GCU; GLN-GCU-R and the functions that compose it, the ReLU and GCU function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. The data was grouped by the parameter variations performed: batch size,

8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summary view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 51, 53, 55, 57 and 59. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 42, 44, 46, 48, 50.

Table 23 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Yeast - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	8	GL	58.27	2.33	62.96	53.87	1.12	0.06	181.28	39.95	60.65	58.27	59.14
Sin			58.11	2.74	64.31	52.19	1.12	0.06	186.64	35.16	60.27	58.11	58.94
Tanh			58.26	2.59	63.30	52.86	**1.11**	0.05	205.64	54.83	60.54	58.26	59.09
GLN	8	HE	57.95	2.89	63.64	53.20	1.14	0.06	156.90	20.14	60.35	57.95	58.9
Sin			57.12	2.76	62.29	50.84	1.18	0.07	151.74	19.41	59.48	57.12	58.03
Tanh			58.10	2.41	63.30	53.87	1.12	0.06	207.30	47.08	60.28	58.1	58.94
GLN	64	GL	58.36	2.54	62.96	53.20	1.17	0.07	153.32	25.84	60.47	58.36	59.16
Sin			58.51	2.48	62.29	53.87	1.17	0.07	**145.80**	22.08	**60.73**	58.51	**59.37**
Tanh			**58.59**	2.53	63.30	53.20	1.13	0.07	193.58	37.65	**60.73**	**58.58**	**59.37**
GLN	64	HE	57.73	2.71	62.29	51.85	1.21	0.07	168.92	16.47	60.26	57.73	58.69
Sin			56.98	2.23	60.94	52.19	1.23	0.06	184.48	13.06	59.65	56.98	57.99
Tanh			58.03	2.68	62.63	52.53	1.15	0.06	197.58	29.28	60.4	58.03	58.88

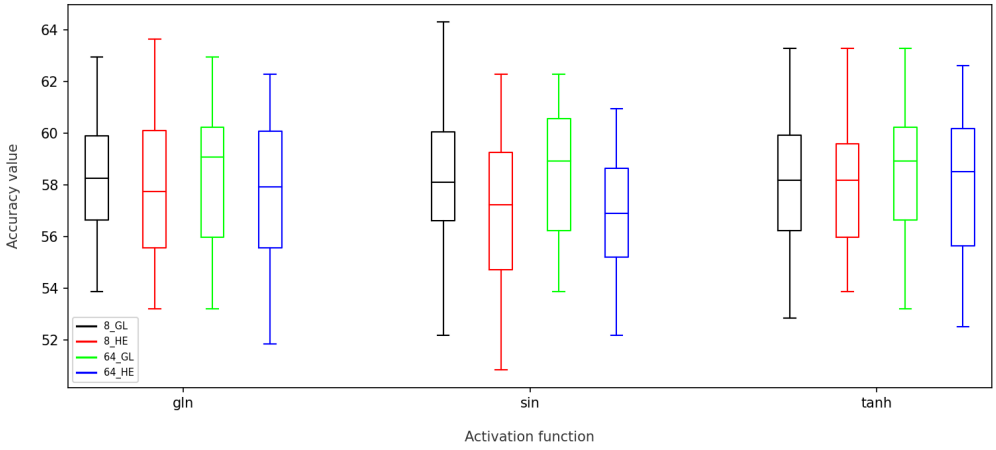


Figure 51 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Yeast - GLN/Tanh/Sine)

According to Table 23 and Figure 51, the hyperbolic tangent function, in the batch

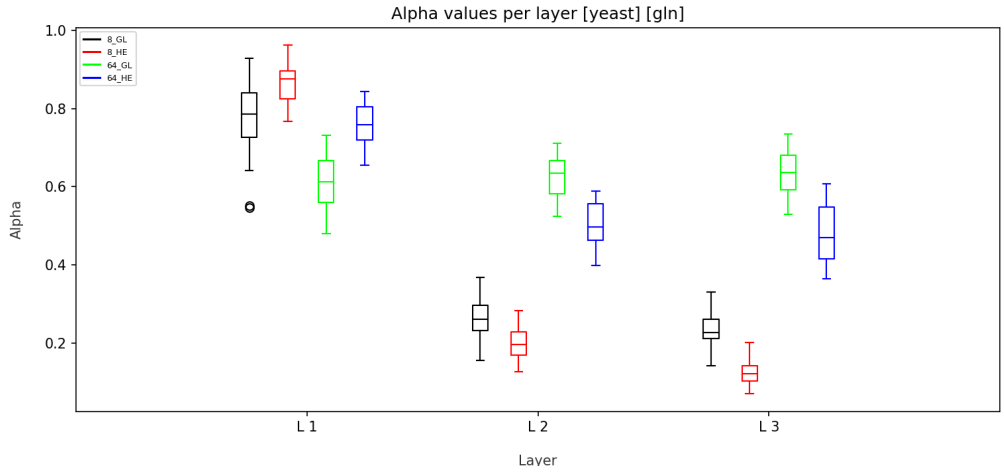


Figure 52 – *Box-Plot* of weight values α per layer (50 experiments) [Yeast - GLN]

variation equal to 64 and GL initialization method, was the one that obtained the highest average accuracy value, 58.59, among the GLN and Sine functions, also having better performance in the precision, recall and f1-score metrics. Despite this higher average, according to Table 28, all models present a statistically similar distribution in all variations. The model with the sine function was the one that presented the shortest convergence time, 145 epochs. No significant difference is noted in the results in the different parameters used.

The distribution of the α values can be seen in Figure 52. A certain difference in the values according to the parameters can be seen. In the first layer, the distribution of values is above 0.5, indicating a preference for the global characteristic, and more so when the batch value is 8. Otherwise, in the next two layers, the distribution is inclined to choose the local characteristic for that same batch value and to slightly choose the global characteristic when the value is 64.

According to Table 24 and Figure 53, the GLN-ReLU function, in the batch variation equal to 64 and GL initialization method, was the one that obtained the highest average accuracy value, 58.63, among the ReLU and Sine functions, also having better performance in the precision, recall and f1-score metrics. According to Table 28, all models however present a statistically similar distribution in all variations. The model with the Sine function was the one that presented the shortest convergence time, 145 epochs in this same variation of parameters. There is no great difference in the results due to the change in parameters.

The distribution of the α values can be seen in Figure 54. A certain difference in

Table 24 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Yeast - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	8	GL	58.45	2.36	62.96	53.54	1.12	0.06	172.26	36.69	60.79	58.45	59.25
ReLU			58.27	2.51	63.30	53.54	1.14	0.07	153.12	24.84	60.34	58.27	58.98
Sin			58.11	2.74	64.31	52.19	1.12	0.06	186.64	35.16	60.27	58.11	58.94
GLN-ReLU	8	HE	58.13	2.25	62.63	54.21	1.12	0.06	148.14	19.83	60.58	58.13	59.05
ReLU			58.28	2.99	64.31	50.51	1.15	0.08	151.34	25.98	60.47	58.28	59.03
Sin			57.12	2.76	62.29	50.84	1.18	0.07	151.74	19.41	59.48	57.12	58.03
GLN-ReLU	64	GL	**58.63**	2.36	62.96	52.53	1.14	0.08	150.20	28.57	**60.82**	**58.63**	**59.43**
ReLU			58.42	2.73	64.31	52.53	1.13	0.08	160.86	25.43	60.44	58.42	59.18
Sin			58.51	2.48	62.29	53.87	1.17	0.07	**145.80**	22.08	60.73	58.51	59.37
GLN-ReLU	64	HE	58.19	2.44	61.62	53.20	1.16	0.08	174.20	16.71	60.48	58.19	59.08
ReLU			58.47	2.25	62.29	53.20	**1.12**	0.07	178.32	20.86	60.6	58.47	59.29
Sin			56.98	2.23	60.94	52.19	1.23	0.06	184.48	13.06	59.65	56.98	57.99

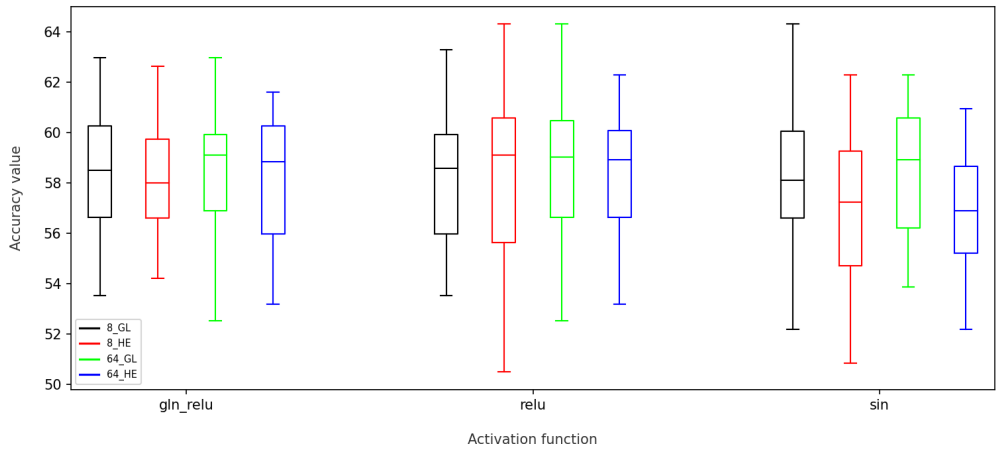


Figure 53 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Yeast - GLN-ReLU/ReLU/Sine)

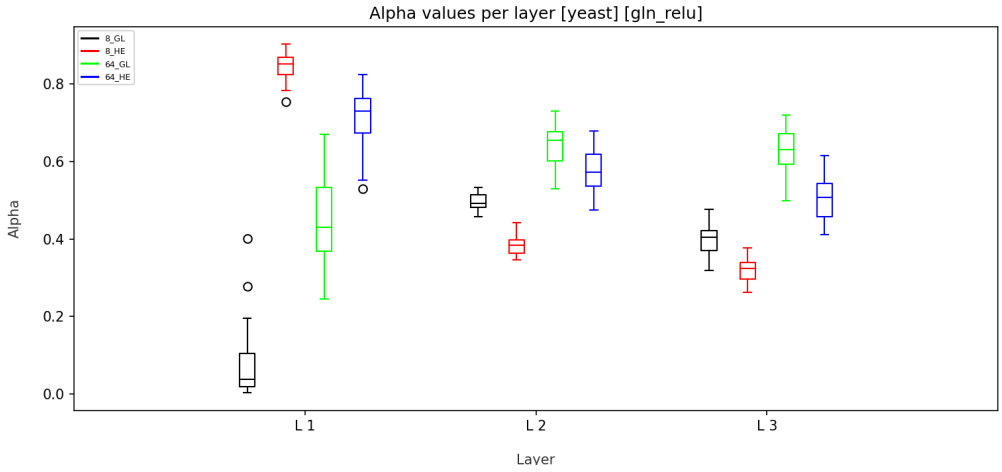


Figure 54 – *Box-Plot* of weight values α per layer (50 experiments) [Yeast - GLN-ReLU]

the values according to the parameters can be seen. In the first layer, the distribution of values is below 0.5, indicating a preference for the local feature when the initialization method is GL, while in the HE method, the preference is reversed. In the next two layers, the distribution is inclined to choose the local feature when the batch value is 8 and to choose the global feature when the value is 64.

Table 25 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Yeast - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	8	GL	58.10	2.18	62.63	52.53	**1.11**	0.06	169.72	29.94	60.28	58.1	58.91
Mish			58.90	2.69	63.97	52.19	1.12	0.06	171.68	48.02	60.83	58.89	59.52
Sin			58.11	2.74	64.31	52.19	1.12	0.06	186.64	35.16	60.27	58.11	58.94
GLN-Mish	8	HE	58.20	2.47	62.96	52.86	1.12	0.06	**144.24**	21.64	60.56	58.19	59.09
Mish			58.43	2.71	64.31	52.86	1.13	0.06	185.16	48.86	60.68	58.43	59.18
Sin			57.12	2.76	62.29	50.84	1.18	0.07	151.74	19.41	59.48	57.12	58.03
GLN-Mish	64	GL	**59.16**	2.46	64.31	54.55	1.13	0.07	148.14	22.58	**61.3**	**59.16**	**59.97**
Mish			58.99	2.11	61.95	53.54	1.11	0.07	174.18	31.44	61.08	58.99	59.78
Sin			58.51	2.48	62.29	53.87	1.17	0.07	145.80	22.08	60.73	58.51	59.37
GLN-Mish	64	HE	58.15	2.56	62.29	53.20	1.16	0.07	173.60	15.60	60.4	58.15	59.02
Mish			58.99	2.85	63.64	52.86	1.12	0.07	179.80	20.40	61.2	58.99	59.8
Sin			56.98	2.23	60.94	52.19	1.23	0.06	184.48	13.06	59.65	56.98	57.99

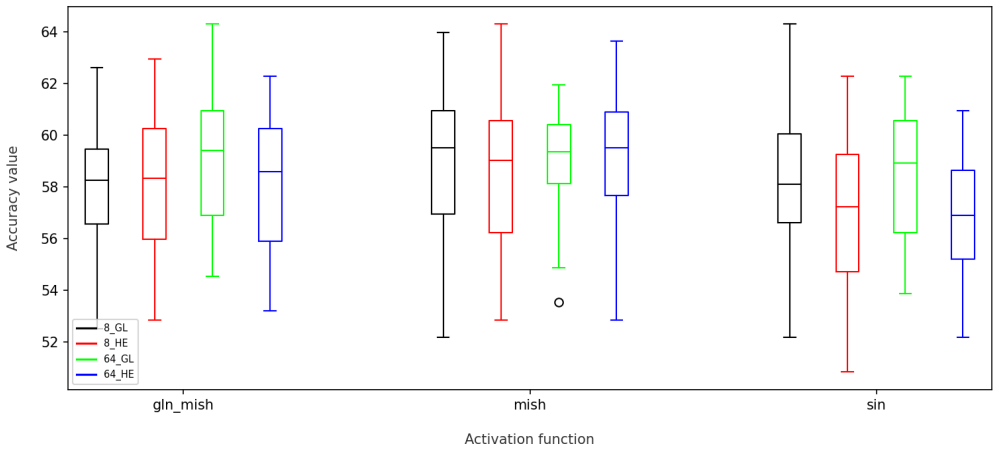


Figure 55 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Yeast - GLN-Mish/Mish/Sine)

As seen in Table 25 and Figure 55, the GLN-Mish function, in the batch variation equal to 64 and GL initialization method, was the one that obtained the highest average accuracy value, 59.16, among the Mish and Sine functions, also having better performance in the precision, recall and f1-score metrics. It was also the one that performed best in

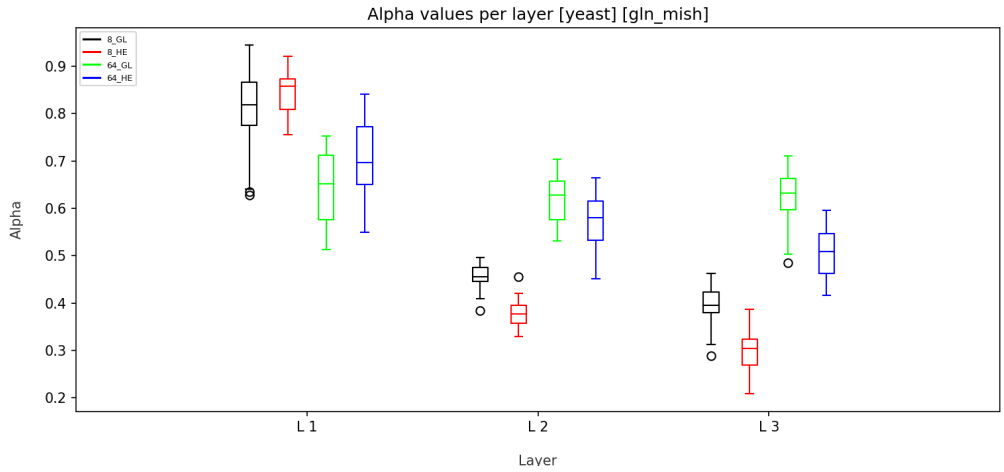


Figure 56 – *Box-Plot* of weight values α per layer (50 experiments) [Yeast - GLN-Mish]

relation to the convergence speed with 144 epochs. All models present a statistically similar distribution in all variations, according to Table 28. No significant difference is observed in the results when the parameters are changed.

The distribution of the α values can be seen in Figure 56. A certain difference in the values can be seen according to the parameters. In the first layer the distribution of values is above 0.5 indicating a preference for the global characteristic and more predominantly when the batch value is 8. Otherwise, in the next two layers, the distribution is inclined to choose the local characteristic when the batch value is 8 and to slightly choose the global characteristic when the value is 64.

Table 26 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Yeast - GLN-GCU/Tanh/GCU)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	55.32	2.51	59.26	49.16	1.21	0.05	294.90	84.68	59.86	55.32	56.83
GLN-GCU			57.83	2.82	62.29	51.52	1.12	0.06	**189.74**	35.21	60.34	57.83	58.78
Tanh			58.26	2.59	63.30	52.86	**1.11**	0.05	205.64	54.83	60.54	58.26	59.09
GCU	8	HE	54.56	3.63	59.60	41.89	1.23	0.07	445.12	64.64	58.28	54.56	55.91
GLN-GCU			54.35	2.90	58.92	46.13	1.24	0.07	400.96	87.97	57.68	54.35	55.64
Tanh			58.10	2.41	63.30	53.87	1.12	0.06	207.30	47.08	60.28	58.1	58.94
GCU	64	GL	54.93	3.31	59.60	45.79	1.22	0.09	276.68	77.04	59.34	54.93	56.24
GLN-GCU			57.99	2.45	62.16	52.86	1.17	0.07	233.94	36.53	60.66	57.99	59.02
Tanh			**58.59**	2.53	63.30	53.20	1.13	0.07	193.58	37.65	**60.73**	**58.58**	**59.37**
GCU	64	HE	50.59	3.40	57.58	42.09	1.30	0.09	646.60	104.72	60.19	50.59	52.76
GLN-GCU			53.21	3.88	59.60	43.10	1.29	0.09	547.24	133.77	58.91	53.21	54.67
Tanh			58.03	2.68	62.63	52.53	1.15	0.06	197.58	29.28	60.4	58.03	58.88

As shown in Table 26 and Figure 57, the hyperbolic tangent function, in the

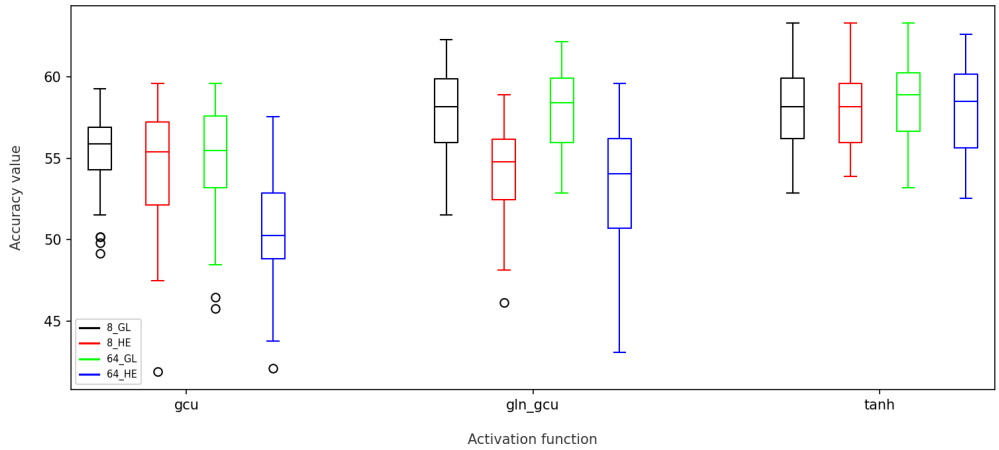


Figure 57 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Yeast - GCU/GLN-GCU/Tanh)

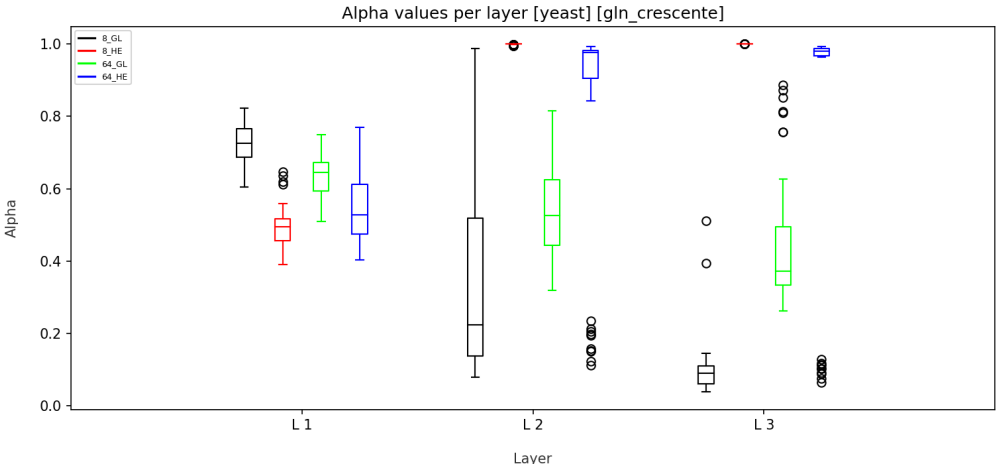


Figure 58 – *Box-Plot* of weight values α per layer (50 experiments) [Yeast - GLN-GCU]

batch variation equal to 64 and GL initialization method, was the one that obtained the highest average accuracy value, 58.59, among the GLN-GCU and GCU functions. This performance was also reflected in the precision, recall and f1-score metrics. However, the shortest convergence time, with the value of 189 epochs, was the GLN-CGU in the 8 GL variation. According to the results presented in the KS test presented in Table 28, there was similarity between the models with the GLN-CGU and the Tanh function for the batch parameters equal to 8 and 64 and GL initialization method.

The distribution of α values can be seen in Figure 58. In the first layer, regardless of the variation of the parameters, the distribution of values was slightly above 0.5, indicating a slight preference for the global function. In the second and third layers, there was a strong inclination towards one of the characteristics, either global or local, depending on

the parameters.

