A critical evaluation of activation functions for autoencoder neural networks

MSc Research Project
Data Analytics

Vikalp Singh Tomar
Student ID: x20239441

School of Computing
National College of Ireland

Supervisor: Dr. Giovani Estrada

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Vikalp Singh Tomar		
Student ID:	X20239441		
Programme:	M.Sc Data Analytics	**Year:**	2022.
Module:	M.Sc Research Project		
Supervisor:	Dr. Giovani Estrada		
Submission Due Date:	15/08/2022		
Project Title:	A critical evaluation of activation functions for autoencoder neural networks.		
Word Count:	7027	**Page Count:**	19

Signature: Vikalp Singh Tomar

Date: 15/08/2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A critical evaluation of activation functions for autoencoder neural network

Vikalp Singh Tomar
X20239441

Abstract

A fundamental component of a neural network layer is the activation function. A lot of study has been done on such functions, so the neural network developer has many options. The most prevalent are probably Relu, Tanh, and Sigmoid. There are however several other functions, such as GELU, SELU, and PRelu, ELU that are available, but little is known about their performance for autoencoder networks. Therefore, in this research article, a comparative examination of multiple activation functions is conducted to determine which activation function best fits the autoencoder neural network.
According to this study, the activation function—which many researchers use when building neural networks—performed worse than the other activation function in terms of MSE. While a new combination of the activation function comes out with the best result in terms of mean square error as well as training time even after having larger number of training parameter.

1 Introduction

Deep learning has grown tremendously in recent years to address difficult issues such as Detection of an object, fraud detection, colouring the blank and white image, self-driving car, disease diagnosis, hate speech detection, natural language processing and so on. And with the increasing number of research in the field of deep learning, various types of models have already been developed such as convolutional neural network, artificial neural network, recurrent neural network, long short term memory. And so many different architectures have been proposed till now such as VGG16, ResNet, YOLO, autoencoder, fastRcnn, fasterRcnn, and many more (Theckedath and Sedamkar 2020) (Li, Song, and Fu 2018). However, the work of a neural network is that it extracted the useful feature from the source data and learn from it.
The main three part of the neural network is input layer, processing layer and output layer but loss function, weight initialization and optimization plays an important role (Carvalho and Ludermir 2007).
Whereas, the activation functions (AFs) are critical for neural networks since they add nonlinear qualities to the network, which is why AF(s) are referred to as non linearities (Hayou, Doucet, and Rousseau 2019) . To comprehend the utility of the activation function, one must first take a step back. A linear polynomial of just one like y equals 2x or y equals x. So, when this equation is plotted on graphs, it always produces a straight line. If more dimensions are added, they create a plane; if more dimensions are added, they form a hyperplane, but their shape remains straight and no curves at all that why they known as

linear. However, every equation in the universe is a non-linear polynomial of higher degree, such as the trigonometric function sin, cosine. When a nonlinear function is plotted, it always produces a line with some curvature. Linear equation are easy to solve but they are limited in their complexity but with the neural network, one should want to represent any kind of function. This is the reason neural network known as universal function approximators, which means they can compute any kind of function at all (Wang et al. 2022). Almost any process that can be seen as a function computation tying, such as naming a music, translating Spanish to English, and so on. That why there is need of a way to compute not just linear function but the non-linear ones as well. If the activation function is not used in the neural network no matter how many layer a neural network has, it will still behave as single layer network because assuming summing these layers would give just another linear function. This not strong enough to model many kind of data though but by using a non-linear activation function. The mapping of the input to the output is non –linear and we want it to be differentiable that means we can calculate the derivative of it. For example a differentiable function of x2 which can be differentiate it to 2x which is it derivateThis is required so that the back propagation optimization approach may be used to find a nonlinear error gradient to learn complicated behaviour. The activation function is designed to approximate how neurons communicate with one another in the brain. Each one is activated by its potential; if it exceeds a specific threshold, it is known whether it activates a neuron or not. The activation function duplicates the spike train of the brain's action potential. So one can think about lots of activation function but how do a person know which one to use this choice is depend of couple of factors. The three most popular ones sigmoid, Tanh, Relu[1].

In this research article, the focus is on which activation is working best with convolutional autoencoder neural network in term of accuracy, training time. There are so many research had been done in this field but mostly the researches had been either using the different dataset for different activation function or using different architecture with different activation function. So still, there is need of fair comparison of different activation function on a same architecture with the same data. Thus, in the research the an autoencoder neural network has been used to do the study and a common dataset has been taken into account and also different combination of activation function and architecture has been tried in order to find the best model accuracy.

In the neural network, the input weighted sum is transferred to get a desired output through the node in a layer of network. This all defined by the activation function. The neural network's capability and performance are significantly influenced by the activation function that is selected, and different activation functions may be applied to different parts of the model. In addition, the autoencoder is the one network that have same number of neuron in input and output layer thus the network mostly learn from the hidden layers where we can try and applies different activation function to see the effect of every activation function, which can help us to compare different activation function.

[1] https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

2 Related Work

In this section of the report, the previous research that has been carried out in this field has been discussed which help to get good knowledge about the activation function, their evaluation and uses.

Let’s first discuss what is neural network is, Kukreja et al. (2016) explained What is a neuron? What is an artificial neuron? The human brain has billions of neurons. A neuron is a very tiny unit that performs sensory tasks. When any information from the sensory organs is delivered, neurons will aid to stimulate the reaction. When the same functionality is performed artificially, it is referred to as an artificial neural network. Neural network form the base of deep learning a sub field of machine learning where the algorithm has been inspired by the structure of the human brain (Mijwel 2018). Neural networks take in data and train themselves to find patterns in the data and predict outcomes for fresh sets of comparable data. A neural network is made up of layers of neurons, and these neurons serve as the network's processing unit. The input layer receives the input, the output layer predicts the eventual result, and the hidden layers do the majority of the calculations necessary by the network (Maind, Wankar, et al. 2014). Each pixel of an image has been fed to the input to each neuron of the first layer neural network. Neuron of one layer has been connected to neuron of the next layer through channels each of these channels has been assigned a numeric value known as weight. the inputs has been multiple to the corresponding weights and their sum is sent as input to the neuron in the hidden layers each of these neuron is associated with a numeric value called the bias which then added to the input sum. This value then pass through a threshold function known as activation function. The result of the activation function determines if particular neuron will activated or not an activated. Neuron transmits data to the neurons of the next layers over the channels in this manner the data is propagated through the network, which is called as forward propagation in the output layers the neuron with the highest value fires and determines the output the value are basically probability. So in order to make right prediction and get highest probability of getting the right output the model has to be trained not only by feeding the input also by feeding the output. The predicted output is compared against the actual output realize the error in prediction the magnitude of the error indicates how wrong one can be. These all information then transferred backward through the network, which is known as back propagation. Now based on this information the weight are adjusted this cycle of forward and backward propagation is iteratively performed with multiple inputs and this process continues until our weight are assigned such that the network can predict the output correctly. This ends the training process. It can take hours or months to train a neural network, but the time is an acceptable trade-off when contrasted to its scope. There are so many example present to see the usefulness of the neural network such as forecasting neural network are trained to understand the patterns and detect the possibilities of a rainfall or arise in stock price with high accuracy (Basha et al. 2020) (T. Kim and H. Y. Kim 2019). Music composition neural networks can even learn patterns and music and train itself enough to compose fresh tune.

2.1 Convolutional Autoencoder

According to Hou and Yan (2020) autoencoder one of a type of artificial neural network.it is an interesting variant with two important change: First, the number of neuron is the same in the input and the output therefore it can be expect that the output is an image that is not only the same size as the input, but actually is the same image. Now this normally would not make any sense, why would researcher wants to invent a neural network to do the job a copying machine?

So here's the second part: there's a bottleneck in the middle of these levels, which means the amount of neurons in those layers is much lower than normal, so it needs to figure out how to represent this type of data as best it can with a considerably lesser number of neurons. Because of this, the output photos are not a carbon replica of the original images, but rather a significantly more accurate depiction of the input images. These autoencoders may generate sparse representations of the input data and can thus be utilized for image compression. Autoencoders provide no visible benefit over traditional picture compression techniques such as JPEG. As a sliver of solace, many distinct variations exist (Raghavendra et al. 2019).

There are denosing autoencoder that after learning these spares representation can be presented with noisy image. As they more or less know, how this kind of data should look like, they can help in denosing the images. (Yu et al. 2019). There are other types of autoencoders available, such as variational autoencoders, which can not only learn the spare representation but also design new images (D.Li, Zhu, and Lin 2017).

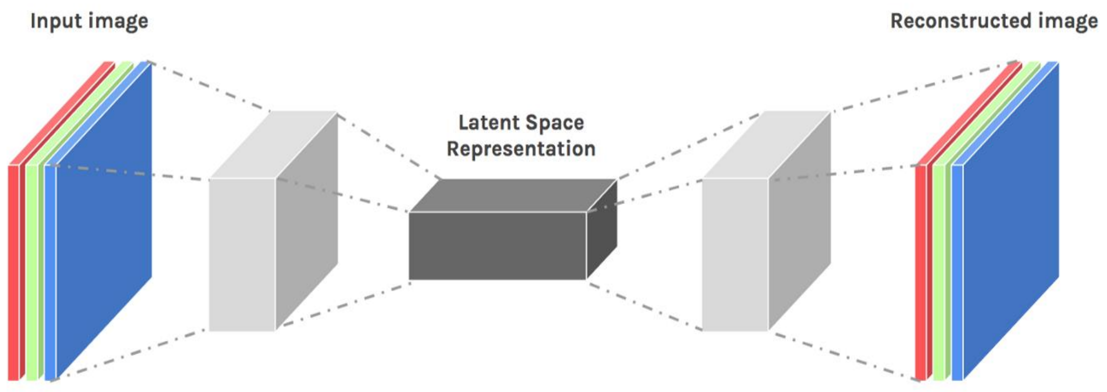

Figure 1: Auto-encoding an RGB image with two Conv2D followed by two Conv2DTranspose[2]