Table 27 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Yeast - GLN-GCU-ReLU/ReLU/GCU))

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GCU	8	GL	55.32	2.51	59.26	49.16	1.21	0.05	294.90	84.68	59.86	55.32	56.83
GLN-GCU-ReLU			57.86	2.46	61.62	52.53	1.17	0.08	177.72	42.11	60.52	57.86	58.79
ReLU			58.27	2.51	63.30	53.54	1.14	0.07	153.12	24.84	60.34	58.27	58.98
GCU	8	HE	54.56	3.63	59.60	41.89	1.23	0.07	445.12	64.64	58.28	54.56	55.91
GLN-GCU-ReLU			54.99	2.72	60.47	49.83	1.25	0.08	346.66	97.80	57.82	54.99	56.02
ReLU			58.28	2.99	64.31	50.51	1.15	0.08	**151.34**	25.98	60.47	58.28	59.03
GCU	64	GL	54.93	3.31	59.60	45.79	1.22	0.09	276.68	77.04	59.34	54.93	56.24
GLN-GCU-ReLU			57.95	2.64	62.84	51.18	1.17	0.07	206.88	39.11	60.4	57.94	58.86
ReLU			58.42	2.73	64.31	52.53	1.13	0.08	160.86	25.43	60.44	58.42	59.18
GCU	64	HE	50.59	3.40	57.58	42.09	1.30	0.09	646.60	104.72	60.19	50.59	52.76
GLN-GCU-ReLU			54.68	3.53	60.61	45.12	1.24	0.09	504.06	144.83	58.98	54.68	55.84
ReLU			**58.47**	2.25	62.29	53.20	**1.12**	0.07	178.32	20.86	**60.6**	**58.47**	**59.29**

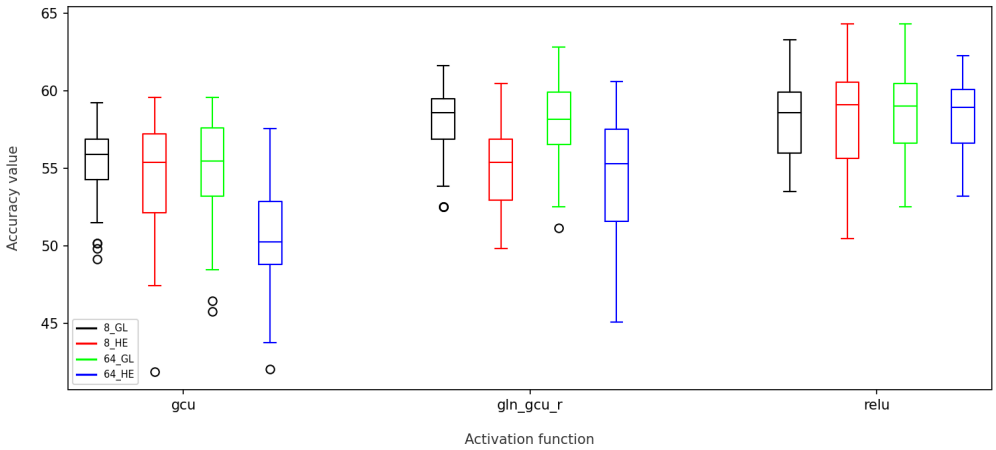


Figure 59 – *Box-Plot* of the accuracy values of the test set by activation function (50 experiments). (Yeast - GCU/GLN-GCU-R/ReLU)

As we can see in Table 27 and Figure 59, the ReLU function, in the batch variation equal to 8 and GL initialization method, was the one that obtained the highest average accuracy value, 58.47, among the GLN-GCU-R and GCU functions. This performance was also reflected in the precision, recall and f1-score metrics. The model with the ReLU function was the one that performed best in relation to convergence speed with 151 epochs. According to the results presented in the KS test presented in Table 28, there was similarity between the models with GLN-CGU and the ReLU function for the batch parameters equal to 8 and 64 and GL initialization method.

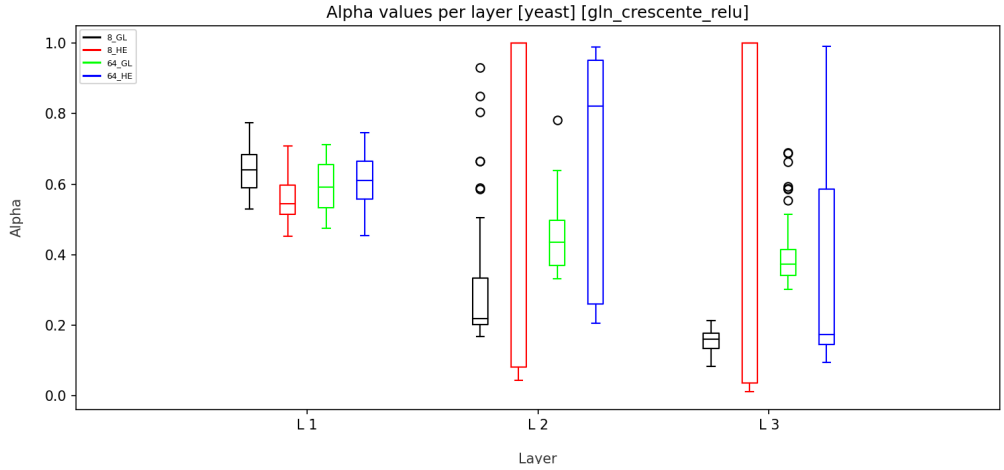


Figure 60 – *Box-Plot* of weight values α per layer (50 experiments) [Yeast - GLN-GCU-R]

The distribution of the α values can be seen in Figure 60. In the first layer, regardless of the variation of the parameters, the distribution of the values was also slightly above 0.5, indicating a slight choice for the global function. In the second and third layers, there was a prevalence in choosing the local ReLU function when the initialization method was GL and a high variability of the values with no defined prevalence when the method was HE.

Table 28 – KS test results for *accuracy* - Yeast

batch size →	8								64							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	Yes	0.998	Yes	1.0	Yes	0.549	Yes	0.967	Yes	1.0	Yes	0.869	Yes	0.396	Yes	0.967
sin		-	Yes	1.0		-	Yes	0.549		-	Yes	0.869		-	Yes	0.068
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	Yes	0.869	Yes	0.717	Yes	0.272	Yes	0.272	Yes	0.967	Yes	0.869	Yes	0.869	No	2.17e-02
relu		-	Yes	0.549		-	Yes	0.068		-	Yes	0.869		-	No	2.83e-03
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	Yes	0.068	Yes	0.998	Yes	0.869	Yes	0.179	Yes	0.869	Yes	0.549	Yes	0.272	Yes	0.068
mish		-	Yes	0.112		-	No	2.17e-02		-	Yes	0.272		-	No	2.46e-04
act	gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh		gln_gcu		tanh	
gcu	No	2.46e-04	No	1.39e-05	Yes	0.396	No	5.82e-04	No	1.31e-03	No	3.80e-05	No	1.31e-03	No	3.43e-16
gln_gcu		-	Yes	0.967		-	No	1.58e-06		-	Yes	0.549		-	No	1.39e-05
act	gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu		gln_gcu_r		relu	
gcu	No	1.39e-05	No	1.58e-06	Yes	0.967	No	4.81e-06	No	9.91e-05	No	3.80e-05	No	3.80e-05	No	3.17e-19
gln_gcu_r		-	Yes	0.717		-	No	4.81e-06		-	Yes	0.869		-	No	1.39e-05

4.2.2 Image Classification Problems

4.2.2.1 Mnist

The Tables 29, 30 and 31 present the results, for the Mnist dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine function; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. These results were grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summarized view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 61, 63 and 65. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 62 and 64, 66.

Table 29 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Mnist - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	32	GL	99.04	0.10	99.20	98.69	0.03	0.0	58.53	7.94	99.04	99.04	99.04
Sin			98.99	0.07	99.09	98.84	0.03	0.0	59.73	8.40	98.99	98.99	98.99
Tanh			**99.09**	0.14	99.29	98.75	**0.03**	0.0	**55.00**	11.51	**99.09**	**99.09**	**99.09**
GLN	32	HE	98.73	0.10	98.96	98.59	0.04	0.0	62.5	7.19	98.73	98.73	98.73
Sin			98.40	0.09	98.59	98.19	0.05	0.0	65.7	7.15	98.4	98.4	98.4
Tanh			98.76	0.07	98.90	98.62	0.04	0.0	63.1	5.01	98.76	98.76	98.76
GLN	128	GL	98.40	0.15	98.66	97.94	0.05	0.0	60.40	8.50	98.4	98.4	98.4
Sin			98.04	0.14	98.37	97.75	0.06	0.0	66.43	9.36	98.04	98.04	98.04
Tanh			98.43	0.10	98.60	98.21	0.05	0.0	65.10	6.92	98.43	98.43	98.43
GLN	128	HE	97.38	0.19	97.67	96.89	0.08	0.01	65.03	8.51	97.38	97.38	97.38
Sin			96.68	0.13	96.86	96.28	0.10	0.00	70.87	9.06	96.68	96.68	96.68
Tanh			97.51	0.16	97.76	97.11	0.08	0.01	64.47	7.82	97.51	97.51	97.51

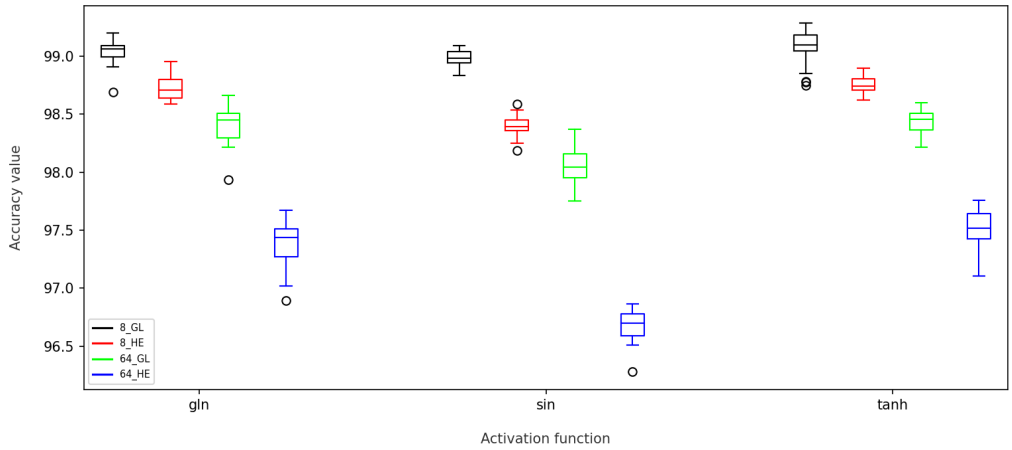


Figure 61 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (MNIST - GLN/Tanh/Sine)

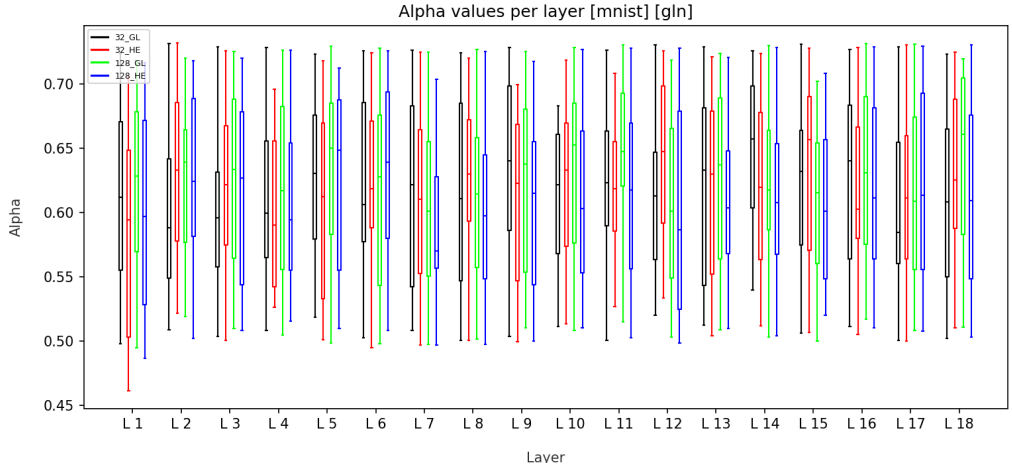


Figure 62 – *Box-Plot* of weight values α per layer (30 experiments) [MNIST - GLN]

As can be seen in Table 29, the hyperbolic tangent function was the one that obtained the highest average accuracy value, 99.09%, and the highest maximum value of 99.29%, between the GLN and Sine functions. This result was obtained for batch size 32 and the Glorot Normal initialization method. These parameters brought the best results for all these activation functions. The better performance was also reflected in the precision, recall and f1-score metrics. As we see in Table 32, only for the 32 HE and 128 HE variations, the distributions of accuracy values for the models with the GLN and Tanh functions are similar. The model with the hyperbolic tangent function was the one with the best convergence time, just 55 epochs. However, less stable than other functions in the same variation.

From Figure 61, it is clear that the variations had a great impact on the results. The best batch value as a parameter for the models is the smallest, 32, and the best initialization method, the normal Glorot, for all activation functions.

The α values are shown in Figure 62, and we can see that all distributions were very similar in all layers. The first was the one with a slightly more spaced distribution. Overall, the average value of α was just above 0.6, indicating a slight predominance for the global function.

Table 30 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Mnist - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	32	GL	99.14	0.16	99.34	98.71	**0.03**	0.0	**51.70**	11.17	99.14	99.14	99.14
ReLU			**99.17**	0.10	99.31	98.84	0.03	0.0	56.47	8.99	**99.17**	**99.17**	**99.17**
Sin			98.99	0.07	99.09	98.84	0.03	0.0	59.73	8.40	98.99	98.99	98.99
GLN-ReLU	32	HE	98.79	0.16	99.06	98.39	0.04	0.01	60.8	11.12	98.79	98.79	98.79
ReLU			98.96	0.10	99.17	98.77	0.03	0.00	61.7	7.73	98.96	98.96	98.96
Sin			98.40	0.09	98.59	98.19	0.05	0.00	65.7	7.15	98.4	98.4	98.4
GLN-ReLU	128	GL	98.52	0.14	98.67	97.95	0.05	0.0	62.83	9.03	98.52	98.52	98.52
ReLU			98.61	0.11	98.79	98.16	0.05	0.0	62.50	8.16	98.61	98.61	98.61
Sin			98.04	0.14	98.37	97.75	0.06	0.0	66.43	9.36	98.04	98.04	98.04
GLN-ReLU	128	HE	97.71	0.19	98.13	97.34	0.07	0.01	64.60	9.23	97.71	97.71	97.71
ReLU			97.78	0.19	98.16	97.35	0.07	0.01	65.23	6.06	97.78	97.78	97.78
Sin			96.68	0.13	96.86	96.28	0.10	0.00	70.87	9.06	96.68	96.68	96.68

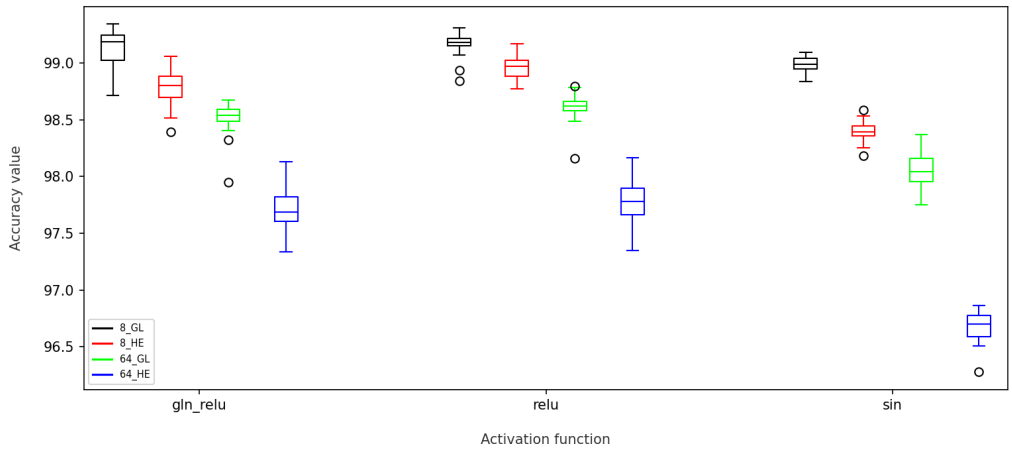


Figure 63 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (MNIST - GLN-ReLU/ReLU/Sine)

In Table 30, where the results for the GLN-ReLU and ReLU and Sine functions are presented, the ReLU function was the one that performed best, obtaining the highest

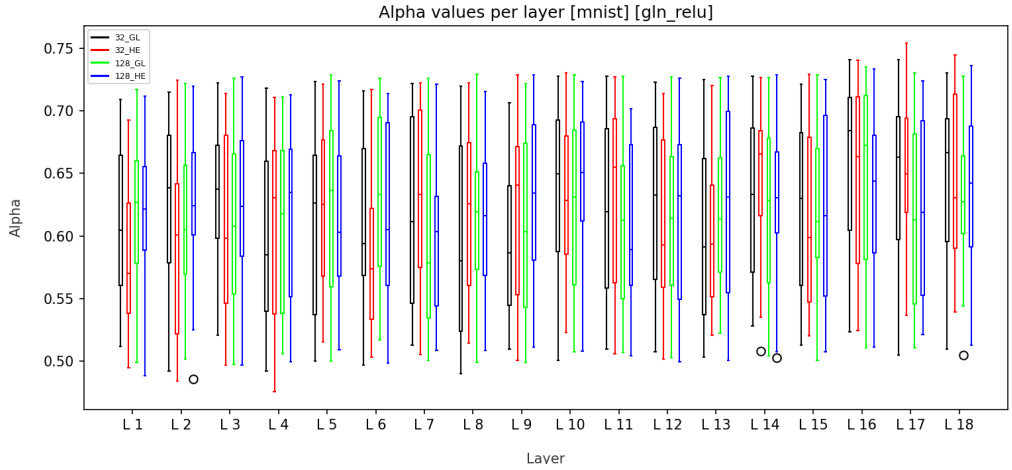


Figure 64 – *Box-Plot* of weight values α per layer (30 experiments) [MNIST - GLN-ReLU]

accuracy value, 99.17, in the batch variation equal to 32 and initialization of Glorot Normal weights. These parameters brought the best results for all these activation functions. The better performance was also reflected in the precision, recall and f1-score metrics. However, the model with the GLN-ReLU function was the one that presented the highest maximum value of 99.34% and also the one that had the shortest convergence time, just 51 epochs. As we see in Table 32, for variations 32 GL, where the model with ReLU obtained the best result, and variation 128 HE, the null hypothesis cannot be discarded, indicating that the distributions of accuracy values for the models with the GLN-ReLU and ReLU functions show similarity.

From Figure 63, it is clear that the variations also had a great impact on the results for the tested activation functions. The best batch value was 32 and the best initialization method was the normal Glorot, for all activation functions. It is also noted here that the normal HE initialization method was not ideal for both the models with the ReLU and GLN-ReLU functions, contrary to expectations.

The values of α are presented in Figure 64, and we can see that all distributions were also very similar in all layers, presenting a graph very similar to the one obtained in the results in Figure 62. Overall, the average value of α was also slightly above 0.6, indicating a slight predominance for the global function.

In Table 31, where the results for the GLN-Mish, Mish and Sine functions are presented, the Mish function was the one that performed best, obtaining the highest accuracy value, 99.23, in the batch variation equal to 32 and initialization of Glorot Normal weights. It was also the one that presented the highest maximum value of 99.42%.

Table 31 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Mnist - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	32	GL	99.07	0.15	99.25	98.74	0.03	0.0	**52.83**	12.18	99.07	99.07	99.07
Mish			**99.23**	0.11	99.42	98.91	**0.03**	0.0	56.27	9.71	**99.23**	**99.23**	**99.23**
Sin			98.99	0.07	99.09	98.84	0.03	0.0	59.73	8.40	98.99	98.99	98.99
GLN-Mish	32	HE	98.89	0.09	99.05	98.62	0.03	0.0	62.97	9.36	98.89	98.89	98.89
Mish			99.01	0.07	99.16	98.89	0.03	0.0	59.63	6.66	99.01	99.01	99.01
Sin			98.40	0.09	98.59	98.19	0.05	0.0	65.70	7.15	98.4	98.4	98.4
GLN-Mish	128	GL	98.58	0.12	98.79	98.28	0.05	0.0	61.50	8.60	98.59	98.58	98.58
Mish			98.69	0.10	98.84	98.37	0.04	0.0	62.80	8.34	98.69	98.69	98.69
Sin			98.04	0.14	98.37	97.75	0.06	0.0	66.43	9.36	98.04	98.04	98.04
GLN-Mish	128	HE	97.70	0.17	98.01	97.36	0.07	0.0	66.40	7.86	97.7	97.7	97.7
Mish			97.98	0.11	98.19	97.74	0.07	0.0	66.17	6.65	97.98	97.98	97.98
Sin			96.68	0.13	96.86	96.28	0.10	0.0	70.87	9.06	96.68	96.68	96.68

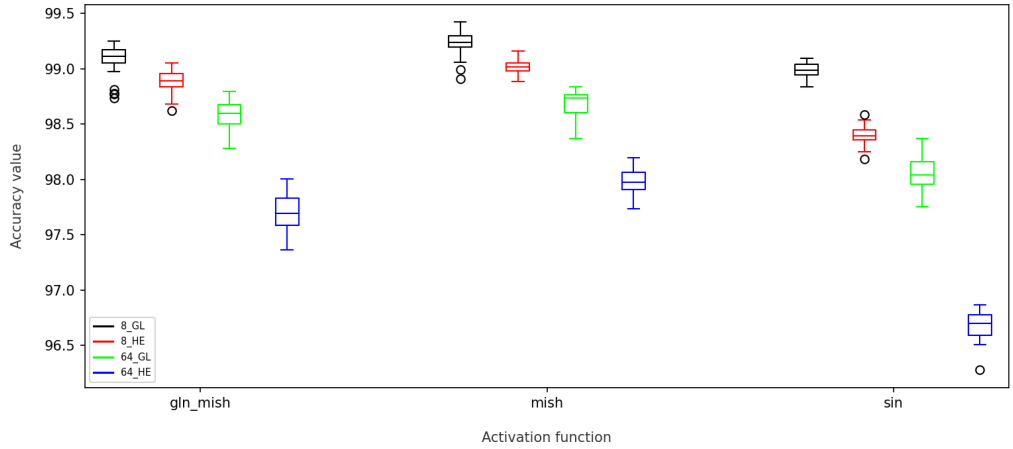


Figure 65 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (MNIST - GLN-Mish/Mish/Sine)

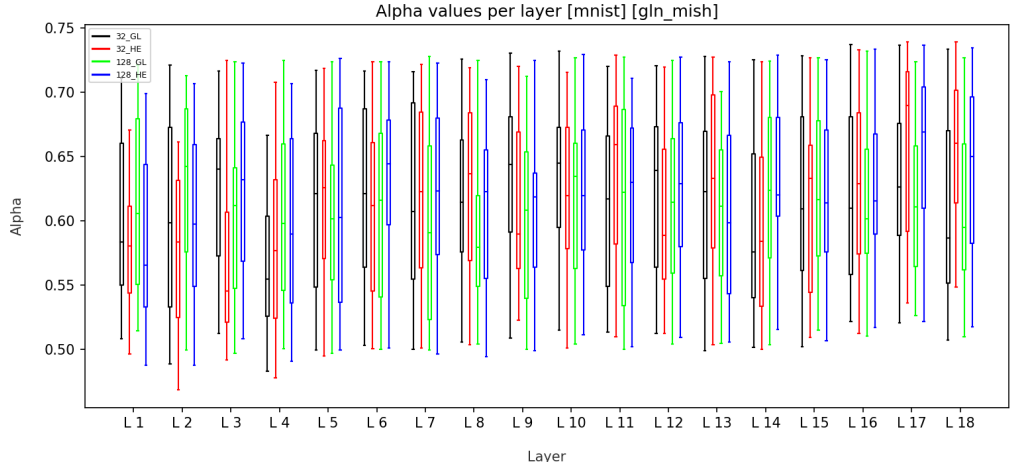


Figure 66 – *Box-Plot* of weight values α per layer (30 experiments) [MNIST - GLN-Mish]

These parameters brought the best results for all these activation functions. The better performance was also reflected in the precision, recall and f1-score metrics.

As we see in Table 32, for variations 32 GL, where the model with Mish obtained the best result, and variation 128 HE, the null hypothesis cannot be discarded, indicating that the distributions of accuracy values for the models with the GLN-Mish and Mish functions are similar. The model with the GLN-Mish function was the one with the best convergence time, just 52 epochs. However, less stable than other functions in the same variation.

From Figure 65, it can be seen that the variations also had a great impact on the results for the tested activation functions. The best batch value was 32 and the best initialization method was the normal Glorot, for all activation functions.

The values of α are presented in Figure 66, here it is also clear that all distributions were also very similar in all layers, presenting a graph very similar to the results in Figure 62 and Figure 64. Overall, the average value of α was also slightly above 0.6, indicating a slight predominance for the global function.

Table 32 – KS test results for *accuracy* - MNIST

batch size →	32								128							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	No	6.55e-03	No	3.46e-02	No	1.01e-15	Yes	0.135	No	5.79e-13	Yes	0.808	No	1.69e-17	No	3.46e-02
sin		-	No	8.74e-05		-	No	1.69e-17		-	No	2.99e-14		-	No	1.69e-17
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	Yes	0.594	No	2.50e-07	No	8.74e-05	No	5.79e-13	No	6.55e-03	No	2.99e-14	Yes	0.135	No	1.69e-17
relu		-	No	8.25e-12		-	No	1.69e-17		-	No	1.01e-15		-	No	1.69e-17
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	No	8.74e-05	No	2.37e-05	No	5.80e-06	No	1.69e-17	No	6.55e-03	No	1.01e-15	No	4.33e-08	No	1.69e-17
mish		-	No	5.79e-13		-	No	1.69e-17		-	No	1.01e-15		-	No	1.69e-17

4.2.2.2 Cifar10

The Tables 33, 34 and 35 present the results, for the Cifar10 dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine function; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function. The average accuracy values for the test data, their maximum and minimum

values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. These results were grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summarized view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 67, 69 and 71. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 68, 70 and 72.

Table 33 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Cifar10 - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	32	GL	**78.44**	0.85	80.18	76.76	**0.65**	0.02	53.87	5.91	**79.26**	**78.44**	**78.69**
Sin			75.05	0.67	76.18	73.46	0.74	0.01	55.60	5.59	76.31	75.05	75.41
Tanh			78.18	0.44	78.97	77.53	0.66	0.02	**51.10**	3.08	79.09	78.18	78.44
GLN	32	HE	73.82	0.82	75.41	72.56	0.76	0.02	59.03	6.12	75.21	73.82	74.19
Sin			70.06	0.42	70.74	69.28	0.86	0.01	61.93	7.59	72.05	70.06	70.59
Tanh			74.01	0.41	75.27	73.32	0.76	0.01	55.60	6.03	75.58	74.01	74.37
GLN	128	GL	69.87	1.00	71.37	66.87	0.86	0.02	58.37	7.08	71.82	69.87	70.39
Sin			65.29	1.01	66.62	60.98	0.98	0.03	58.47	7.10	67.77	65.29	65.91
Tanh			69.62	0.98	70.79	66.12	0.86	0.02	57.17	8.25	72.02	69.62	70.16
GLN	128	HE	63.85	0.80	65.39	62.29	1.02	0.02	65.70	7.81	67.11	63.85	64.61
Sin			58.18	0.71	59.25	56.62	1.17	0.02	64.20	7.80	61.97	58.18	59.05
Tanh			63.69	0.62	64.72	62.51	1.02	0.02	58.27	6.80	67.09	63.69	64.36

As can be seen in Table 33, the GLN function was the one that obtained the highest average accuracy value, 78.44%, among the functions that compose it, tanh and Sine. It was also the one with the highest maximum value of 80.18%. Obtaining this result for batch size 32 and the Glorot Normal initialization method. These parameters brought the best results for all these activation functions. However, the one with the best convergence time, just 51 epochs and with greater stability, was the model with the hyperbolic tangent function.

As we see in Table 36, only in the 32 HE variation, the distributions of accuracy

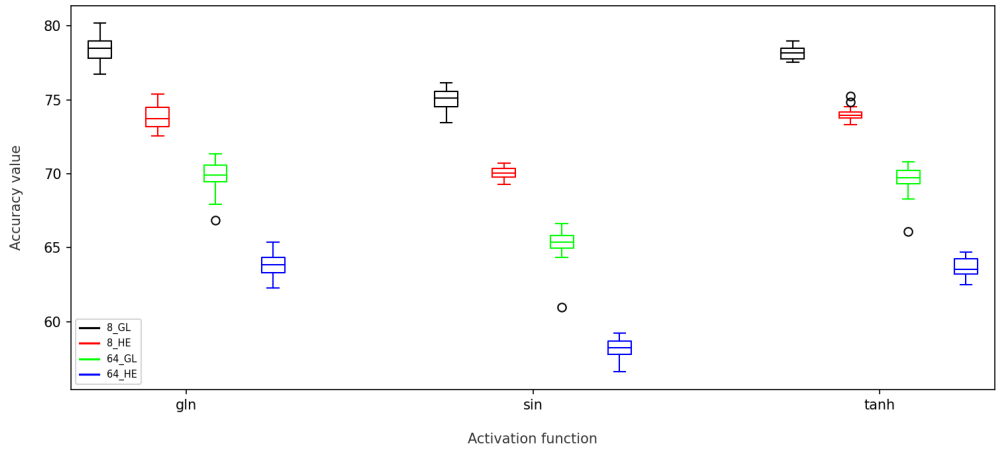


Figure 67 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (CIFAR10 - GLN/Tanh/Sine)

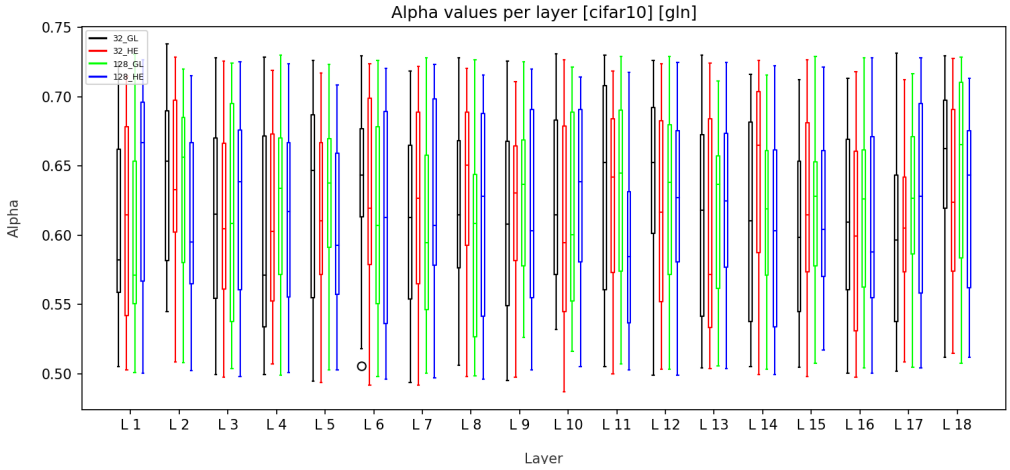


Figure 68 – *Box-Plot* of weight values α per layer (30 experiments) [CIFAR10 - GLN]

values for the models with the GLN and Tanh functions did not show similarity.

From Figure 67, it can be seen that the variations had a great impact on the results. The best batch value as a parameter for the models is the lowest of 32 and the best initialization method is the normal Glorot, for all activation functions.

The values of α are presented in Figure 68, the same behavior of the values of α for all layers of the model was very similar to the behavior of the values of α for the database MNIST data. It ranged from approximately 0.5 to 0.75, indicating a slight predominance also for the global function.

In Table 34, where the results for the GLN-ReLU, ReLU and Sine functions are presented, the ReLU function was the one that performed best, obtaining the highest accuracy value, 81.43, and the highest maximum value, 82.83%, in batch variation equal

Table 34 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Cifar10 - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	32	GL	81.19	0.83	81.98	77.30	0.57	0.02	50.93	5.22	81.75	81.19	81.36
ReLU			**81.43**	1.73	82.83	75.39	**0.56**	0.04	**47.40**	5.65	**81.95**	**81.43**	**81.6**
Sin			75.05	0.67	76.18	73.46	0.74	0.01	55.60	5.59	76.31	75.05	75.41
GLN-ReLU	32	HE	77.80	0.54	78.99	76.72	0.65	0.02	58.80	6.27	78.85	77.8	78.1
ReLU			77.65	0.35	78.27	76.92	0.66	0.01	57.23	6.52	78.61	77.65	77.96
Sin			70.06	0.42	70.74	69.28	0.86	0.01	61.93	7.59	72.05	70.06	70.59
GLN-ReLU	128	GL	74.15	0.64	75.58	72.99	0.74	0.02	60.47	6.55	75.67	74.15	74.56
ReLU			72.86	0.69	74.23	71.01	0.79	0.02	61.00	8.22	74.45	72.86	73.32
Sin			65.29	1.01	66.62	60.98	0.98	0.03	58.47	7.10	67.77	65.29	65.91
GLN-ReLU	128	HE	67.64	0.96	69.33	65.68	0.91	0.02	60.90	6.72	70.0	67.64	68.22
ReLU			65.24	0.41	65.95	64.49	0.99	0.01	67.33	9.10	68.05	65.24	66.03
Sin			58.18	0.71	59.25	56.62	1.17	0.02	64.20	7.80	61.97	58.18	59.05

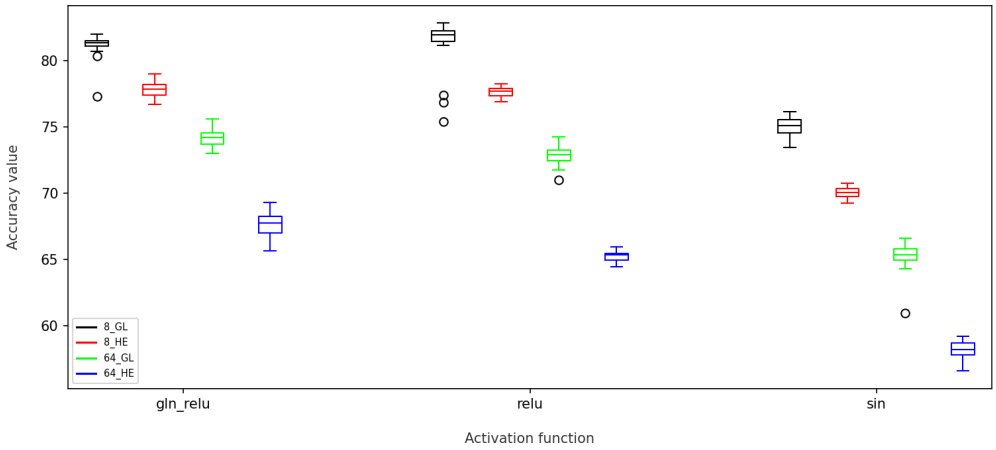


Figure 69 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (CIFAR10 - GLN-ReLU/ReLU/Sine)

to 32 and initialization of Glorot Normal weights. These parameters brought the best results for all these activation functions. The better performance was also reflected in the precision, recall and f1-score metrics. Also, the model with the ReLU function was the one that obtained the best convergence time, just 47 epochs.

As we see in Table 36, only in the 32 HE variation, the distributions of accuracy values for the models with the GLN-ReLU and ReLU functions showed similarity.

From Figure 69, it can be seen that the variations also had a great impact on the results for the tested activation functions. The best batch value was 32 and the best initialization method was the normal Glorot, for all activation functions. It is also noted here that the normal HE initialization method was not ideal for both the models with the

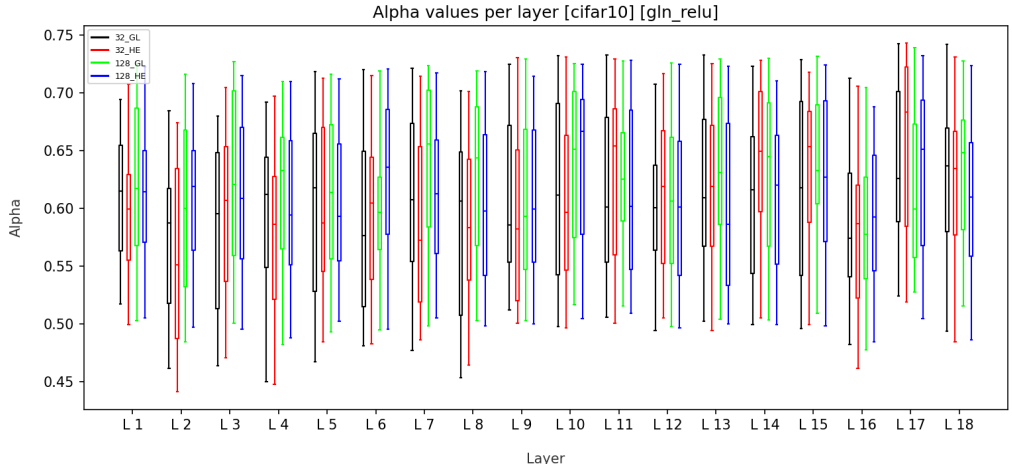


Figure 70 – *Box-Plot* of weight values α per layer (30 experiments) [CIFAR10 - GLN-ReLU]

ReLU and GLN-ReLU functions, again contradicting expectations.

The values of α are shown in Figure 70, the same behavior of the values of α for all layers of the model was very similar to the behavior of the values of α in all functions activation of the GLN type in the MNIST database and the GLN function in the CIFAR10 database. It varied from approximately 0.45 to 0.75, again indicating a slight predominance also for the global function.

Table 35 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Cifar10 - GLN-Mish/Mish/Sineh)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	32	GL	81.25	1.04	82.34	77.16	0.57	0.02	48.90	5.18	81.79	81.25	81.42
Mish			**82.09**	1.42	83.18	78.42	**0.56**	0.03	**44.83**	5.67	**82.44**	**82.09**	**82.21**
Sin			75.05	0.67	76.18	73.46	0.74	0.01	55.60	5.59	76.31	75.05	75.41
GLN-Mish	32	HE	78.00	0.59	79.07	76.81	0.65	0.02	56.30	5.51	79.01	78.0	78.3
Mish			79.08	0.42	79.81	78.00	0.62	0.01	50.73	3.51	79.75	79.08	79.29
Sin			70.06	0.42	70.74	69.28	0.86	0.01	61.93	7.59	72.05	70.06	70.59
GLN-Mish	128	GL	74.41	1.19	75.72	69.27	0.74	0.03	60.20	9.76	75.94	74.41	74.84
Mish			75.27	0.55	76.18	74.03	0.72	0.01	60.07	8.87	76.3	75.27	75.58
Sin			65.29	1.01	66.62	60.98	0.98	0.03	58.47	7.10	67.77	65.29	65.91
GLN-Mish	128	HE	67.92	1.04	69.42	65.81	0.91	0.03	59.80	9.08	70.51	67.92	68.55
Mish			68.32	0.41	69.03	67.22	0.90	0.01	70.97	9.64	70.4	68.32	68.87
Sin			58.18	0.71	59.25	56.62	1.17	0.02	64.20	7.80	61.97	58.18	59.05

The Mish function obtained the highest accuracy value compared to the GLN-Mish and Sine functions, as can be seen in Table 35. It obtained an average of 82.09% and the highest maximum value, 83.18%. The lowest epoch value, 44, was also obtained by the

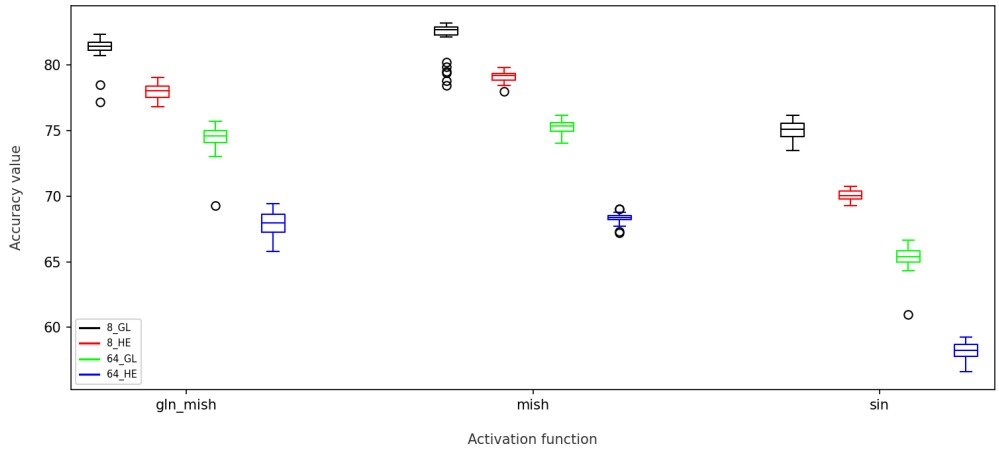


Figure 71 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (CIFAR10 - GLN-Mish/Mish/Sine)

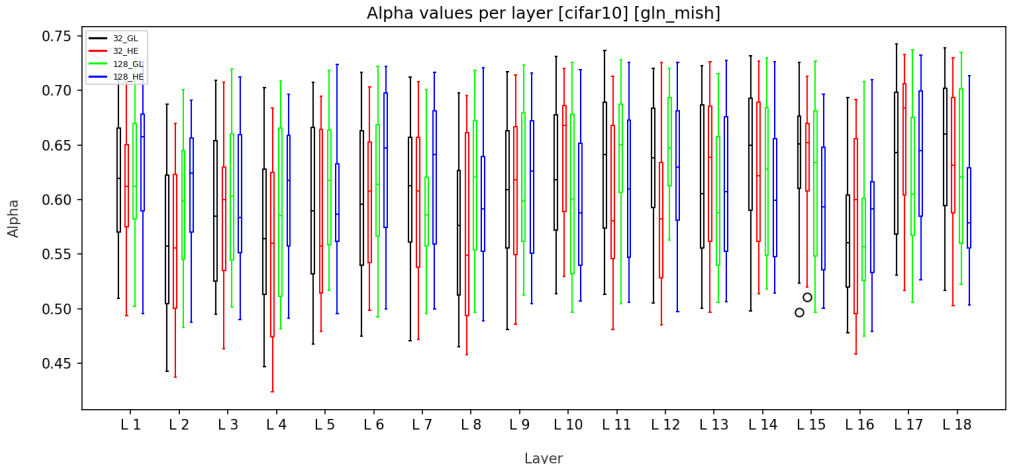


Figure 72 – *Box-Plot* of weight values α per layer (30 experiments) [CIFAR10 - GLN-Mish]

Mish function. These parameters brought the best results for all these activation functions. The better performance was also reflected in the precision, recall and f1-score metrics.

As we see in Table 36, in all variations, the null hypothesis can be discarded, indicating that the distributions of accuracy values for the models are not similar.

From Figure 71, it can be seen that the variations also had a great impact on the results for the tested activation functions. The best batch value was 32 and the best initialization method was the normal Glorot, for all activation functions.

The values of α are shown in Figure 72, the same behavior of the values of α for all layers of the model was very similar to the behavior of the values of α in all functions activation of the GLN type in the MNIST database and the GLN function in the CIFAR10 database. It varied from approximately 0.45 to 0.75, again indicating a slight predominance

for the global function.

Table 36 – KS test results for *accuracy* - CIFAR10

batch size →	32								128							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	No	1.69e-17	Yes	0.071	No	1.69e-17	No	3.46e-02	No	1.69e-17	Yes	0.594	No	1.69e-17	Yes	0.594
sin		-	No	1.69e-17		-	No	1.69e-17		-	No	1.01e-15		-	No	1.69e-17
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	No	2.93e-04	No	1.69e-17	Yes	0.135	No	1.69e-17	No	2.50e-07	No	1.69e-17	No	2.99e-14	No	1.69e-17
relu		-	No	1.01e-15		-	No	1.69e-17		-	No	1.69e-17		-	No	1.69e-17
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	No	4.33e-08	No	1.69e-17	No	6.53e-09	No	1.69e-17	No	9.00e-04	No	1.69e-17	No	1.56e-02	No	1.69e-17
mish		-	No	1.69e-17		-	No	1.69e-17		-	No	1.69e-17		-	No	1.69e-17

4.2.2.3 Lung Colon Cancer

The Tables 37, 38 and 39 present the results, for the Lung Colon Cancer dataset, of the models that trained with the activation functions respectively: GLN and the functions that compose it, the Tanh and Sine function; GLN-ReLU and the functions that compose it, the ReLU and Sine function; GLN-Mish and the functions that compose it, the Mish and Sine function. The average accuracy values for the test data, their maximum and minimum values and the standard deviation are presented. It also presents the loss values for this same data set and its standard deviation, the value of the time, where the lowest loss value was obtained for the validation set, and its standard deviation. Precision, recall and f1-score values are also presented. These results were grouped by the parameter variations performed: batch size, 8 and 64, and the two initialization methods tested, Glorot Normal and He Normal.

To provide a summarized view of the data distribution and complement the information in these tables, we present the accuracy data in bloxpot format in the Figures 73, 75 and 77. Each graph represents the distribution of accuracy results for each activation function tested in the 4 parameter variations performed.

The values of α per layer of the neural network are also presented in bloxpot format in the Figures 74, 76 and 78.

As can be seen in Table 37, The hyperbolic tangent function was the one that obtained the highest average accuracy value, 97.24%, between the GLN and Sine functions.