Therefore, for this study only a convolutional autoencoder is taken in account, which consist of two part. There is, first of all an encoder that is the first layer which takes in some input and learn how to efficiently compress and encode that data into something that known as the "code". Then there is decoder that learns how to reconstruct that encoded data representation. In addition, eventually creates the output. And that output is as similar to the original input data as possible. Effectively, an autoencoder learns to recognise which aspects of observable data are relevant, and limit noise in data that can be discarded. Separate the signal from the

[2] https://hackernoon.com/autoencoders-deep-learning-bits-1-11731e200694

noise. Convolution autoencoders have a variety of use cases related to images. By learning, what makes up the signal and what makes up the noise researcher can also have the ability to detect when something is not a part of the status quo, and that anomaly detection. Anomalies are a significant deviation from the general behaviour of the data, and convolutional autoencoder are a very good at telling us when something does not fit. As result convolutional autoencoder are widely used in anomaly detection in things such as fault, fraud, and intrusion detection. So convolutional autoencoder are a great way of compressing noise, recognizing relevant feature and detection anomalies. A handy toolbox for handling with all sort of data

2.2 Activation Functions

It has been discussed before that that the activation function is the function that takes one from the input to the output for a perceptron, now let discuss about the different activation function that are used in different studies such as sigmoid, tanh, relu, elu, prelu and many more. An output of basic activation function can be a X b where a is an input and b is constant. This type of activation function knows as linear activation function. Such types of activation have no additional non-linearity in the neural network but it is important to have non-linearity in the network otherwise network will produce the same result as the input despite of have many no. of layers. So to add the non linearities in the neural network activation function come into play.
According to Bawa and Kumar (2019), Sigmoid has the mathematical form of f of x equals one over one plus e to the negative X it takes some number and squashes it into a range between 0 and 1. It is one of the first to be used because it could be interpreted as the firing rate of a neuron where zero means no firing and one means a fully saturated firing it pretty easy to understand but it has two problem that have made it fall out of popularity as per Ding, Qian, and Zhou (2018). The first is that causes our gradients to vanish when neuron's activation saturates close to either zero or one the gradient at these regions is very close to zero during back propagation this local gradient will be multiple by the gradient of this gate's output for the whole objective so the local gradient is really small. It will make the gradient slowly vanish and close too no signal will flow through the neuron to its weights and recursively to its data the second problem is that its output is not zero centred. It starts from zero and ends up at 1 which means the value after the function will be positive and that makes the gradient of the weights become either all positive or all negative. This makes the gradient updates go too far in directions, which makes optimization harder.
So, In order to improve on it there is Tanh function was introduced. According KILIC¸ ARSLAN, Kemal, and C¸elik (2021) hyperbolic tangent function (Tanh) squashes the real number into a range between a negative one and one instead of zero and one. That's why its output is zero cantered which makes optimization easier. So in practice its always preferred to sigmoid but just like the sigmoid it also suffer from the vanishing gradient problem (Bengio, Simard, and Frasconi 1994) .
To overcome from the problem of vanishing gradient Rectified linear unit activation function was developed. This activation function become so popular in the last few years. It Just max (0,x)which means that the value is zero when x is less than zero and linear with the slop of

one when x is greater than 1. It is noted that it had a 6x improvement in convergence over tanh and a landmark in the ImageNet classification paper by Krizhevsky, Sutskever, and Hinton (2012).A lot of time in computer science it has been seen that the most elegant and simplest solution is the best this applies to relu AFs too. Well it does not involve expensive operation like tanh or sigmoid that why it learn faster and it avoids the vanishing gradient problem almost all deep neural network uses relu now a days. However, the problem is it only used for the hidden layers the output layers uses softmax function or sigmoid function for classification since it gives probability for different classes and a linear function for regression function since the signal goes through unchanged. The main problem with the relu activation function is that some unit can be fragile during training and die meaning a big gradient flowing through a relu neuron can cause a weight update that makes it never activate on any data point again so then gradient flowing through it will always be zero from that point on. So to tackle this problem a new version of relu was introduce called as leaky relu (Bedi et al. 2022).
Instead of function being zero when X is less than zero. Leaky relu has a small negative slope refer to figure 2. Another common option is max out, which is a generalized form of relu and leaky relu but doubles the number of parameters for each neural, therefore there is a trade-off.

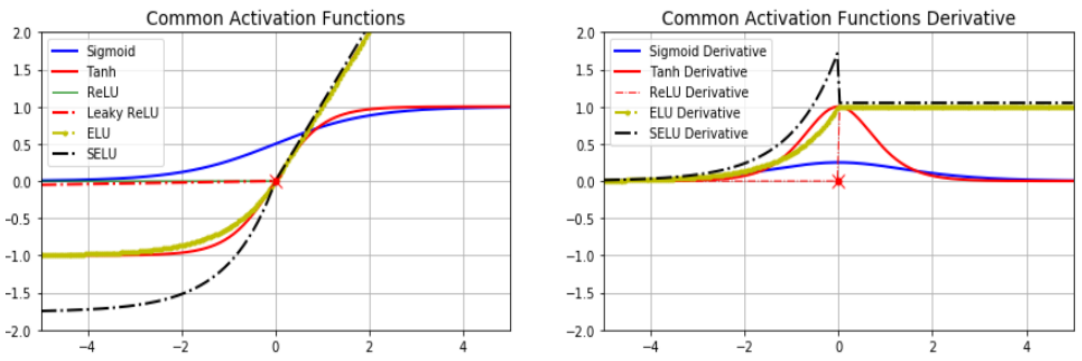

Figure 2: Different types of activation function in neural network[3]

In the recent time, like Leaky Relu other many more activation function has been developed like Selu,Gelu, PRelu, and Elu these all are the modified version of relu. The Elu activation function in comparison of Relu contain negative value that allows them to drive mean unit activations closer to 0, similar to batch normalization, and with less computing cost (Clevert, Unterthiner, and Hochreiter 2015) Similar like Prelu (He et al. 2015). However, the difference between these two-activation function are that Prelu do not guarantee a noise-free deactivation condition.
Klambauer et al. (2017) first introduced Scaled exponential linear unit .In comparison of relu, selu also solve the problem of Vanishing gradient in activation function and also Selu AF never die. Thus, the learning rate of selu is quit faster as compare to other activation function.
Gaussian error linear Unit (Gelu) is the activation function is recently developed in year 2016. It just a combination of Tanh and approximate number .Google Bert transformer use

[3] https://medium.com/@kshitijkhurana3010/activation-functions-in-neural-networks-ed88c56b611b

this activation function. It perform best for the NLP transform models[4] and avoid the vanishing gradient problem (Nguyen et al. 2021). From the figure 2, it can be easily concluded that how different activation function gives the output value which is the weights multiply by input plus bias. Different activation function have different range of output values which is why different activation gives different output. Also from the figure, we can see value of the derivate of different activation function, which help the neural network in learning during backpropagation differently.

All the previous research that has been carried out have used the different activation function with different architecture and different-different dataset. Little is known on the activation function and architecture to use for autoencoders. Most people will just try a number of combinations. Thus, in this paper, all the above mention activation function has been evaluated with an autoencoder neural network also with a common RGB image dataset of wheat head. In order to compare different activation function the mean square error and training time were calculated.

3 Research Methodology

The approach utilized to conduct the research is detailed in this section. In addition, the design difficulty encountered throughout the research is also discussed here.

3.1 Methodology

The Knowledge Discovery in database (KDD) has been implemented to recognize a wheat head form the high definition open field images. KDD is known as the process of finding the useful knowledge from a collection of data and the knowledge that was gained can be feeded backward to the start of the cycle which can be seen through the figure no 3. This help to build a process more efficient at the end. The modified KDD flows a six-step road map that are understanding of problem and data, data preparation data mining, evolution of the knowledge that has been discovered and the use of the discovered knowledge. For this research work all the steps has been taken into account

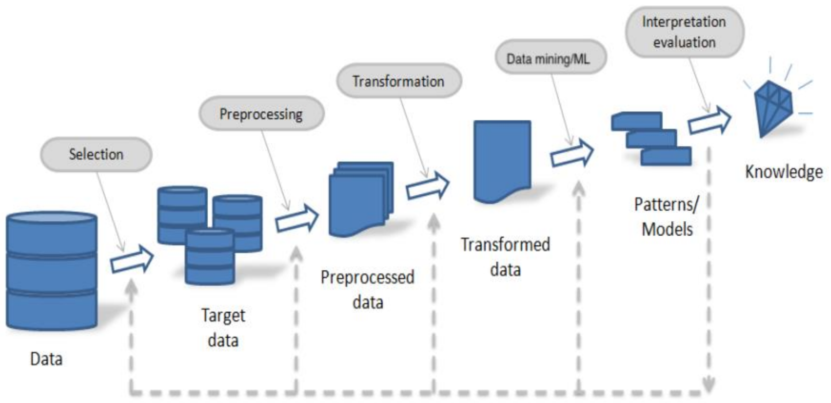

Figure 3: Different types of activation function in neural network[5]

4 https://mlfromscratch.com/activation-functions-explained/#/

[5] http://www.e2matrix.com/blog/2017/10/10/data-mining/

3.1.1 Understanding of problem

When it comes to understanding the problem which is a fundamental step of KDD. first, the need to identify the problem statement and the goal and the result from the customer's point of view need to be taken into account when a problem has been defined. The issue statement here is to discover the optimal architecture and activation function that perform efficiently and quickly in terms of recognizing wheat heads in open field photographs all over the world. A road map has also been proposed here to carry out the data mining aim, as well as the tool and approach that may be used further in the process.

3.1.2 Understanding of Data.

For the study, the 3491 high definition RGB images of open farm field has been taken into account, which was, gather by the nine different institute from seven different countries.
The date have different variety of wheat, which are differ in colour, genotype, size etc. In addition, all the images are taken at different location, different condition, different light and different weather that makes the data more complex. For this study, the images are classified into two categories one images that contain the wheat head and second is no wheat head. Some of the images from the dataset has been shown in below figure 4 and from the figure it can be justified the dataset have diversity in term of image acquisition (David et al. 2020).