Table 37 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Lung Colon Cancer - GLN/Tanh/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN	32	GL	96.96	0.70	97.87	94.90	0.09	0.02	53.73	13.13	96.97	96.96	96.96
Sin			96.97	0.37	97.57	96.13	0.08	0.01	52.57	9.04	96.98	96.97	96.97
Tanh			**97.24**	0.68	97.80	94.97	0.08	0.02	54.07	11.36	**97.24**	**97.24**	**97.24**
GLN	32	HE	96.85	1.08	97.90	92.03	0.09	0.03	46.23	12.94	96.86	96.85	96.85
Sin			97.19	0.44	97.87	96.03	**0.08**	0.01	46.43	7.84	97.2	97.19	97.19
Tanh			96.85	0.45	97.43	95.67	0.08	0.01	43.97	9.31	96.87	96.85	96.85
GLN	128	GL	78.53	20.74	97.43	33.30	0.56	0.63	**36.63**	23.53	78.49	78.53	78.51
Sin			75.90	24.29	98.27	28.77	0.61	0.66	40.67	29.84	78.69	75.9	76.42
Tanh			89.59	13.49	98.00	37.67	0.25	0.29	47.20	22.89	89.79	89.59	89.64
GLN	128	HE	88.52	20.27	97.77	32.27	0.34	0.66	52.7	21.44	88.64	88.52	88.53
Sin			95.26	8.01	97.53	52.90	0.12	0.18	56.4	12.21	95.27	95.26	95.25
Tanh			91.52	14.66	97.77	37.93	0.18	0.27	55.8	18.13	91.45	91.52	91.44

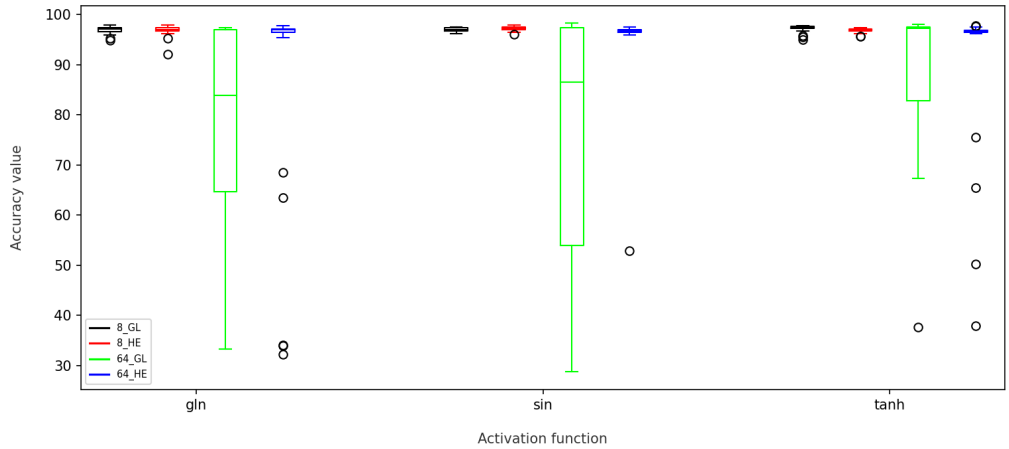


Figure 73 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (Lung Colon Cancer - GLN/Tanh/Sine)

This result was obtained for batch size 32 and the Glorot Normal initialization method. These parameters brought the best results for the Tanh and GLN functions. While the sine function showed better results in the 32 HE configuration. As we see in Table 40, the KS test demonstrates that all distributions of accuracy values for the models with the GLN, Tanh and Sine functions showed similarity.

The model with the hyperbolic tangent function also obtained the best convergence time, just 43 epochs. While the model with the Sine function, in the 128 GL configuration, presented the highest maximum value of 98.27%. However, also in this variation, it was the one that presented the lowest minimum value, 28.77. There is less stability of the results. This instability also occurred with all activation functions in this variation, as we

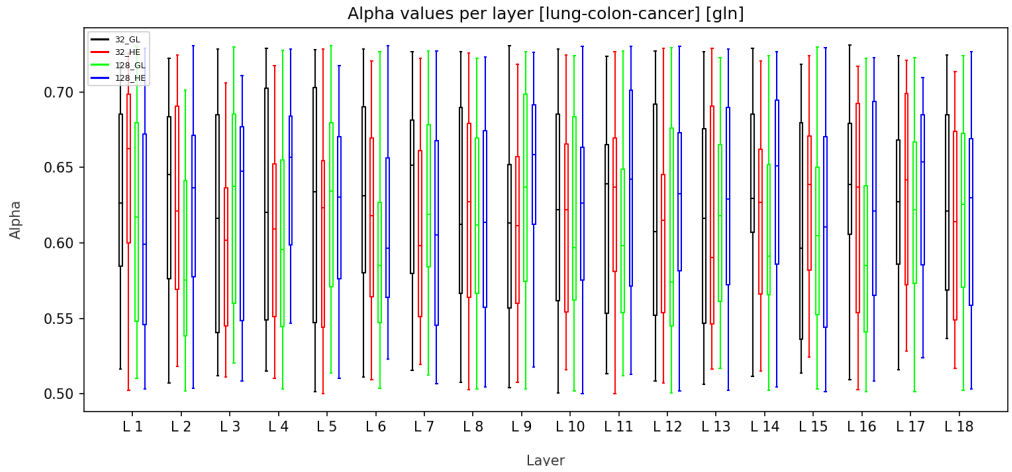


Figure 74 – *Box-Plot* of weight values α per layer (30 experiments) [Lung Colon Cancer - GLN]

can see in Figure 73.

The α values are shown in Figure 74, the same behavior of the α values for all layers of the model was also very similar to those presented in the other databases. The variation in all layers was between approximately 0.5 and 0.75, also indicating a slight predominance for the global function.

Table 38 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Lung Colon Cancer - GLN-ReLU/ReLU/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-ReLU	32	GL	97.62	0.99	98.73	93.07	0.06	0.02	55.70	14.42	97.62	97.62	97.62
ReLU			**98.72**	0.43	99.33	97.57	**0.03**	0.01	60.23	9.14	**98.72**	**98.72**	**98.72**
Sin			96.97	0.37	97.57	96.13	0.08	0.01	52.57	9.04	96.98	96.97	96.97
GLN-ReLU	32	HE	97.14	0.54	97.90	95.30	0.08	0.01	47.40	16.94	97.14	97.14	97.14
ReLU			98.39	0.38	98.90	97.03	0.04	0.01	54.87	8.72	98.39	98.39	98.39
Sin			97.19	0.44	97.87	96.03	0.08	0.01	**46.43**	7.84	97.2	97.19	97.19
GLN-ReLU	128	GL	52.75	22.21	98.20	31.00	1.05	0.55	20.80	16.59	58.36	52.75	54.11
ReLU			33.44	0.42	34.17	32.67	1.11	0.00	11.00	0.00	81.56	33.44	44.87
Sin			75.90	24.29	98.27	28.77	0.61	0.66	40.67	29.84	78.69	75.9	76.42
GLN-ReLU	128	HE	85.04	24.25	97.63	33.4	0.32	0.50	50.93	21.60	85.11	85.04	85.04
ReLU			58.31	23.71	98.30	34.1	0.79	0.42	23.93	21.67	62.86	58.31	57.95
Sin			95.26	8.01	97.53	52.9	0.12	0.18	56.40	12.21	95.27	95.26	95.25

As can be seen in Table 38, the ReLU function was the one that obtained the highest average accuracy value, 98.72%, between the GLN-ReLU and Seno functions, and which presented the highest maximum value of 99.33% . These results were obtained for batch size 32 and the Glorot Normal initialization method. These same parameters were

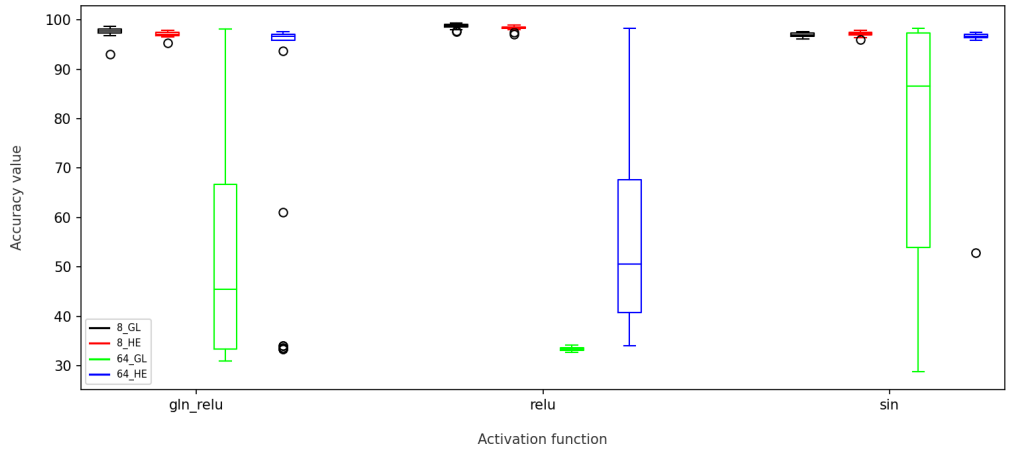


Figure 75 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (Lung Colon Cancer - GLN-ReLU/ReLU/Sine)

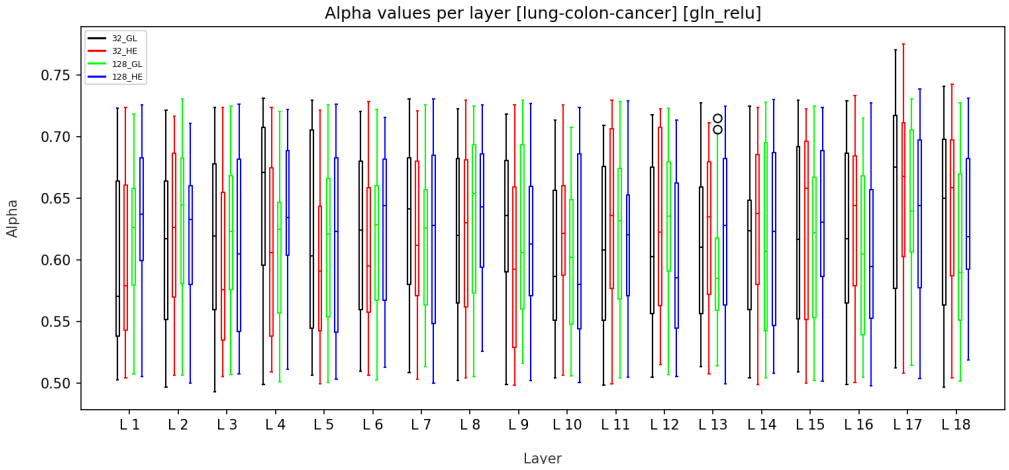


Figure 76 – *Box-Plot* of weight values α per layer (30 experiments) [Lung Colon Cancer - GLN-ReLU]

also ideal for the GLN-ReLU function, while for the sine function it was 32 HE. As we see in Table 40, the KS test demonstrated that GLN-ReLU and ReLU are not statistically similar in any variation. Furthermore, the model with the Sine function was the one that obtained the best convergence time, only 46 epochs and greater stability, among the variations that obtained the best results.

As we can see in Figure 75, the model with the ReLU function was unable to converge when configured with batch 128 and GL initialization method. It also did not perform very well with the HE initialization method in this same batch size, although it managed to converge with some samples. While models with the GLN-ReLU and sine functions had better results. It can be seen that this difficulty in convergence did not change the behavior of the alpha values, as we can see in Figure 76. Continuing to have

results very similar to the other distribution of α values in other databases.

Table 39 – Average, maximum, minimum and standard deviation values of accuracy, average and standard deviation values of loss, average and standard deviation values of the epoch and values of precision, recall and f1-score. (Lung Colon Cancer - GLN-Mish/Mish/Sine)

act	batch size	ini. weight	acc_mean	acc_std	max	min	loss_mean	loss_std	epoch_mean	epoch_std	precision	recall	f1-score
GLN-Mish	32	GL	97.77	0.80	98.63	94.50	0.06	0.02	56.03	17.39	97.78	97.77	97.77
Mish			**98.36**	0.26	98.80	97.77	**0.05**	0.01	42.20	7.55	**98.36**	**98.36**	**98.36**
Sin			96.97	0.37	97.57	96.13	0.08	0.01	52.57	9.04	96.98	96.97	96.97
GLN-Mish	32	HE	97.35	0.49	98.03	96.17	0.07	0.01	47.53	14.58	97.36	97.35	97.35
Mish			97.32	0.28	97.97	96.77	0.07	0.01	**35.17**	4.49	97.33	97.32	97.32
Sin			97.19	0.44	97.87	96.03	0.08	0.01	46.43	7.84	97.2	97.19	97.19
GLN-Mish	128	GL	44.75	17.41	97.07	32.63	1.25	0.63	17.10	9.79	49.61	44.75	45.91
Mish			33.64	0.52	34.60	32.93	1.14	0.02	11.00	0.00	53.55	33.64	40.72
Sin			75.90	24.29	98.27	28.77	0.61	0.66	40.67	29.84	78.69	75.9	76.42
GLN-Mish	128	HE	71.54	31.38	97.53	31.87	0.61	0.69	42.53	27.85	72.11	71.54	71.47
Mish			55.19	24.63	98.40	32.97	0.82	0.42	23.50	22.44	68.31	55.19	55.73
Sin			95.26	8.01	97.53	52.90	0.12	0.18	56.40	12.21	95.27	95.26	95.25

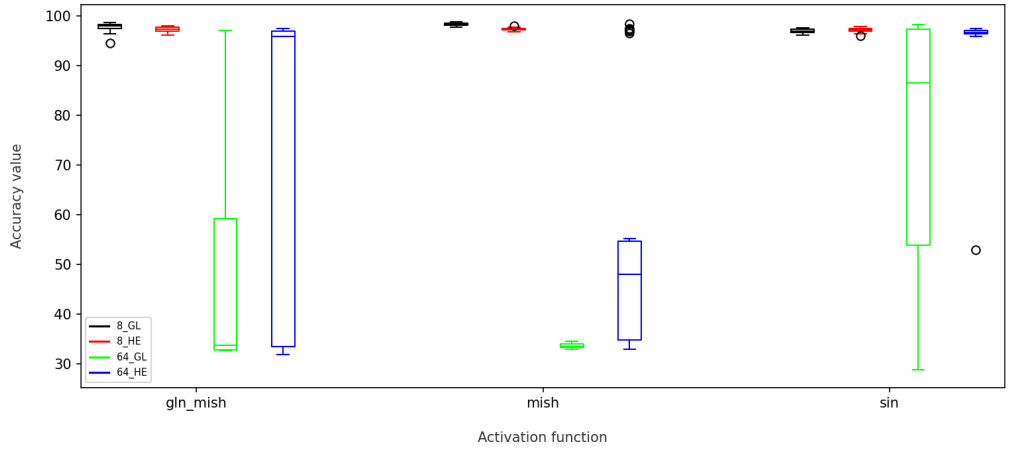


Figure 77 – *Box-Plot* of the accuracy values of the test set by activation function (30 experiments). (Lung Colon Cancer - GLN-Mish/Mish/Sine)

The model with the Mish function obtained the highest accuracy value compared to the GLN-Mish and Sine functions, as can be seen in Table 39, average of 98.36%, the highest maximum value, 98.80%, for batch size 32 and the Glorot Normal initialization method. As shown in Table 40, the KS test demonstrated that GLN-ReLU and ReLU are statistically similar only in the 32 HE variation. The model with the Mish function was also the one that presented the best convergence time, 35 epochs, with the HE Normal weights initialization method.

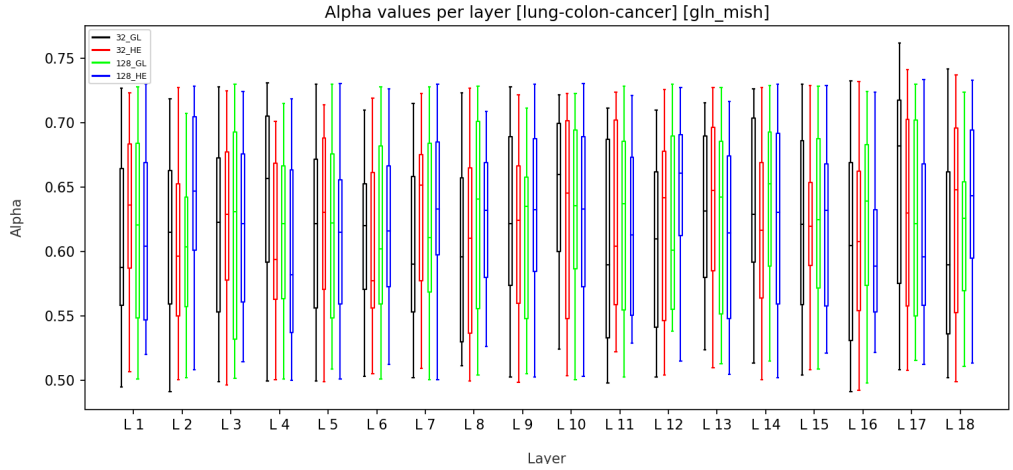


Figure 78 – *Box-Plot* of weight values α per layer (30 experiments) [Lung Colon Cancer - GLN-Mish]

The α values are presented in Figure 74, the same behavior of the α values for all layers of the model was also very similar to those presented in the other databases and in the other variations. The values of α in all layers ranged from approximately 0.5 to 0.75, also indicating a slight predominance for the global function.

Table 40 – KS test results for *accuracy* - Lung Colon Cancer

batch size →	32								128							
ini. weight →	GL				HE				GL				HE			
KS-test →	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values	Similar	p-values
act	sin		tanh		sin		tanh		sin		tanh		sin		tanh	
gln	Yes	0.808	Yes	0.071	Yes	0.239	Yes	0.393	Yes	0.393	No	2.53e-03	Yes	0.393	Yes	0.071
sin		-	No	9.00e-04		-	No	1.56e-02		-	No	1.56e-02		-	Yes	0.999
act	relu		sin		relu		sin		relu		sin		relu		sin	
gln_relu	No	6.53e-09	No	1.28e-06	No	2.99e-14	Yes	0.999	No	8.74e-05	No	6.55e-03	No	8.74e-05	Yes	0.239
relu		-	No	1.01e-15		-	No	2.99e-14		-	No	1.01e-15		-	No	4.33e-08
act	mish		sin		mish		sin		mish		sin		mish		sin	
gln_mish	No	2.53e-03	No	2.50e-07	Yes	0.239	Yes	0.393	No	6.55e-03	No	5.80e-06	No	3.46e-02	No	2.53e-03
mish		-	No	1.69e-17		-	Yes	0.239		-	No	1.01e-15		-	No	4.33e-08

5 Conclusions

This work aimed to evaluate the performance of an artificial neuron called GLN in image compression, multiclass classification and image classification problems, expanding the scope of previously carried out tests. This neuron has an activation function composed of two complementary mathematical functions, one with a global characteristic and the other with a local characteristic. These characteristics are given by the behavior of the chosen functions. The global behavior function must present a variation characteristic throughout its domain, while the local behavior function presents variation in only a fraction of its domain. In this work, the Sine and Growing Cossine Unit functions were chosen as global functions, and the hyperbolic tangent, ReLU, and Mish functions as local functions. These new functions composed different GLN neurons, differing from the original proposal with global function sine and local function hyperbolic tangence (FERREIRA et al., 2021). However, it is worth highlighting that other functions could also be used, as long as they comply with global and local behavioral requirements.

This work was divided into two distinct phases. In the first phase, the GLN neuron was tested in an image compression problem and, in the second phase, in addition to the traditional GLN, composed of the Tanh and Sine functions. Other variations were also tested such as the GLN-ReLU neuron (composition of the Sine and ReLU functions), GLN-Mish (composition of Sine and Mish functions), GLN-GCU (composition of GCU and Tanh functions), and GLN-GCU-R (composition of GCU and ReLU functions) in two new problem categories, multiclass classification and classification of image.

For the first phase, three similar models of an Autoencoder MLP neural network with different activation functions were created. The first with GLN and the others using neurons with conventional activation functions. One with the hyperbolic tangent (Tanh) activation function and the other with the Sine function. Various sizes of bottleneck layers were tested to compress the training images from the CIFAR10 database by 75%, 63%, 50%, 38%, and 25% relative to their original sizes. After training the models, compressed test images were generated by Encoder. These images, with reduced dimensionality, were then trained and tested in a WRN neural network and the SVM algorithm to verify the classification accuracy. All of these models were run 48 times to obtain meaningful samples, given the stochastic nature of neural networks.

As a result of this first phase, the latent representation learned by the autoencoder model with the GLN activation function was the most robust for compressions not exceeding 50%, in a WRN neural network, and the highest precision for compressions above 50%, when using the SVM Classifier. Its use also brings superior performance in training speed, in all compression variations, compared to the isolated activation functions used in the experiment, which make up the GLN neuron itself. The ability of this neuron to adapt to image classification problems has also been proven.

As a behavioral analysis of the GLN model, it was found that the weight value α was consistent with what was expected, given the tendency to generalize the learning of neural networks. For larger compressions, the network tends to learn and represent more general or global information in the data, with less detail, given the restriction on the flow of auto-associative information. With the increase in the number of neurons in the bottleneck, resulting in a decrease in the compression of the information flow, the network starts to represent information with greater weight for details, that is, it starts to increase the importance of the local component, logically implying an increase in auto-associative performance.

A second phase of this work was carried out to expand the GLN experiments to other types of problems not yet studied, multiclass classification and image classification problems, and verify the performance of the GLN neuron using other activation functions. At this stage, ten similar models of an MLP neural network with different activation functions were created for the multiclass classification problem, GLN, GLN-ReLU, GLN-Mish, GLN-GCU, GLN-GCU-R, Tanh, Seno, GCU, ReLU and Mish. For the image classification problem, ten other similar models of a convolutional neural network (CNN) were created with the same activation functions.

The results of the second phase for the first problem addressed showed that the models using the GLN, GLN-ReLU and GLN-Mish functions performed as well as the best results of the models with the isolated functions, Tanh, ReLu and Mish, and in several cases superior to them. They were also the models that obtained the best convergence speeds, where GLN-Mish had the shortest time in Breast Cancer and Yeast and GLN-ReLU in Wine.

In general, comparing all models, for the multiclass classification problem, the GLN-Mish and Mish activation functions were the functions that performed best. Each

had the highest average accuracy value among the other activation functions in two of the four databases evaluated for this type of problem. While the Mish function stood out in the Iris and Wine databases, the GLN-Mish function was the one that performed best for the Breast Cancer and Yeast datasets. These databases were the most challenging among those selected, which leads to the conclusion that the adaptive and trainable functions used, especially GLN, are ideal for more complex problems. For simpler problems, the computational cost ends up being a disadvantage, since isolated, simpler functions have a lower computational cost and solve these problems well.

In contrast, it did not have a superior or statistically equivalent performance in most experiments for solving image classification problems with convolutional networks. A possible explanation is that in very deep networks such as the VGG19 network, the high complexity of the GLN model has a negative impact on its performance. We can also assume that the sparsity characteristic (neurons have zero activation), very present in the ReLU functions, and also in the Mish function, is an important factor, allowing better generalization in these models, with less risk of overfitting and consequently resulting in higher accuracy.