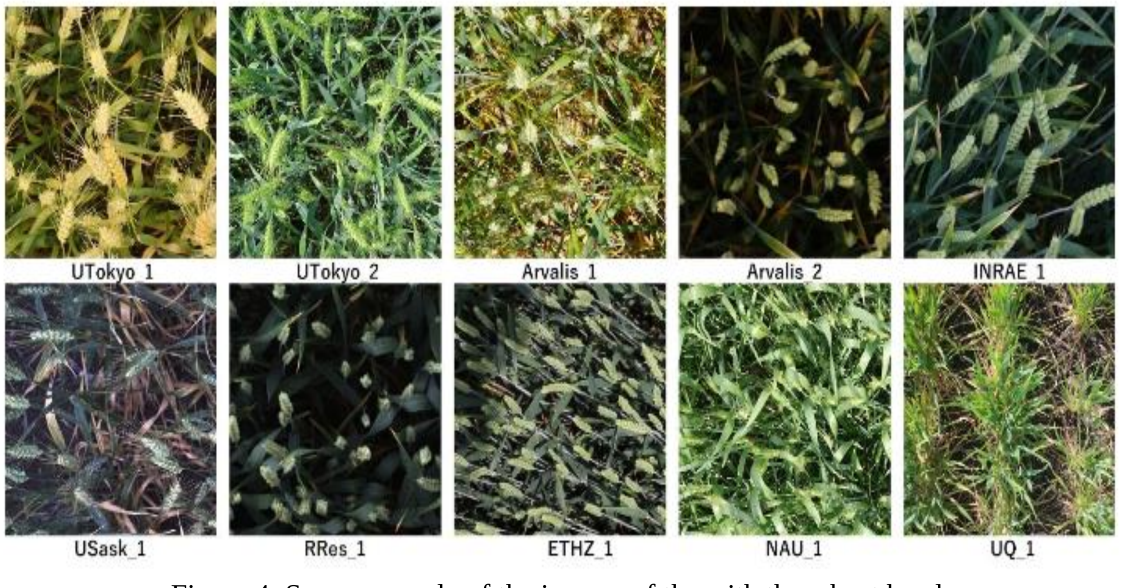

Figure 4: Some example of the images of the with the wheat head.

3.1.3 Preparation of data

Following data comprehension, the next stage is data preparation, in which raw data has been turned into structured data. As a result, it is ideal for model building. This step is pretty much time consuming as compare to other part because in this part several preparation process has been completed more accurately because it plays an important role in the further steps.

In the data preparation first the image size is reduce to 128*128 because by reducing the size of the image the training time can be reduced. The data is then separated into train and validate to feed the neural network, and the images are rescaled to build a new version of the current image with varied dimensions, which improves the neural network's training accuracy. Image datagenrator class which has been used in the study to which help to augment the images in the real time while the model preforming the training task

3.1.4 Data mining

In this section, we have defined the architecture of convolutional autoencoder neural network as base model using tanh as activation function. In addition, this model was run to determine if there is any knowledge that can help us find the solution to the desired research topic also previously defined model like CNN was also tried. Then in the next step the model is been train with different activation function as well as different architecture of autoencoder such as combination of two autoencoder and combination of multiple activation function in a single network. Then these neural network model has been evaluated. And to evaluate the MSE score, training time was taken into consideration, and multiple visualizations were employed to display and comprehend the results. The detail implementation of the different autoencoder has been discussed in the implementation section 5.

4 Design Specification

The two-layer architecture has been used in the research. The two-layer design combines the logic and data layers, which means that all logic coding and training are done in the business layer, which facilitates in data selection, pre-processing, and transformation, and in the presentation layer or known as client layer, the knowledge that is gain in the first layer can be presented and visualised.