Comparing GLN with other activation functions recently proposed and evaluated for the image classification problem, such as APALU (SUBRAMANIAN et al.,), ErfReLU (RAJANAND; SINGH, 2024), TanhSoft (BISWAS et al., 2021) and LEAF (BODYANSKIY; KOSTIUK, 2023), we can see a difference in the results, despite having been tested on similar convolutional networks, VGG16 and VGG19, and the same databases, MNIST and CIFAR10. While GLN did not obtain good overall results for image classification problems, all of these functions, which are similar to GLN because they are trainable and also because they do not have zero derivatives throughout their domain, obtained superior results in the vast majority of cases presented.

However, significant methodological differences are observed, without a robust statistical analysis, since results were presented based on few training rounds. An average accuracy was captured for only 5 samples (SUBRAMANIAN et al.,), the best results for each function only (BISWAS et al., 2021), or even the result of a single execution (BODYANSKIY; KOSTIUK, 2023). In any case, an interesting aspect was the observation that there was a tendency towards rectification of the LEAF activation function. Both LEAF-as-Tanh and LEAF-as-Sigmoid, LEAF functions initially configured

to behave similarly to Tanh and Sigmoid functions respectively, showed a tendency to "rectify" themselves in convolutional networks, which suggests advantages in using linear functions in CNNs.

Models that use periodic activation functions, especially Growing Cosine Unit (GCU), presented the worst performances in most tests and parameter variations. These evaluated parameters may not have contributed to the convergence of the network in these models, since the inadequate choice of parameters can lead to very slow convergence or the inability to leave local minima, which is already naturally difficult due to the presence of multiple local minima (PARASCANDOLO et al., 2017). As the original article on the GCU function used a custom VGG 19 architecture and provided few details about the parameters used, we conclude that the parameters used in this work may not have brought the best potential of this activation function.

This poor performance of oscillatory functions, despite having influenced the performance of the GLN, made the potential for using this trainable neuron even more evident, since it adapted as expected, seeking the best combinations to reach at least the closest or until achieving a performance superior to that obtained by independent functions. Thus, it helps reduce the need to manually choose activation functions for different layers and architectures of neural networks, given their ability to automatically and optimally learn the best composition between functions according to different input data during training.

Given the observed results and the limitations of this experiment, we point out some promising paths that can be explored in future work to increase our understanding of trainable functions with two complementary characteristics such as GLN.

We observed from the final results that the hyperparameters used in the models may not have been optimal for periodic functions. This led to one of the main limitations identified after the experiments. Therefore, a potential future study could conduct a more thorough and refined investigation of these parameters, including testing new ones, such as improved weight initialization methods specifically tailored for these oscillatory functions. The use of hyperparameter optimization techniques, such as grid search or genetic algorithms, may help identify better combinations across a broader range of options more efficiently.

Another important path is to carry out experiments on more modern and

competitive network architectures, such as Transformers, EfficientNets, and ResNet network architectures, among others. Evaluating the performance of GLN and its derivations on these other architectures can reveal new opportunities and challenges.

The application of GLN-type neurons to each neuron in the neural network is another promising line of investigation. In current research, alpha weights are calculated per layer. In other words, these weights are the same for all neurons in the same layer. However, allowing each neuron to define its own value, choosing between global and local characteristics, or some composition of the two, can increase the representation capacity of the network, as has already been demonstrated in other works (DUSHKOFF; PTUCHA, 2016; ERTUğRUL, 2018).

Bibliography

AEBERHARD, S.; FORINA, M. **Wine**. 1991. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5PC7J.

APICELLA, A.; DONNARUMMA, F.; ISGRò, F.; PREVETE, R. A survey on modern trainable activation functions. **Neural Networks**, Elsevier BV, v. 138, p. 14–32, jun. 2021. ISSN 0893-6080. Disponível em: <http://dx.doi.org/10.1016/j.neunet.2021.01.026>.

APICELLA, A.; ISGRò, F.; PREVETE, R. A simple and efficient architecture for trainable activation functions. **Neurocomputing**, Elsevier B.V., v. 370, p. 1–15, 2019. ISSN 18728286.

AWAD, M.; KHANNA, R. Support vector machines for classification. In: _____. **Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers**. Berkeley, CA: Apress, 2015. p. 39–66. ISBN 978-1-4302-5990-9. Disponível em: <https://doi.org/10.1007/978-1-4302-5990-9_3>.

BENGIO, Y.; BOULANGER-LEWANDOWSKI, N.; PASCANU, R. Advances in optimizing recurrent networks. In: IEEE. **2013 IEEE international conference on acoustics, speech and signal processing**. [S.l.], 2013. p. 8624–8628.

BISHOP, C. M. **Neural networks and their applications**. 1994. Disponível em: <https://doi.org/10.1063/1.1144830>.

BISWAS, K.; KUMAR, S.; BANERJEE, S.; PANDEY, A. K. Tanhsoft—dynamic trainable activation functions for faster learning and better performance. **IEEE Access**, v. 9, p. 120613–120623, 2021.

BODYANSKIY, Y.; KOSTIUK, S. Learnable extended activation function for deep neural networks. **International Journal of Computing**, p. 311–318, 10 2023.

BUDUMA, N.; LOCASCIO, N. **Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms**. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2017. ISBN 1491925612.

CHAN, K.; IM, S. K.; KE, W.; LEI, N.-L. Sinp[n]: A fast convergence activation function for convolutional neural networks. **2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)**, p. 365–369, 2018. Disponível em: <https://api.semanticscholar.org/CorpusID:57763203>.

CLEVERT, D.-A.; UNTERTHINER, T.; HOCHREITER, S. Fast and accurate deep network learning by exponential linear units (elus). **arXiv preprint arXiv:1511.07289**, 2015.

DELGADO, J. H. D. M.; FERREIRA, T. A. E. Autoencoder performance analysis with adaptive and trainable activation function to compress images. In: **2022 IEEE Latin American Conference on Computational Intelligence (LA-CCI)**. [S.l.: s.n.], 2022. p. 1–6.

DU, K.-L.; SWAMY, M. N. **Neural Networks and Statistical Learning**. [S.l.]: Springer Publishing Company, Incorporated, 2013. ISBN 144715570X.

DUSHKOFF, M.; PTUCHA, R. Adaptive activation functions for deep networks. **Electronic Imaging**, v. 2016, p. 1–5, 02 2016.

ERTUğRUL Ömer F. A novel type of activation function in artificial neural networks: Trained activation function. **Neural Networks**, v. 99, p. 148–157, 2018. ISSN 0893-6080. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0893608018300078>.

FERREIRA, T. A. E.; MATTHEAKIS, M.; PROTOPAPAS, P. **A New Artificial Neuron Proposal with Trainable Simultaneous Local and Global Activation Function**. 2021.

FISHER, R. A. **Iris**. 1988. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C56C76.

GIBBONS, J.; CHAKRABORTI, S. **Nonparametric Statistical Inference, Fourth Edition: Revised and Expanded**. Taylor & Francis, 2014. ISBN 9780203911563. Disponível em: <https://books.google.com.br/books?id=kJbVO2G6VicC>.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: TEH, Y. W.; TITTERINGTON, M. (Ed.). **Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics**. Chia Laguna Resort, Sardinia, Italy: PMLR, 2010. (Proceedings of Machine Learning Research, v. 9), p. 249–256. Disponível em: <https://proceedings.mlr.press/v9/glorot10a.html>.

GOODFELLOW, I. J.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge, MA, USA: MIT Press, 2016. <http://www.deeplearningbook.org>.

GUSTINELI, M. **A survey on recently proposed activation functions for Deep Learning**. 2022. Disponível em: <https://arxiv.org/abs/2204.02921>.

HAYKIN, S. **Redes Neurais - 2ed.** Bookman, 2000. ISBN 9788573077186. Disponível em: <https://books.google.com.br/books?id=lBp0X5qfyjUC>.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: **Proceedings of the IEEE international conference on computer vision**. [S.l.: s.n.], 2015. p. 1026–1034.

_____. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: **ICCV**. IEEE Computer Society, 2015. p. 1026–1034. ISBN 978-1-4673-8391-2. Disponível em: <http://dblp.uni-trier.de/db/conf/iccv/iccv2015.html#HeZRS15>.

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. **science**, American Association for the Advancement of Science, v. 313, n. 5786, p. 504–507, 2006.

HORTON, P.; NAKAI, K. A probabilistic classification system for predicting the cellular localization sites of proteins. **Proceedings. International Conference on Intelligent Systems for Molecular Biology**, v. 4, p. 109–15, 1996. Disponível em: <https://api.semanticscholar.org/CorpusID:964254>.

JAGTAP, A. D.; KARNIADAKIS, G. E. **How important are activation functions in regression and classification? A survey, performance comparison, and future directions**. 2022. Disponível em: <https://arxiv.org/abs/2209.02681>.

KILIçARSLAN, S.; CELIK, M. Rsigelu: A nonlinear activation function for deep neural networks. **Expert Systems with Applications**, Elsevier Ltd, v. 174, p. 114805, 2021. ISSN 09574174. Disponível em: <https://doi.org/10.1016/j.eswa.2021.114805>.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. arXiv, 2014. Disponível em: <https://arxiv.org/abs/1412.6980>.

KRIZHEVSKY, A.; HINTON, G. Learning multiple layers of features from tiny images. **Master's thesis, Department of Computer Science, University of Toronto**, Citeseer, 2009.

KWAK, S. G.; KIM, J. H. Central limit theorem: the cornerstone of modern statistics. **Korean journal of anesthesiology**, Korean Society of Anesthesiologists, v. 70, n. 2, p. 144, 2017.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, p. 436–44, 05 2015.

LIAO, Z. Trainable activation function in image classification. 2020. Disponível em: <http://arxiv.org/abs/2004.13271>.

MALLICK, C.; BHOI, S. K.; PANDA, S. K.; JENA, K. K. An efficient learning algorithm for periodic perceptron to test xor function and parity problem. **SN Applied Sciences**, v. 2, 2020. Disponível em: <https://api.semanticscholar.org/CorpusID:210081437>.

MASTROMICHALAKIS, S. Sigmorelu : An improvement activation function by combining sigmoid and relu. p. 1–10, 2021.

MCCAUGHAN, D. On the properties of periodic perceptrons. In: **Proceedings of International Conference on Neural Networks (ICNN'97)**. [S.l.: s.n.], 1997. v. 1, p. 188–193 vol.1.

MCCULLOCH, W.; PITTS, W. A logical calculus of the ideas immanent in nervous activity (1943). **Ideas That Created the Future**, 2021.

MEHTA, I.; GHARBI, M.; BARNES, C.; SHECHTMAN, E.; RAMAMOORTHI, R.; CHANDRAKER, M. Modulated periodic activations for generalizable local functional representations. In: **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.: s.n.], 2021. p. 14214–14223.

MISRA, D. **Mish: A Self Regularized Non-Monotonic Activation Function**. 2020. Disponível em: <https://arxiv.org/abs/1908.08681>.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: **ICML 2010**. [S.l.: s.n.], 2010. p. 807–814.

NAKAI, K. **Yeast**. 1996. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5KG68.

NAKAI, K.; KANEHISA, M. Expert system for predicting protein localization sites in gram-negative bacteria. **Proteins: Structure, Function, and Bioinformatics**, v. 11, n. 2, p. 95–110, 1991. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.340110203>.

NARKHEDE, M. V.; BARTAKKE, P. P.; SUTAONE, M. S. A review on weight initialization strategies for neural networks. **Artificial Intelligence Review**, v. 55, p. 291 – 322, 2021. Disponível em: <https://api.semanticscholar.org/CorpusID:237793845>.

NG, A. Cs294 a lecture notes- sparse autoencoder. **URL: https://web.stanford.edu/class/cs294a/sparseAutoencoder_2011new.pdf**, Stanford Univ. Press, 2010.

NOEL, M. M.; TRIVEDI, A.; DUTTA, P. et al. Growing cosine unit: A novel oscillatory activation function that can speedup training and reduce parameters in convolutional neural networks. **arXiv preprint arXiv:2108.12943**, 2021.

NWANKPA, C.; IJOMAH, W.; GACHAGAN, A.; MARSHALL, S. **Activation Functions: Comparison of trends in Practice and Research for Deep Learning**. 2018. Disponível em: <https://arxiv.org/abs/1811.03378>.

PARASCANDOLO, G.; HUTTUNEN, H.; VIRTANEN, T. **Taming the waves: sine as activation function in deep neural networks**. 2017. Disponível em: <https://openreview.net/forum?id=Sks3zF9eg>.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. et al. Scikit-learn: Machine learning in python. **Journal of machine learning research**, v. 12, n. Oct, p. 2825–2830, 2011.

RAJANAND, A.; SINGH, P. Erfrelu: adaptive activation function for deep neural network. **Pattern Anal. Appl.**, Springer-Verlag, Berlin, Heidelberg, v. 27, n. 2, maio 2024. ISSN 1433-7541. Disponível em: <https://doi.org/10.1007/s10044-024-01277-w>.

RAMACHANDRAN, P.; ZOPH, B.; LE, Q. V. Swish: a self-gated activation function. **arXiv: Neural and Evolutionary Computing**, 2017. Disponível em: <https://api.semanticscholar.org/CorpusID:196158220>.

RAMASINGHE, S.; LUCEY, S. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2022. p. 142–158.

ROBIN, D. A. R.; SCAMAN, K.; LELARGE, M. Periodic signal recovery with regularized sine neural networks. In: **NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations**. [s.n.], 2022. Disponível em: <https://openreview.net/forum?id=7W4lWxxzgDA>.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958. ISSN 0033-295X. Disponível em: <http://dx.doi.org/10.1037/h0042519>.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **nature**, Nature Publishing Group UK London, v. 323, n. 6088, p. 533–536, 1986.

SITZMANN, V.; MARTEL, J.; BERGMAN, A.; LINDELL, D.; WETZSTEIN, G. Implicit neural representations with periodic activation functions. **Advances in neural information processing systems**, v. 33, p. 7462–7473, 2020.

SOPENA, J.; ROMERO, E.; ALQUEZAR, R. Neural networks with periodic and monotonic activation functions: a comparative study in classification problems. In: **1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)**. [S.l.: s.n.], 1999. v. 1, p. 323–328 vol.1.

SUBRAMANIAN, B.; JEYARAJ, R.; AKHRORJON, R.; UGLI, A.; KIM, J. **APALU: A Trainable, Adaptive Activation Function for Deep Learning Networks**.

WOLBERG WILLIAM, M. O. S. N.; STREET, W. **Breast Cancer Wisconsin (Diagnostic)**. 1995. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5DW2B.

WU, L. Learning a single neuron for non-monotonic activation functions. In: PMLR. **International Conference on Artificial Intelligence and Statistics**. [S.l.], 2022. p. 4178–4197.

XU, B.; HUANG, R.; LI, M. Revise saturated activation functions. **arXiv preprint arXiv:1602.05980**, 2016.

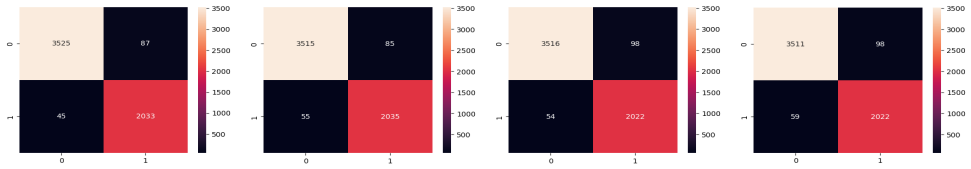
XU, B.; WANG, N.; CHEN, T.; LI, M. Empirical evaluation of rectified activations in convolutional network. **arXiv preprint arXiv:1505.00853**, 2015.

ZAGORUYKO, S.; KOMODAKIS, N. Wide residual networks. **arXiv preprint arXiv:1605.07146**, 2016.

ZIYIN, L.; HARTWIG, T.; UEDA, M. Neural networks fail to learn periodic functions and how to fix it. **Advances in Neural Information Processing Systems**, v. 33, p. 1583–1594, 2020.

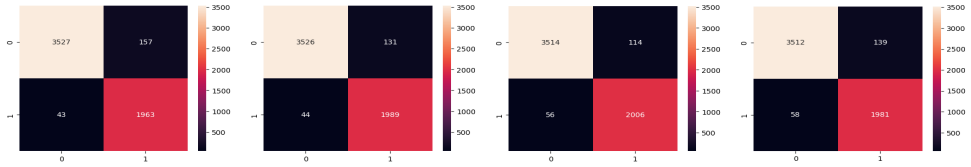
APPENDIX

.1 Breast Cancer



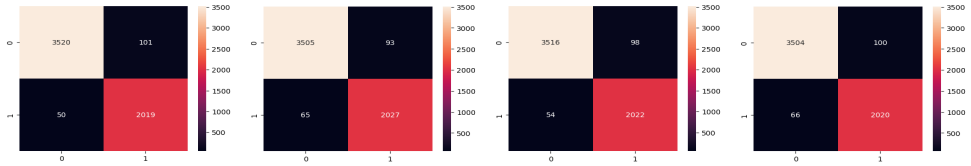
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 79 – Heatmap Breast Cancer (Tanh)



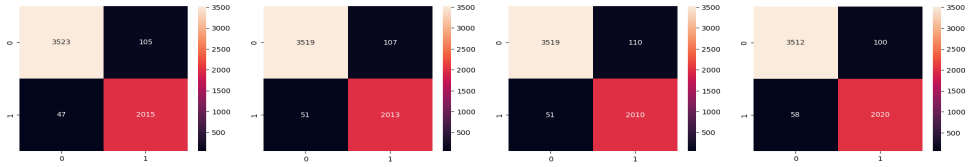
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 80 – Heatmap Breast Cancer (Sin)



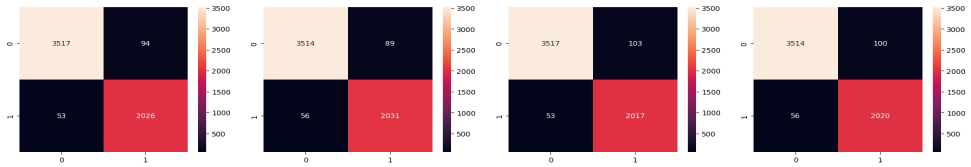
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 81 – Heatmap Breast Cancer (GLN)



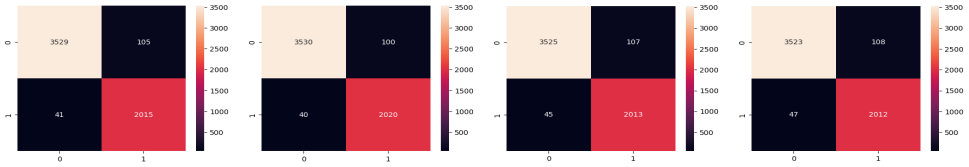
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 82 – Heatmap Breast Cancer (ReLU)



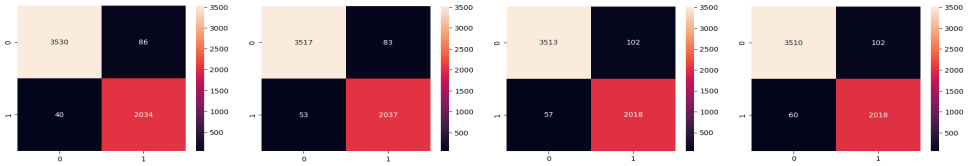
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 83 – Heatmap Breast Cancer (GLN-ReLU)



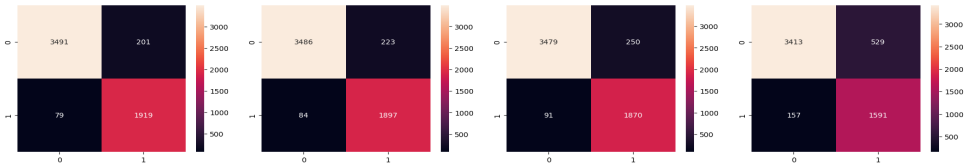
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 84 – Heatmap Breast Cancer (Mish)



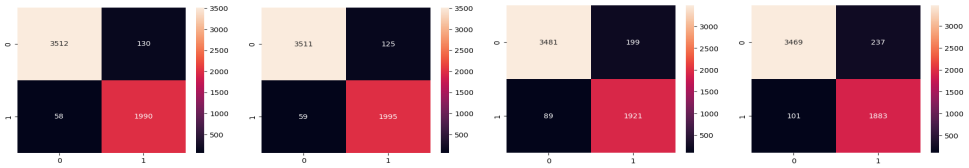
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 85 – Heatmap Breast Cancer (GLN-Mish)



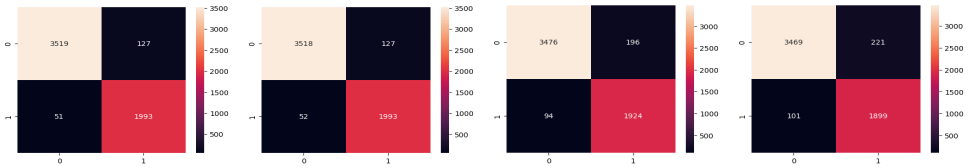
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 86 – Heatmap Breast Cancer (GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

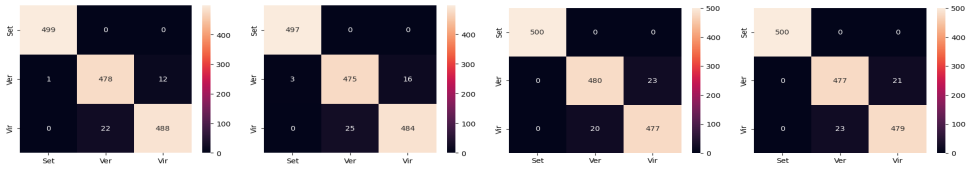
Figure 87 – Heatmap Breast Cancer (GLN-GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

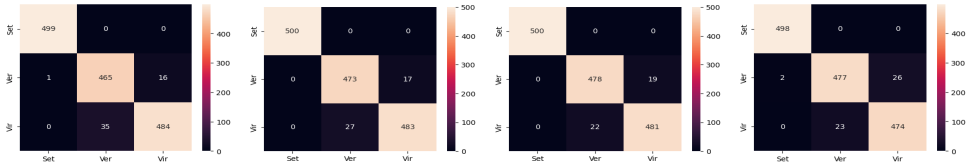
Figure 88 – Heatmap Breast Cancer (GLN-GCU-ReLU)