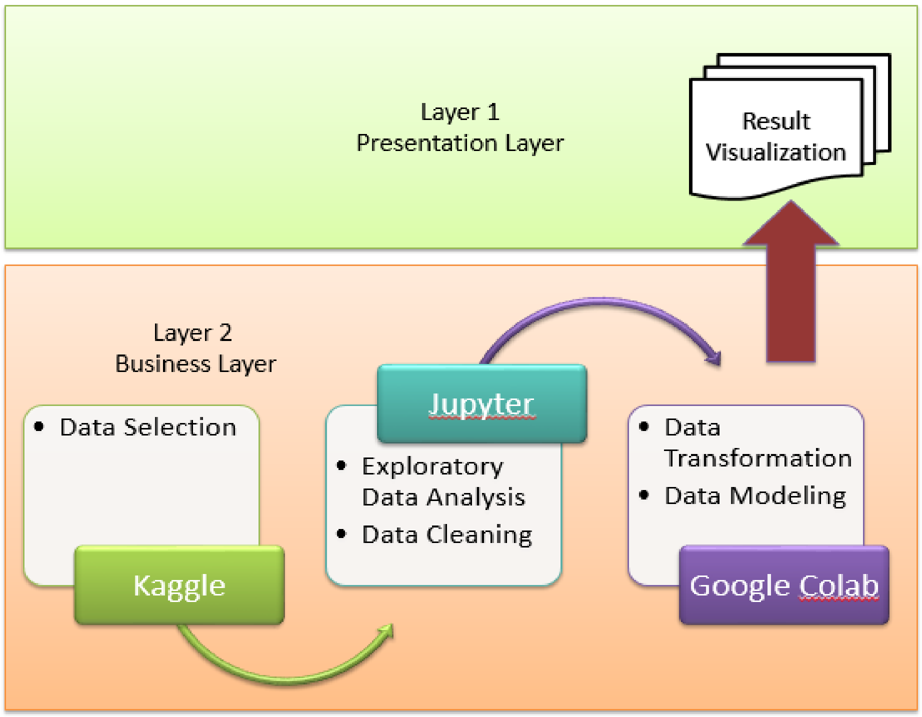

Figure 5: Layered Architecture

5 Implementation

The different activation and combination of AFs function for an autoencoder have been implemented and analysed in this area of the paper. Before reaching to this step exploratory data analysis, data cleaning, transformation has been carried out in the above sections. The batch size is 64 is fixed for every model so also the epoch has been fixed to carried out the fair comparison between the training of every model different model and activation function. The data have 3491 RGB images of open field which are divided into 3 folder train (images have grains), no grain (image have no grain) and test. We put approx. 10 images from the train folder to test folder so we can use the test image for validation during training process.
So the train , no grain and test folder contains the images 3422,59 and 10 respectively.
As earlier discussed in pervious section an autoencoder goes from the large size to a smaller size and then back to a large size in terms of image size. Therefore, in the base model architecture we start with 64x64 filters and goes down to 16x16. The image sizes goes down as we employs maxpooling of 2x2 and then we doing the upsampling 2x2 so that we can create a mirror image of encoder part. Also symmetrically going back in term of filters as well. Finally, because the pictures are RGB, the last layer provides with three channels as per the output required. We use the sigmoid and combine it with a mean square error because comparing the input picture to the output image using mean square error is an excellent method.

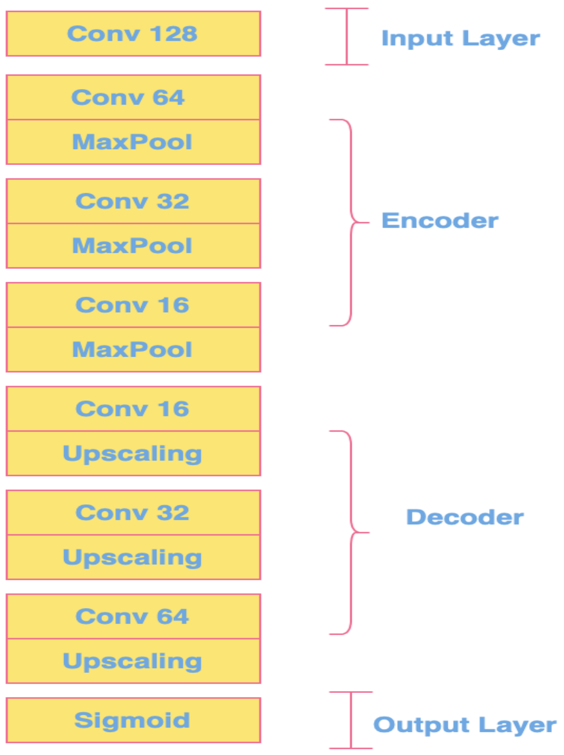

Figure 6: Convolutional Autoencoder Neural Network

By looking at the summary of the model, it can be seen that the encoder part started with 128x128x3 and the next layer 64 and gradually it goes down to the bottleneck layer which is 16x16x16 after that the decoder part come which is an exact mirror image of the encoder part that discussed above. This is the base model and the architecture can be seen in figure 6 below, which has been trained by using different activation function for 25 epoch for every different activation function that are testes using the model-fit. After that the validation and training loss has been plotted to look how training went on which was discussed in the next section of the report.

6 Evaluation and Result

In this segment of the study, we will talk about the criteria and parameters that has been used to evaluate and compare the models. In addition, results of a different model that has been implemented throughout the study has been discussed in the section. To evaluate the different model a mean square error and the training time are taken into account, because as MSE work best in comparing the input and the output image. There are two main advantages of finding the mean square error. The first one is to find the correctness of decryption algorithm and second one is to find if the original image has been retrieved at the output end without any distortion or not (Nicolson and Paliwal 2019). The training time is also the second assessment criterion that was used to evaluate the model. As an example, if the model has the same mean square error or is extremely near, training time will have a crucial effect to check which model is best fitted model. In the further sub-section the result has been discussed of different model on the basis of evaluation matrices.

6.1 Autoencoder using TanH activation function.

The first convolution autoencoder using a Tanh activation function has been performed on the wheat head image dataset. TanH AFs have a vanishing gradient problem, as was previously discussed; however, no such issue materialized while training the model with this AFs. Thought the learning the model showed a continues learning and the mean square error for training data reduce to 0.277 whereas the MSE for validation 0.0305. This can show from the below graph (Figure 7). From the graph, it can be seen that the validation loss during the training is not smooth as compare to training loss.