.2 Iris



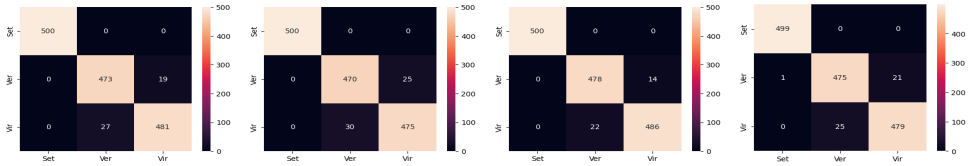
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 89 – Heatmap Iris (Tanh)



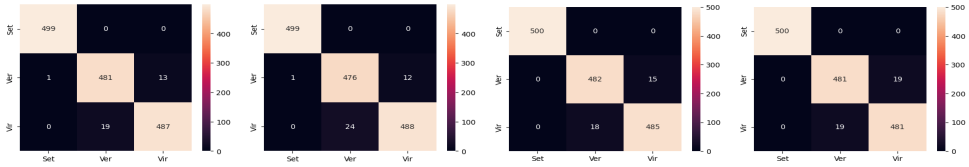
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 90 – Heatmap Iris (Sin)



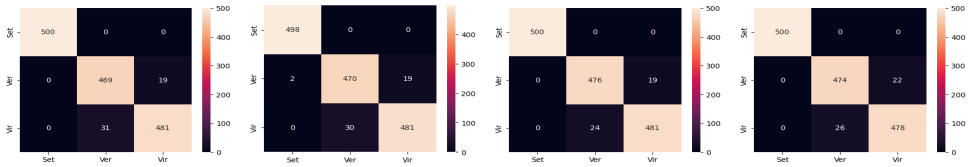
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 91 – Heatmap Iris (GLN)



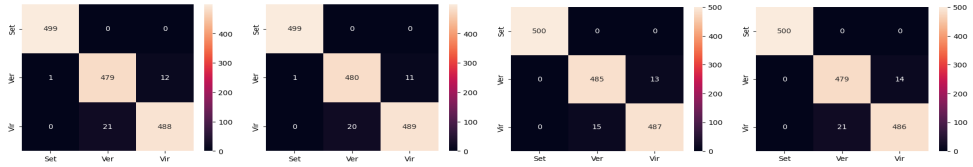
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 92 – Heatmap Iris (ReLU)



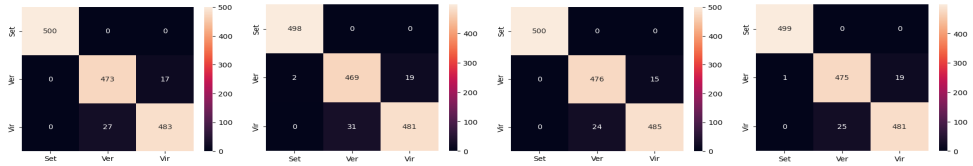
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 93 – Heatmap Iris (GLN-ReLU)



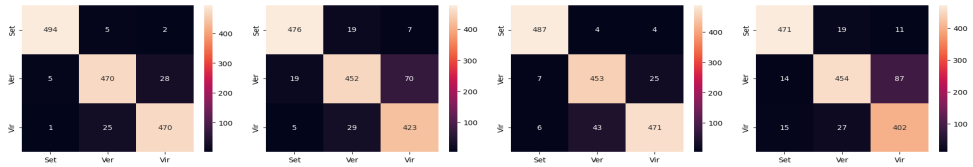
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 94 – Heatmap Iris (Mish)



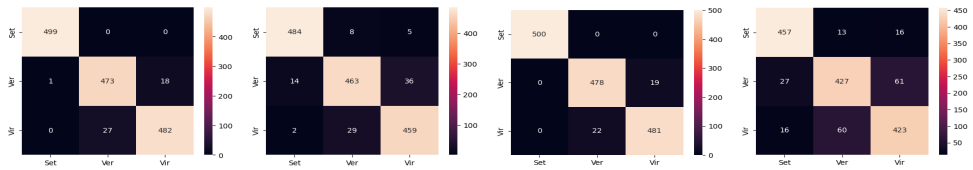
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 95 – Heatmap Iris (GLN-Mish)



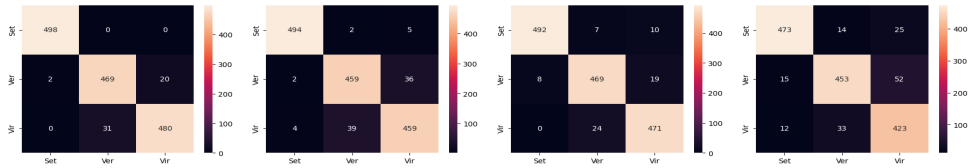
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 96 – Heatmap Iris (GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

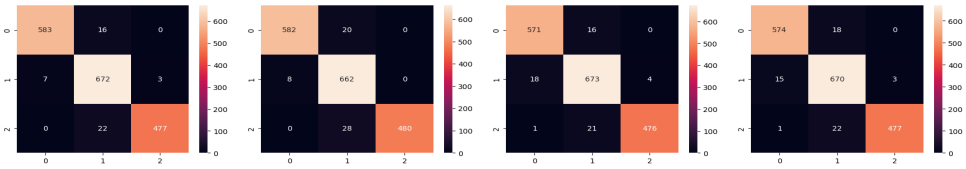
Figure 97 – Heatmap Iris (GLN-GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

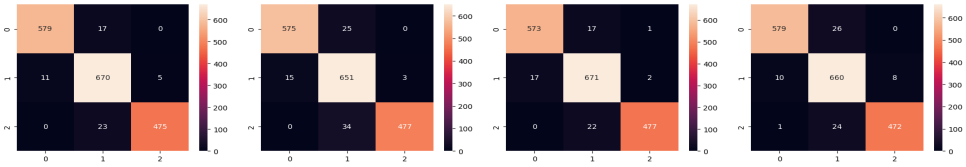
Figure 98 – Heatmap Iris (GLN-GCU-ReLU)

.3 Wine



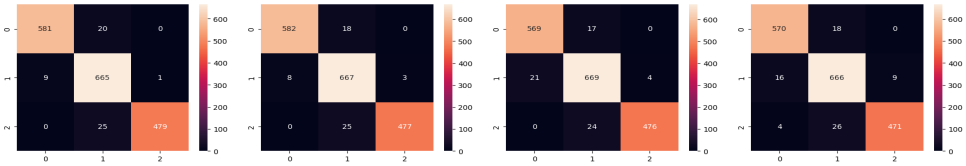
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 99 – Heatmap Wine (Tanh)



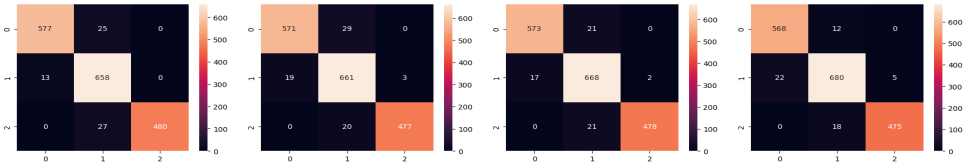
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 100 – Heatmap Wine (Sin)



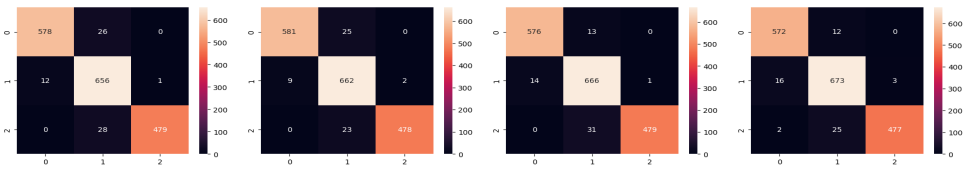
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 101 – Heatmap Wine (GLN)



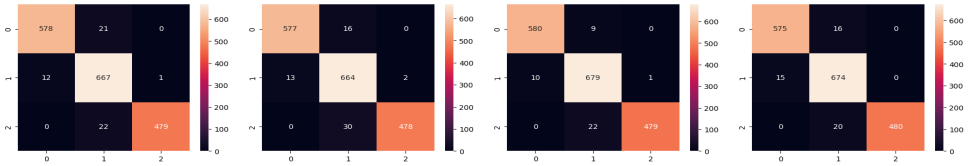
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 102 – Heatmap Wine (ReLU)



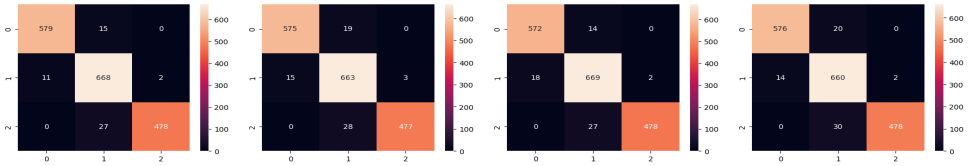
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 103 – Heatmap Wine (GLN-ReLU)



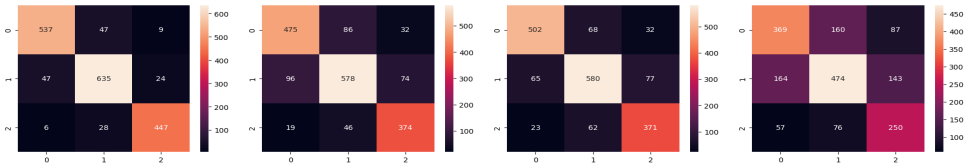
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 104 – Heatmap Wine (Mish)



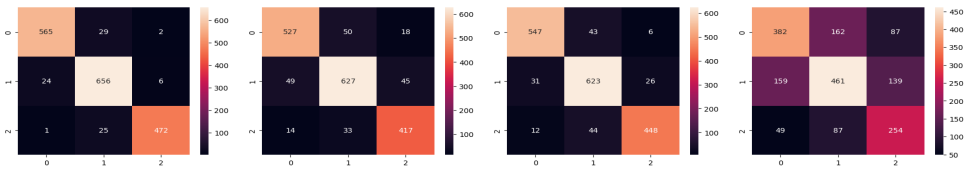
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 105 – Heatmap Wine (GLN-Mish)



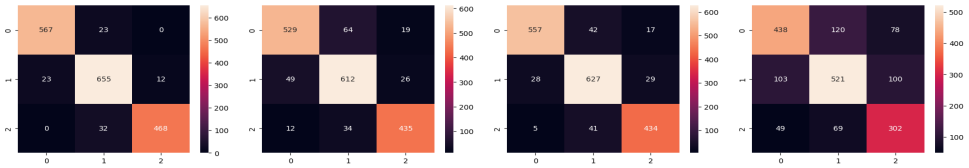
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 106 – Heatmap Wine (GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

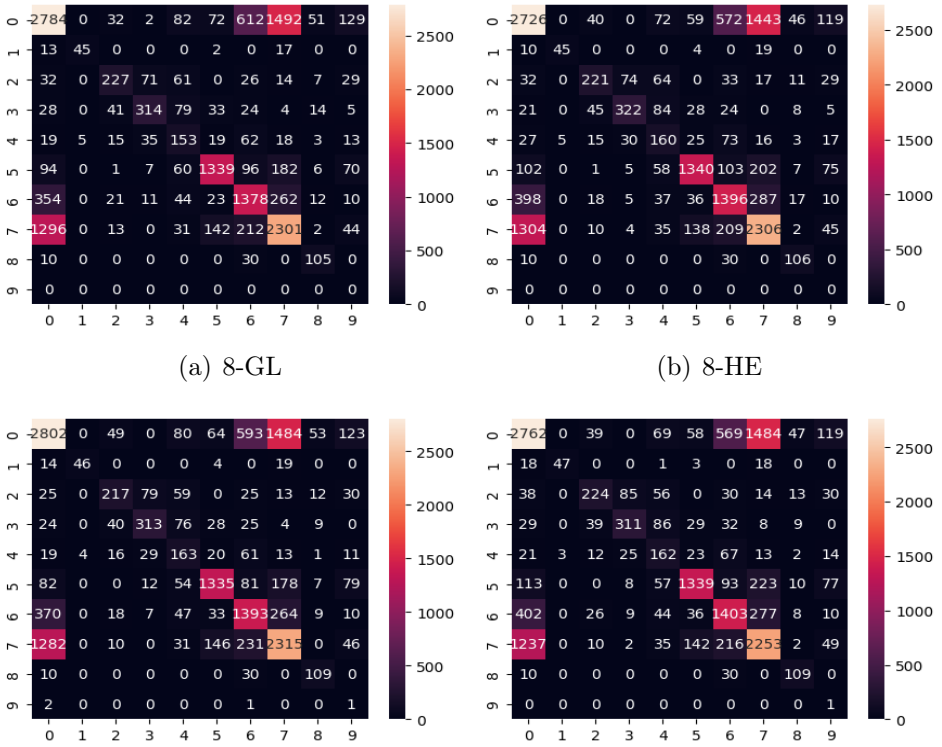
Figure 107 – Heatmap Wine (GLN-GCU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 108 – Heatmap Wine (GLN-GCU-ReLU)

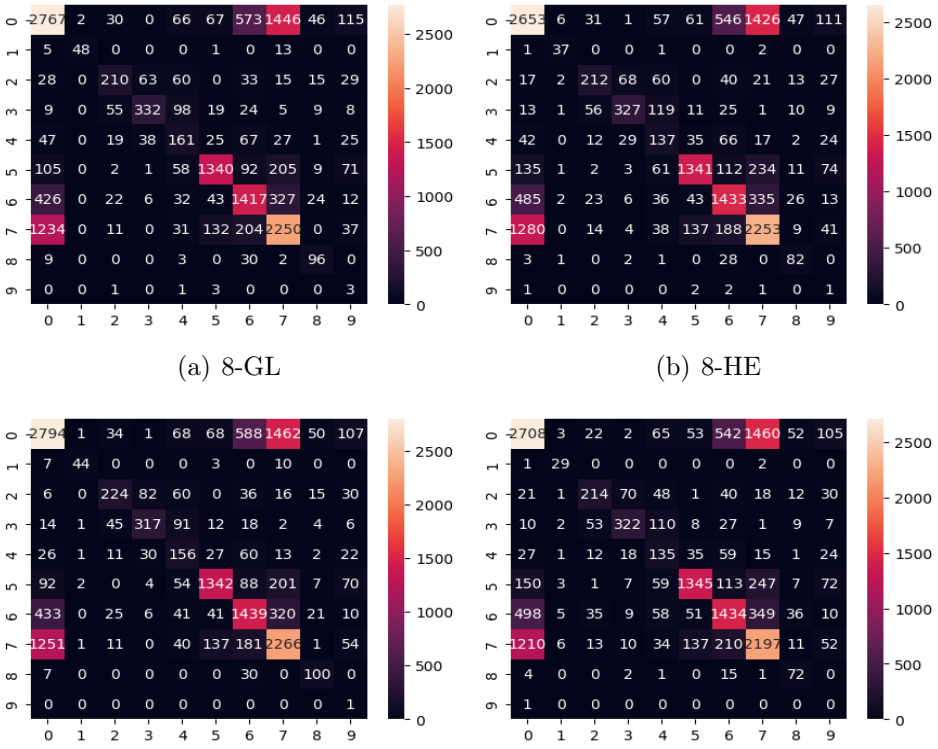
.4 Yeast



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

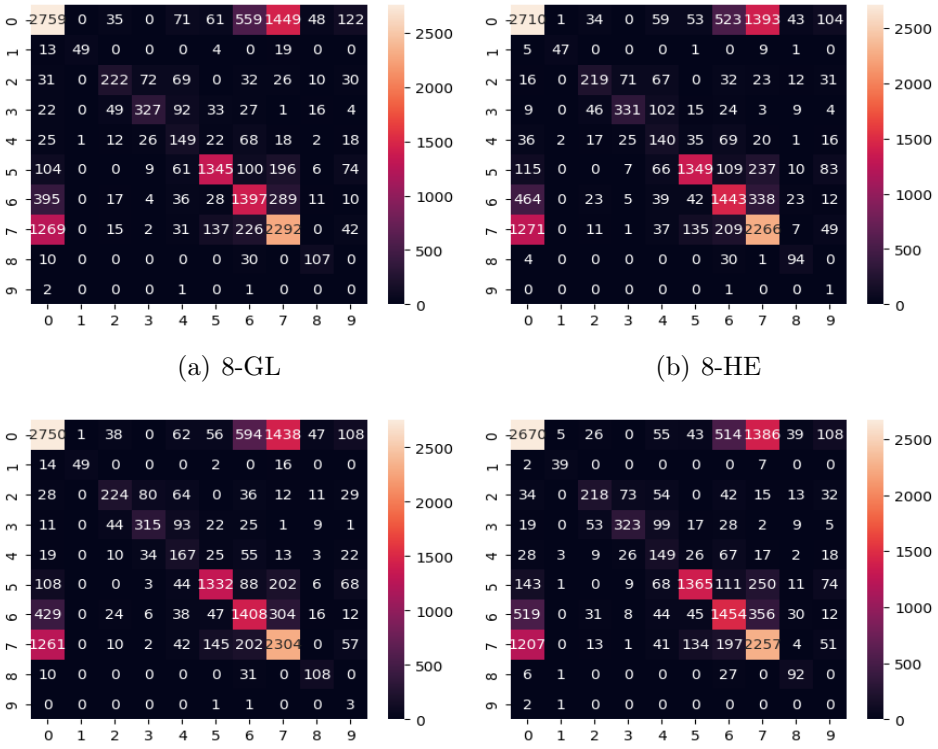
Figure 109 – Heatmap Yeast (Tanh)



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

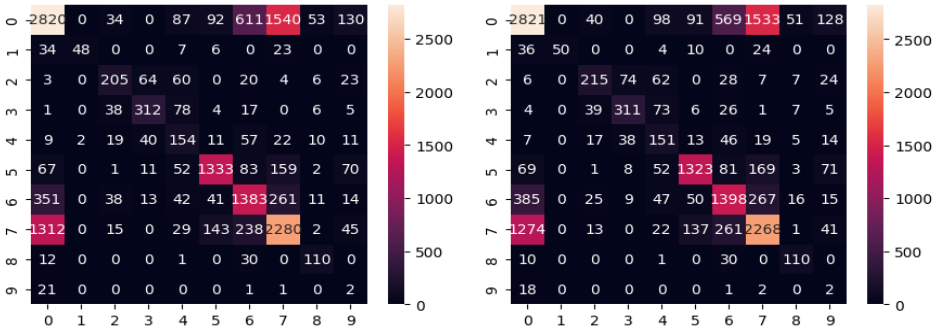
Figure 110 – Heatmap Yeast (Sin)



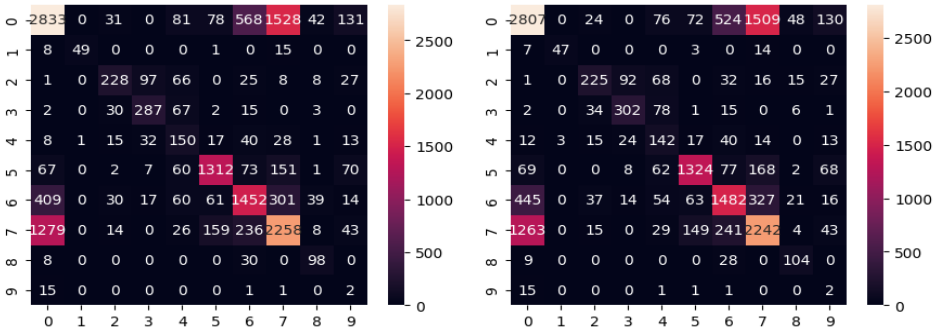
(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 111 – Heatmap Yeast (GLN)

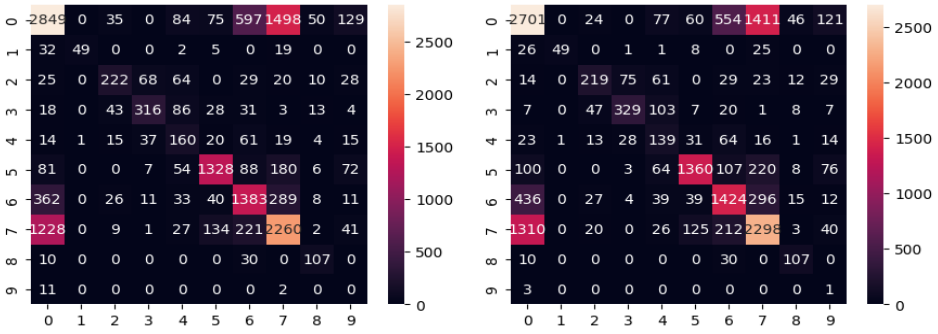


(a) 8-GL (b) 8-HE

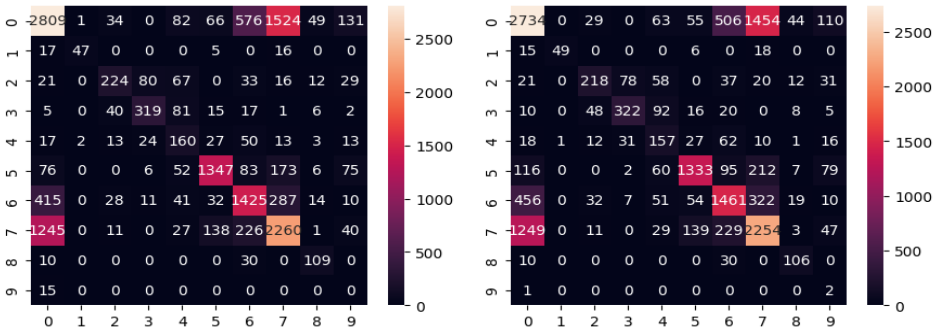


(c) 64-GL (d) 64-HE

Figure 112 – Heatmap Yeast (ReLU)

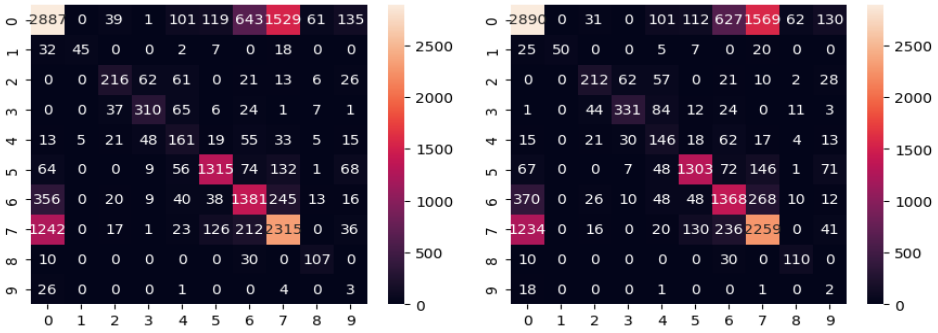


(a) 8-GL (b) 8-HE

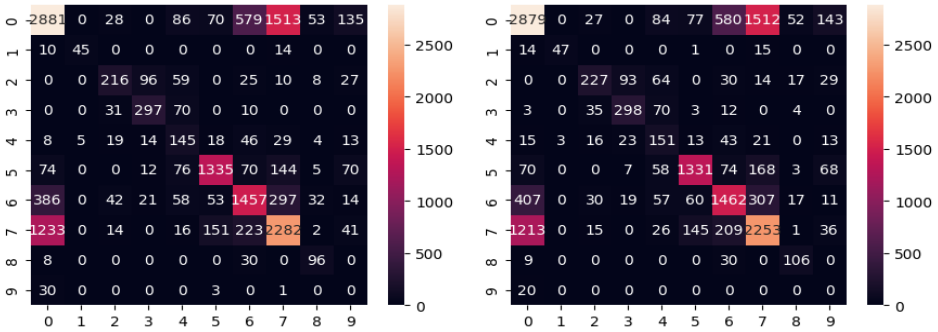


(c) 64-GL (d) 64-HE

Figure 113 – Heatmap Yeast (GLN-ReLU)

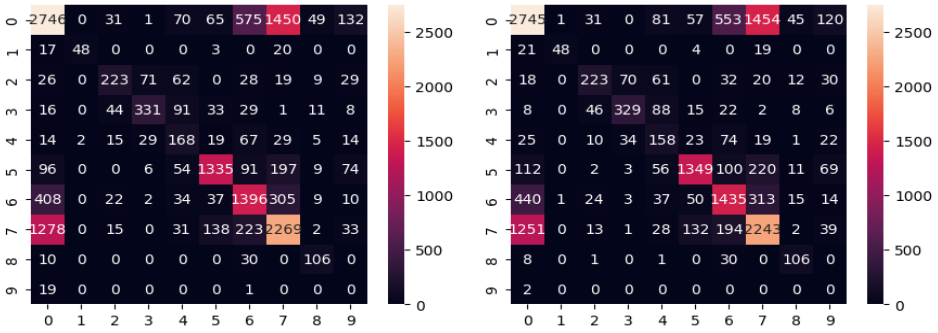


(a) 8-GL (b) 8-HE

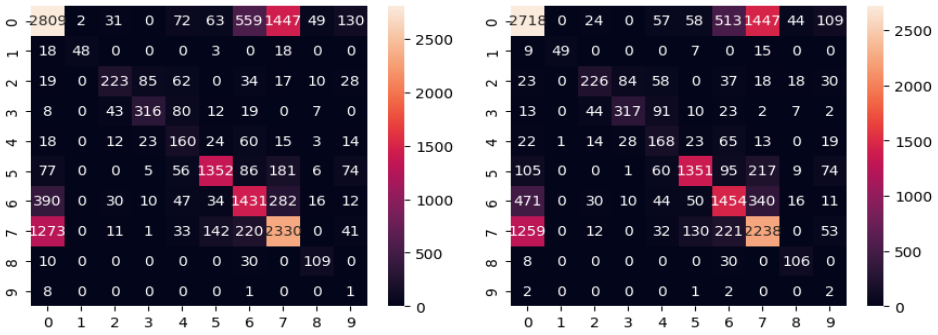


(c) 64-GL (d) 64-HE

Figure 114 – Heatmap Yeast (Mish)

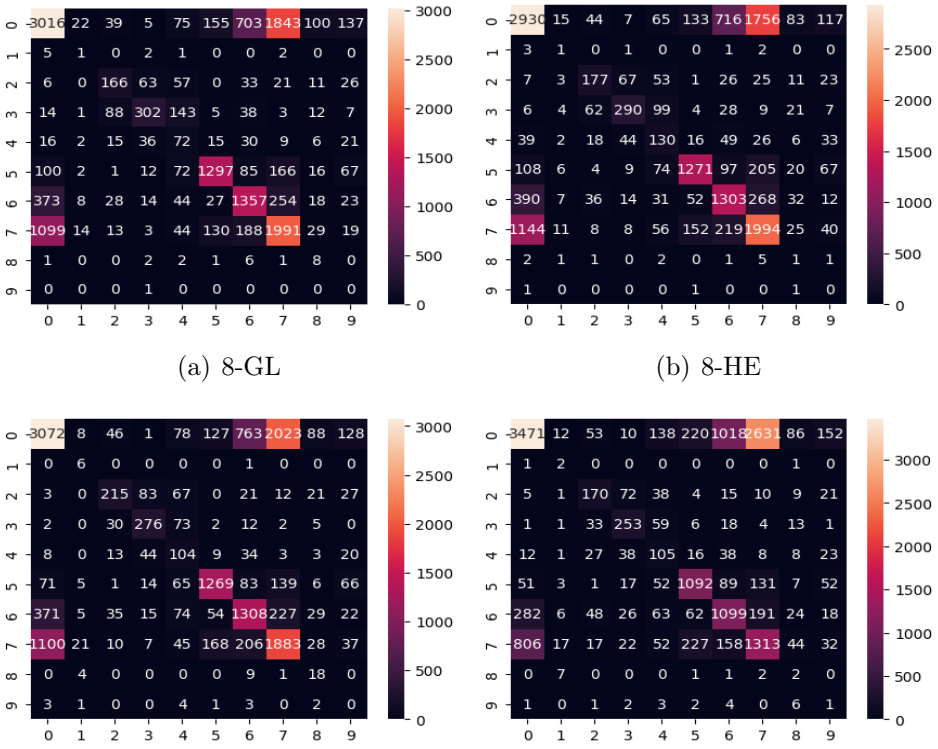


(a) 8-GL (b) 8-HE



(c) 64-GL (d) 64-HE

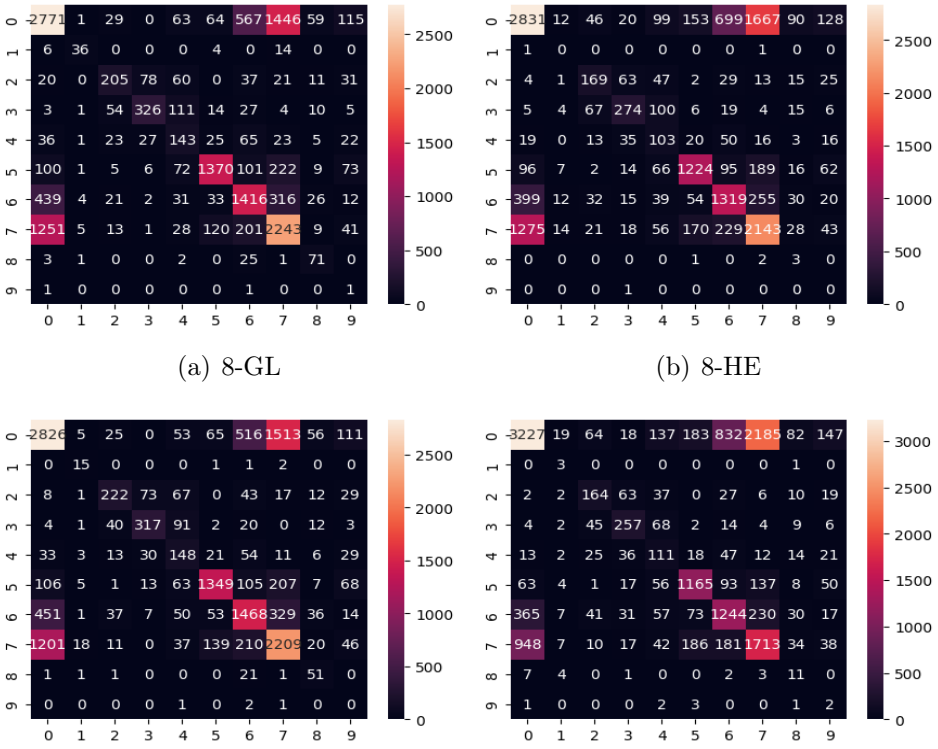
Figure 115 – Heatmap Yeast (GLN-Mish)



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

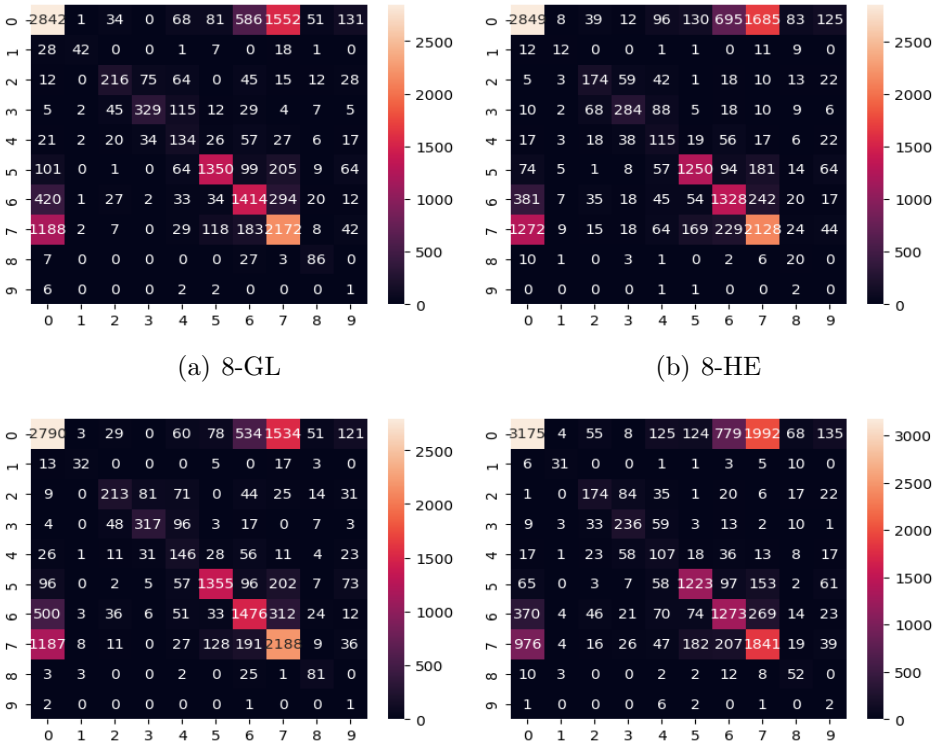
Figure 116 – Heatmap Yeast (GCU)



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 117 – Heatmap Yeast (GLN-GCU)

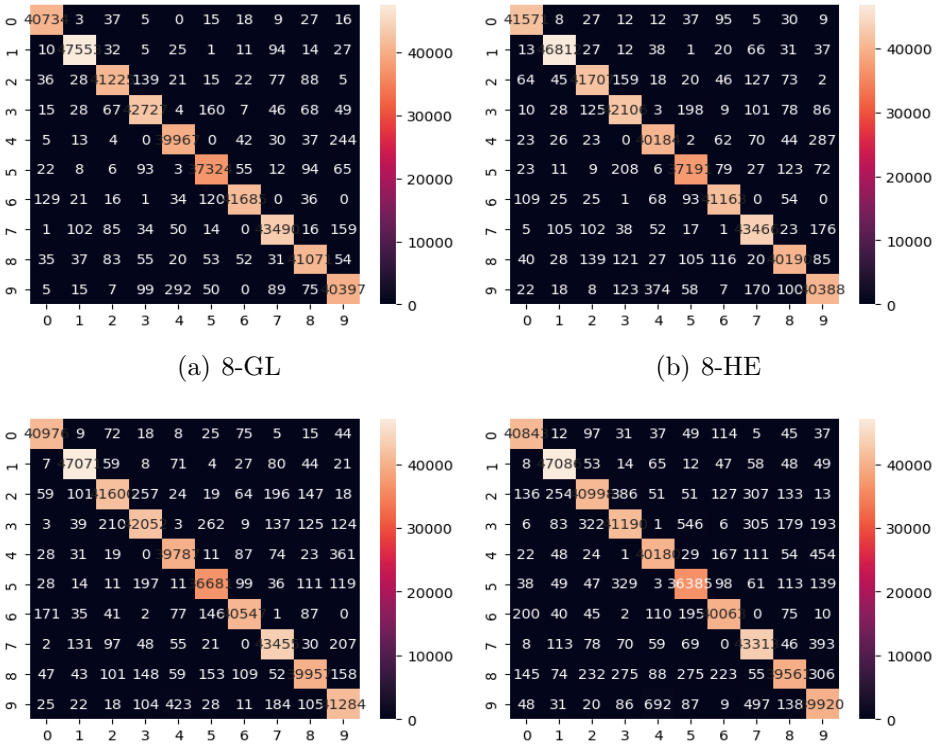


(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 118 – Heatmap Yeast (GLN-GCU-ReLU)

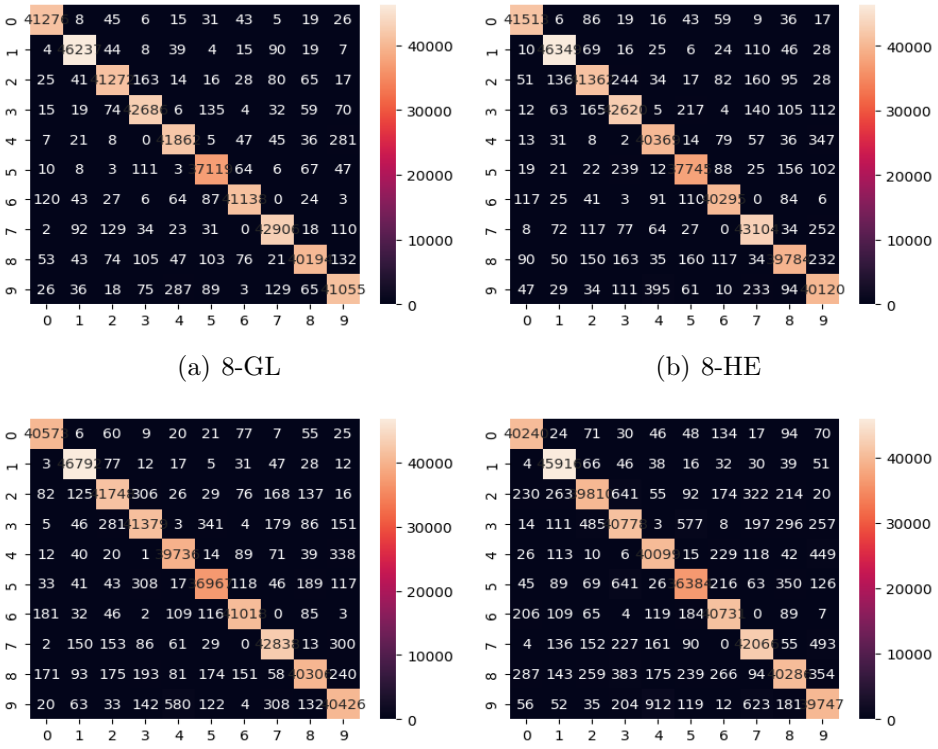
.5 Mnist



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

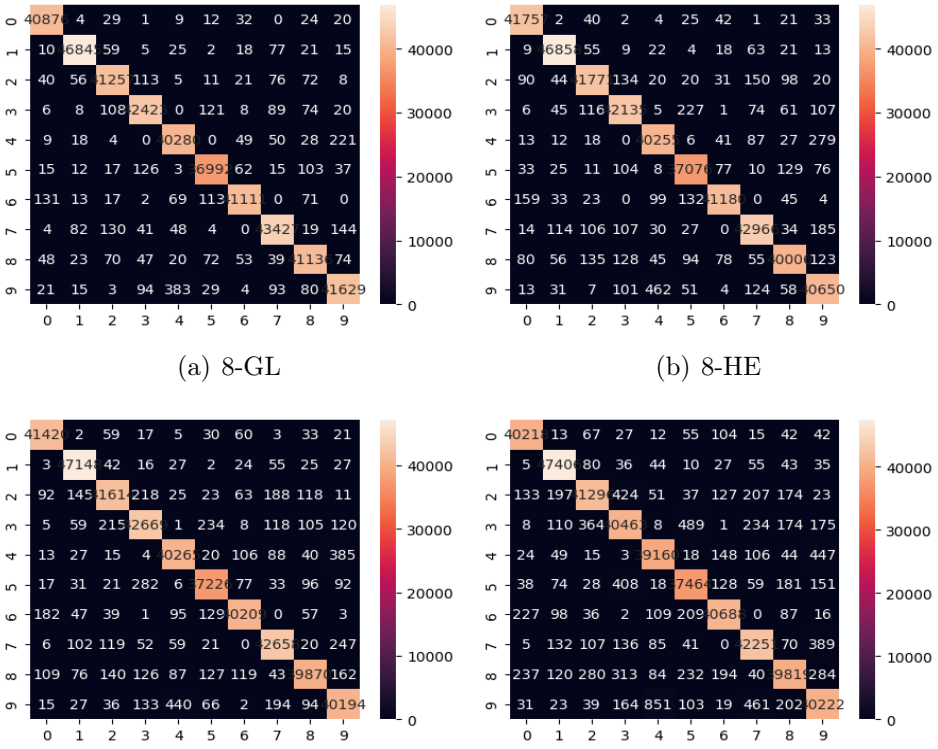
Figure 119 – Heatmap Mnist (Tanh)



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

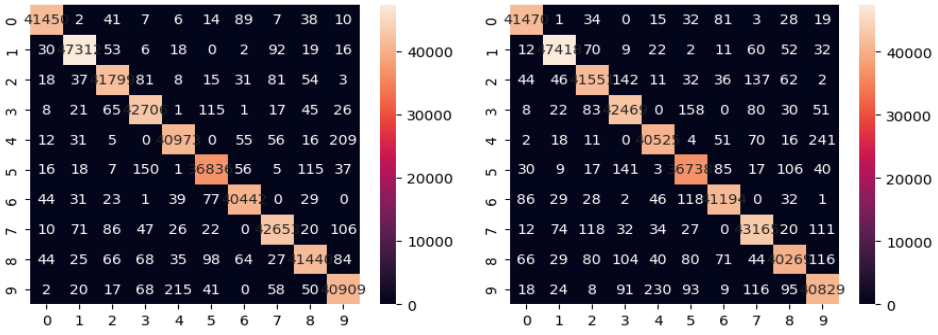
Figure 120 – Heatmap Mnist (Sin)



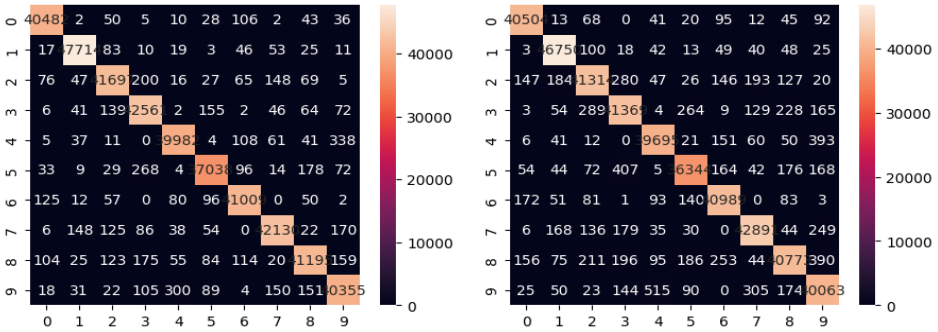
(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 121 – Heatmap Mnist (GLN)

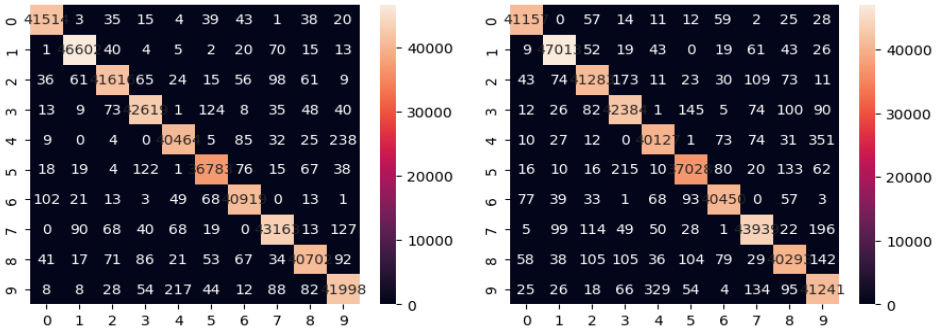


(a) 8-GL (b) 8-HE

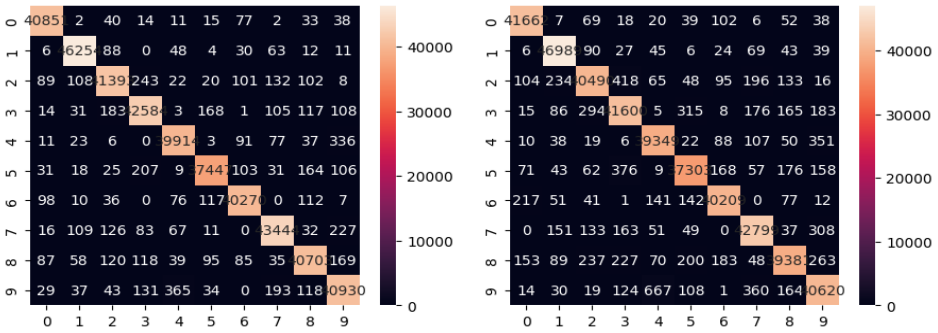


(c) 64-GL (d) 64-HE

Figure 122 – Heatmap Mnist (ReLU)