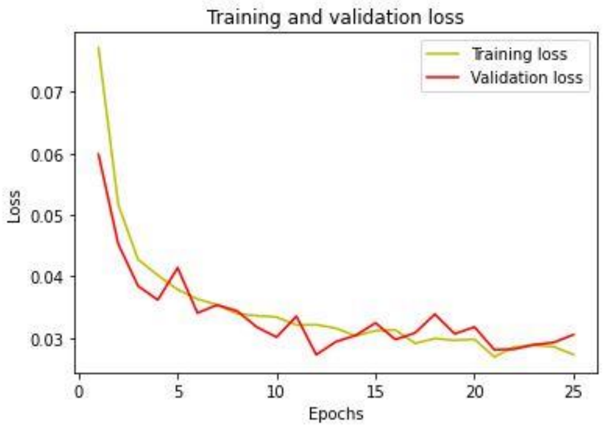

Figure 7: Training & validation MSE loss graph for TanH AFs

6.2 Autoencoder using ReLU activation function.

For the second model, convolution autoencoder using a Relu activation function has been applied on the same data set with the same number of epoch. and it can be seen that the mean square is in this case achieve the value 0.0294 for training and 0.0309 for validation.

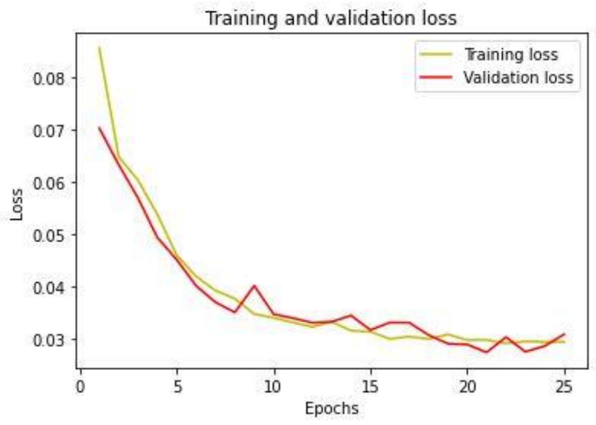

Figure 8: Training & validation MSE loss graph for ReLU AFs

From the graph, it can be seen that the validation loss during the training is smooth as compare to training loss but now we cannot say that it is a best fitted model

6.3 Autoencoder using PReLU activation function.

In the next model, the Prelu activation function was used to train the model. As previously explained, Prelu adds a negative parameter to the function's negative side, which aids with the problem of dead neurons in the relu section. It helps the model to train more correctly. when the autoencoder has been trained with the Prelu activation function. It has been observed that MSE loss for training is 0.0277 whereas the validation loss achieve by the model is 0.0280.this can be shown by the below figure 9.

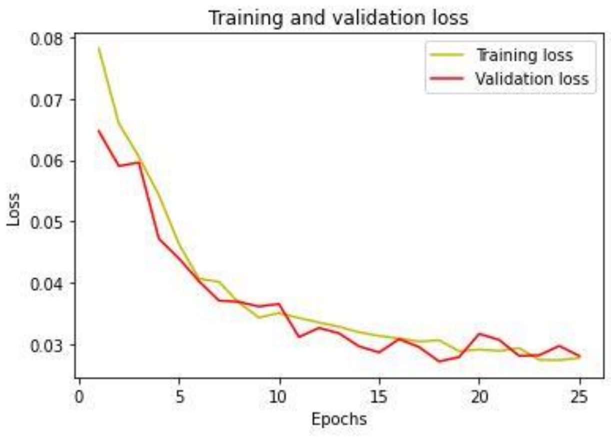

Figure 9: Training & validation MSE loss graph for PReLU

However, the problem with this model is that for training each step it takes 9 sec that is much higher it just because this activation function generate more parameter as compare to above two model.

6.4 Autoencoder using ELU activation function.

For the third case, the activation function used is Exponential Linear Unit. It also fix the dead neuron and vanishing gradient problem also aid the network to adjust the weight and the biases to in proper direction as by generating the negative output. When the model has been train with this activation function, the mean square error achieved by the model is 0.275 for training and validation loss is 0.0289. Also it takes less time as compare to Prelu. It can be also seen by the below training validation loss graph (Figure 10).