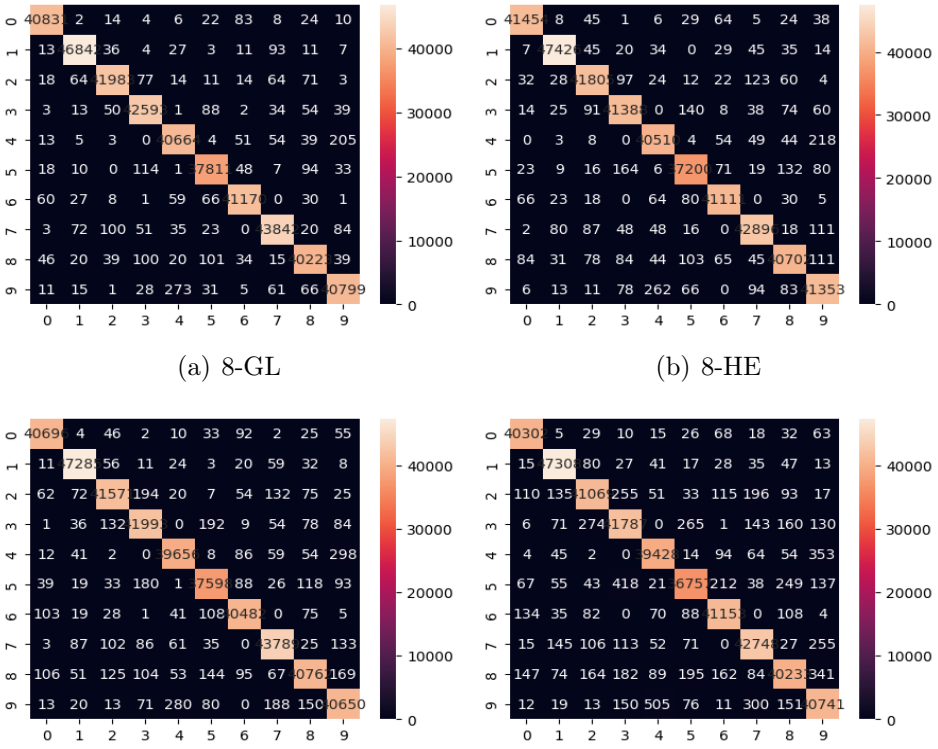


(a) 8-GL (b) 8-HE



(c) 64-GL (d) 64-HE

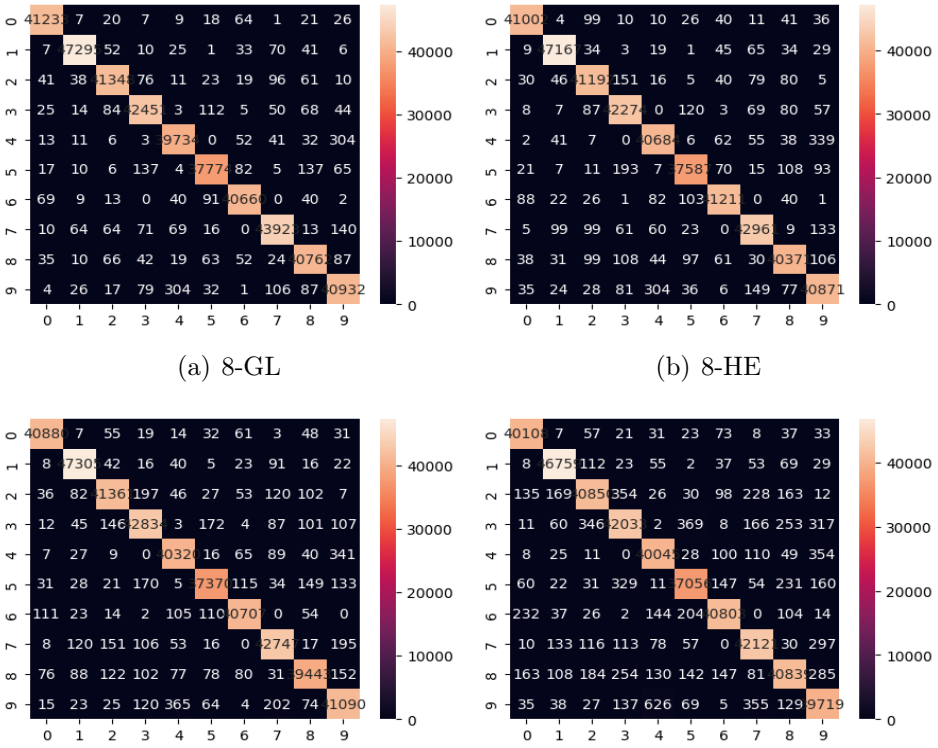
Figure 123 – Heatmap Mnist (GLN-ReLU)



(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 124 – Heatmap Mnist (Mish)

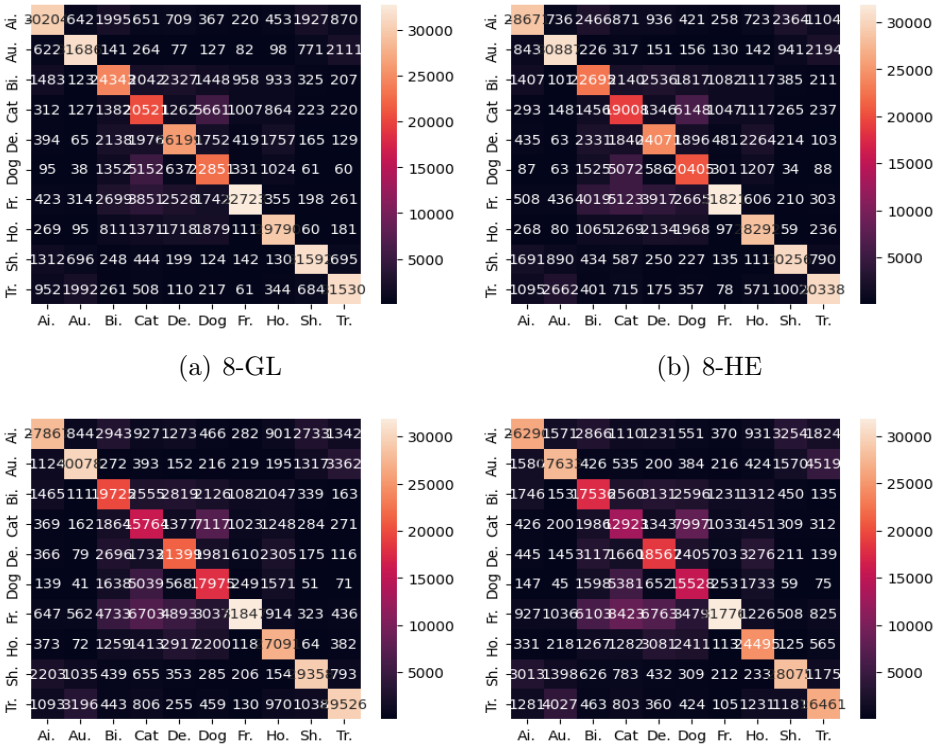


(a) 8-GL (b) 8-HE

(c) 64-GL (d) 64-HE

Figure 125 – Heatmap Mnist (GLN-Mish)

.6 CIFAR10



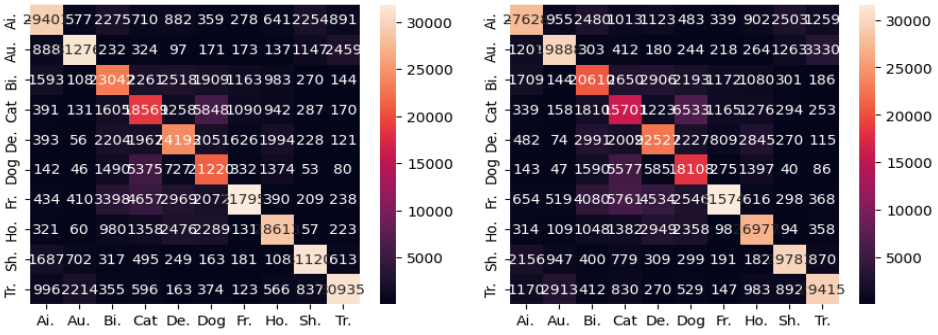
(a) 8-GL

(b) 8-HE

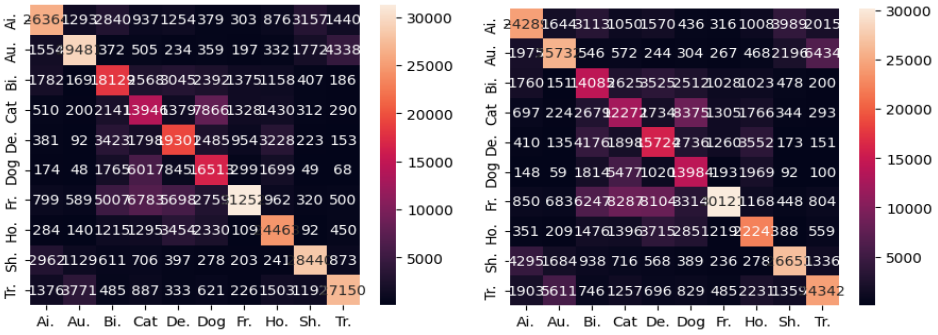
(c) 64-GL

(d) 64-HE

Figure 126 – Heatmap CIFAR10 (Tanh)

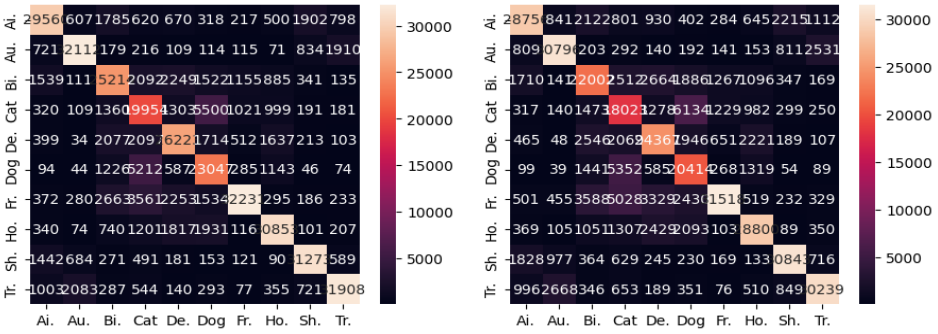


(a) 8-GL (b) 8-HE

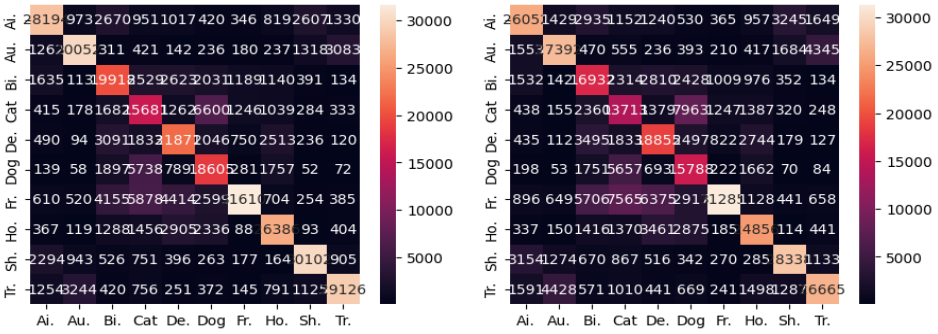


(c) 64-GL (d) 64-HE

Figure 127 – Heatmap CIFAR10 (Sin)

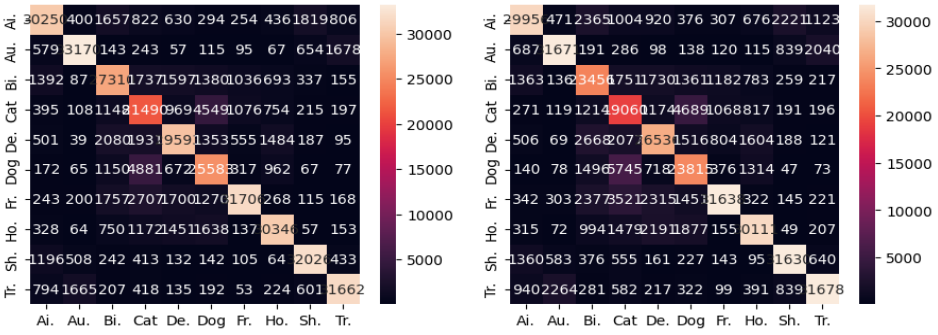


(a) 8-GL (b) 8-HE

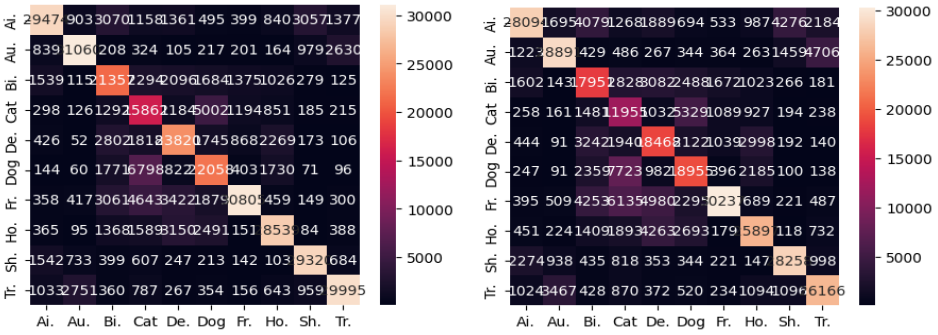


(c) 64-GL (d) 64-HE

Figure 128 – Heatmap CIFAR10 (GLN)

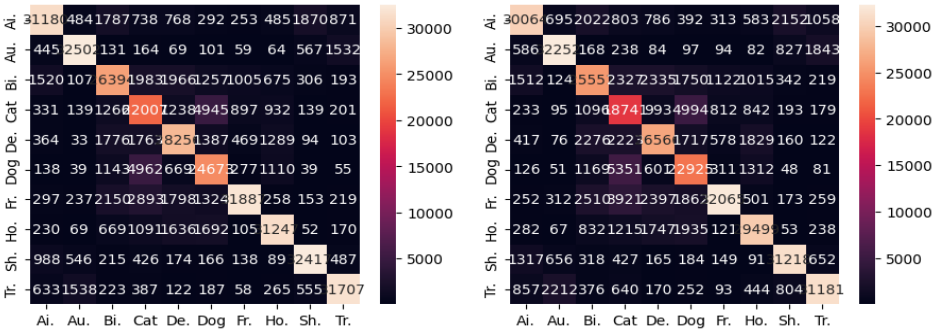


(a) 8-GL (b) 8-HE

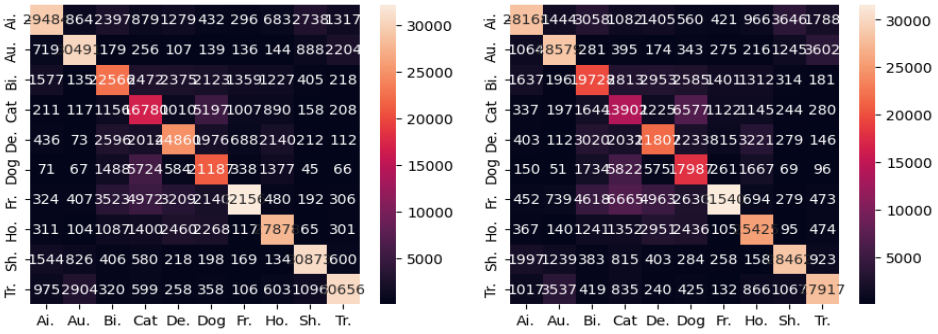


(c) 64-GL (d) 64-HE

Figure 129 – Heatmap CIFAR10 (ReLU)

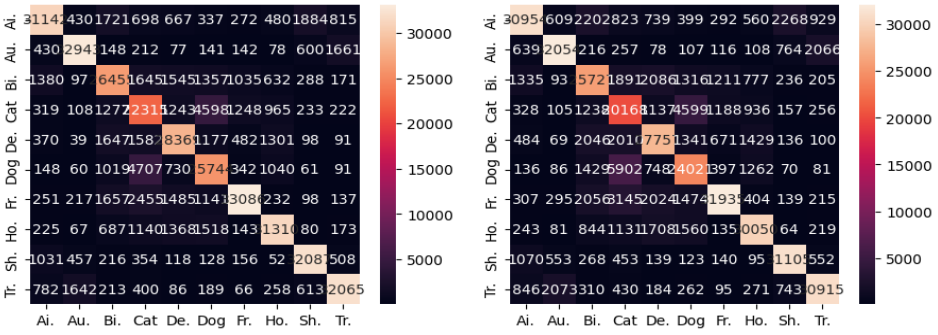


(a) 8-GL (b) 8-HE

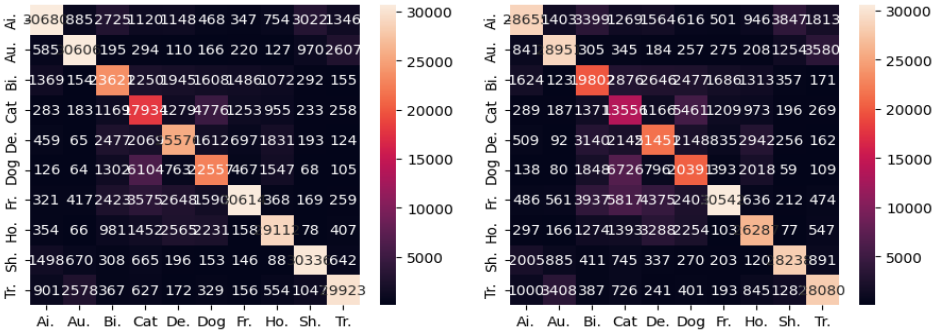


(c) 64-GL (d) 64-HE

Figure 130 – Heatmap CIFAR10 (GLN-ReLU)

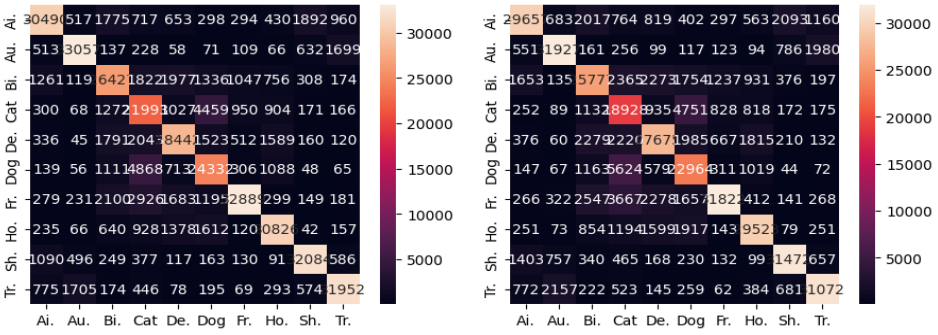


(a) 8-GL (b) 8-HE

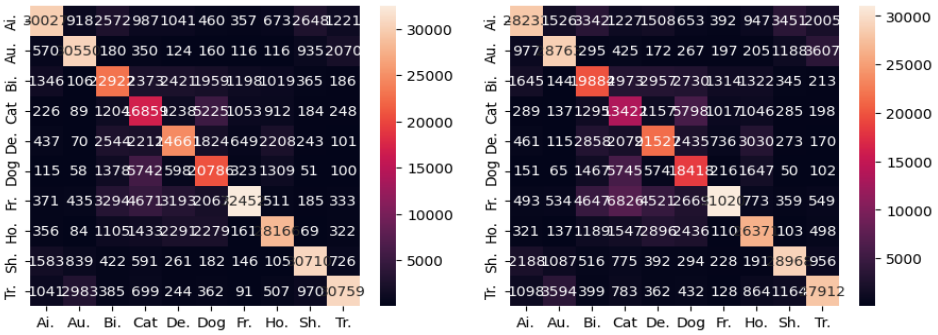


(c) 64-GL (d) 64-HE

Figure 131 – Heatmap CIFAR10 (Mish)



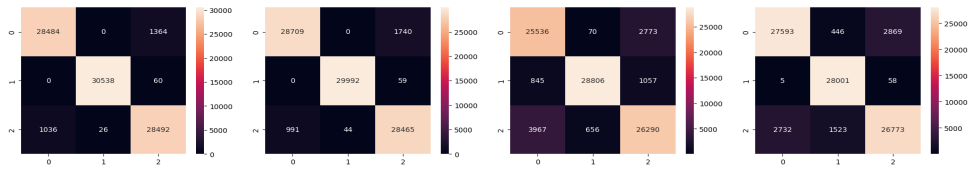
(a) 8-GL (b) 8-HE



(c) 64-GL (d) 64-HE

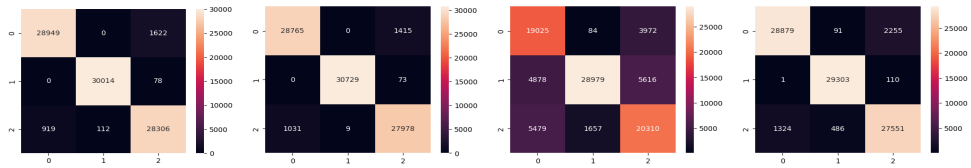
Figure 132 – Heatmap CIFAR10 (GLN-Mish)

.7 Lung Colon Cancer



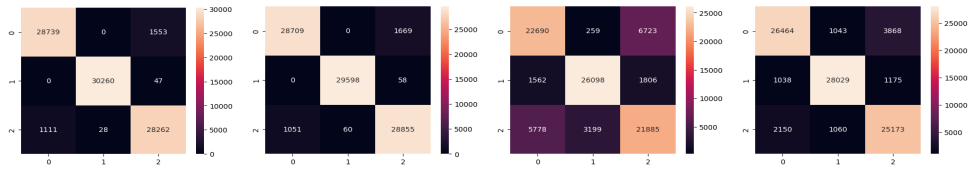
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 133 – Heatmap Lung Colon Cancer (Tanh)



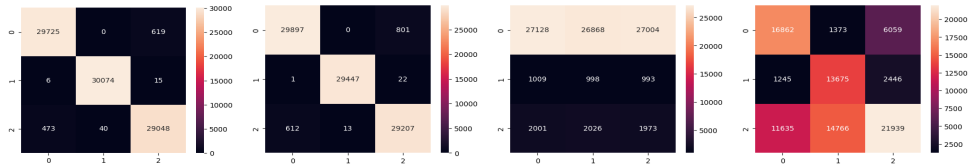
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 134 – Heatmap Lung Colon Cancer (Sin)



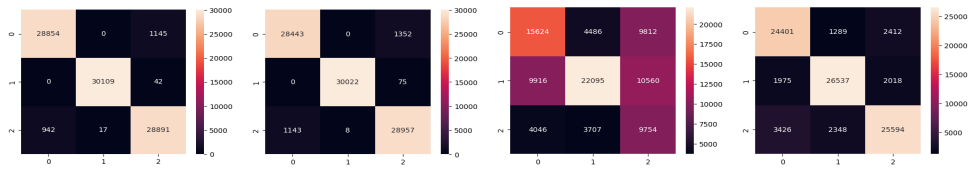
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 135 – Heatmap Lung Colon Cancer (GLN)



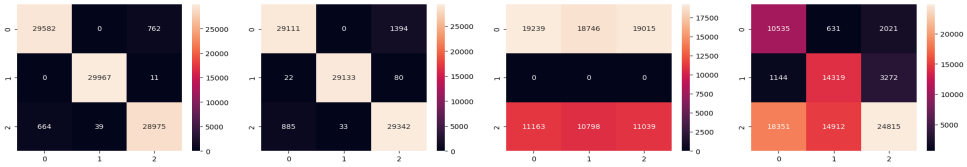
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 136 – Heatmap Lung Colon Cancer (ReLU)



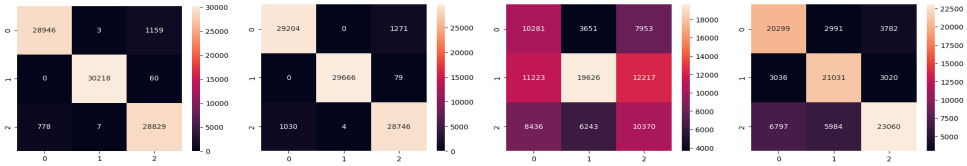
(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 137 – Heatmap Lung Colon Cancer (GLN-ReLU)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 138 – Heatmap Lung Colon Cancer (Mish)



(a) 8-GL (b) 8-HE (c) 64-GL (d) 64-HE

Figure 139 – Heatmap Lung Colon Cancer (GLN-Mish)