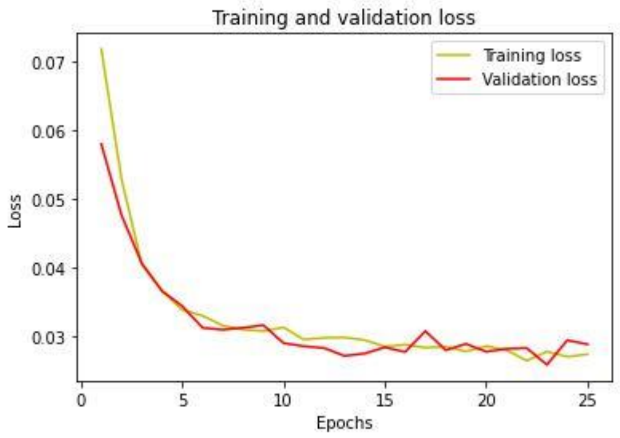

Figure 10: Training & validation MSE loss graph for ELU AFs

6.5 Autoencoder using SELU activation function.

Now the next activation function is used is Scaled exponential linear unit. It reduce the internal normalization as it used banach fixed-point theorem. If the input is giving within a pre-defined interval and activation function is within a pre-defined interval then output generate is also within the predefined interval. Because it multiples the input with the activation function to length the pattern in the data. This function propagate through many neural network and converge toward minimum, which help to robust the model efficiency. This also showed in the result, the mean square error achieved by this model is 0.0273 for training and 0.0264 for validation part showed in figure 11. In addition, it takes 7 sec/step to train the model.

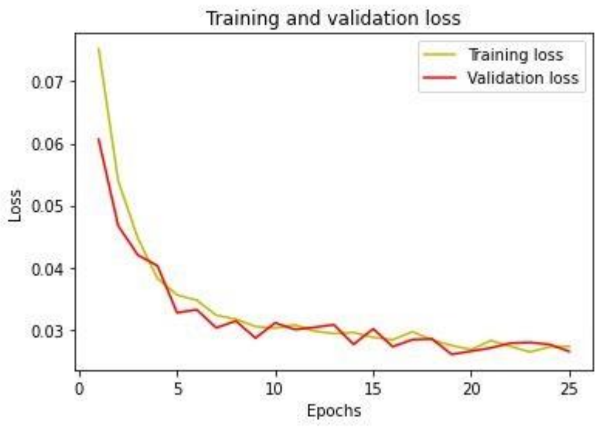

Figure 11: Training & validation MSE loss graph for SELU AFs

6.6 Autoencoder using GELU activation function.

For further study it we used Gaussian Error Linear Unit (Gelu) activation function. As we already discussed about the benefits of the activation Gelu. When the model is trained by using this activation function, it has been observed that the mse for training and validation is 0.0289 and 0.0314 respectively showed in figure 11. Moreover, because of its complexity this activation function achieve more training time as compare to other activation function.

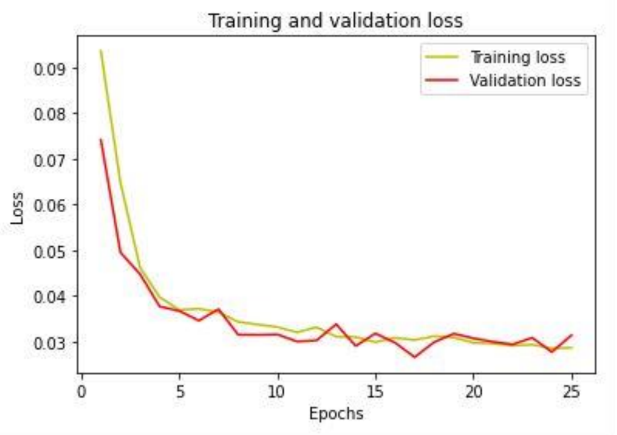

Figure 11: Training & validation MSE loss graph for GELU AFs

6.7 Autoencoder using multiple activation function.

For more study, we also use the multiple activation function in the single autoencoder. We use relu in the first layer then prelu and elu in the next two layer of the encoder and decoder part is just the mirror image of encoder part. By making this change, the loss was decreased from 0.0725 to 0.0268 for training and from 0.064 to 0.0245 for validation which also can be seen from the below figure 12, which is the best so far.

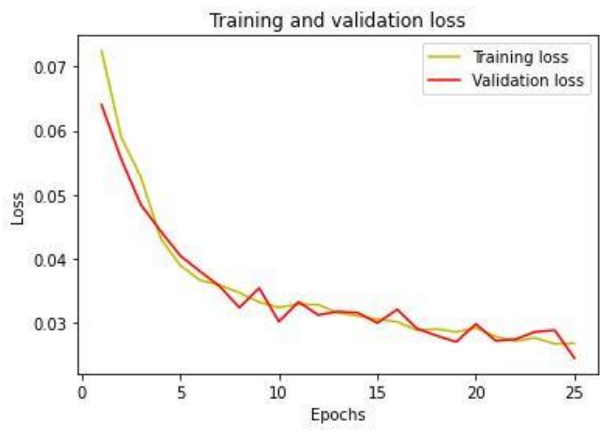

Figure 12: Training & validation MSE loss graph for (ReLU+PReLU+ELU) AFs

7 Discussion

In this section, we are going to discuss about the finding of the cases that has been performed throughout the resersearch. When we evaluate all of the models with different activation functions, we discover that the combination of activation (Relu+Prelu+ELU) performed significantly better than other activation. Furthermore, when all of the models are trained on the same dataset and with the same amount of epochs, it takes less time to train the model to produce results that are more accurate showed figure 13 even with the increased number parameters. Therefore, from the studies it can be said that the combination of activation function can be used with convolution autoencoder to perform the image identification task. Also to increase more model accuracy, one can train the model with a greater number of epochs or feed the network with a greater number or variety of pictures.

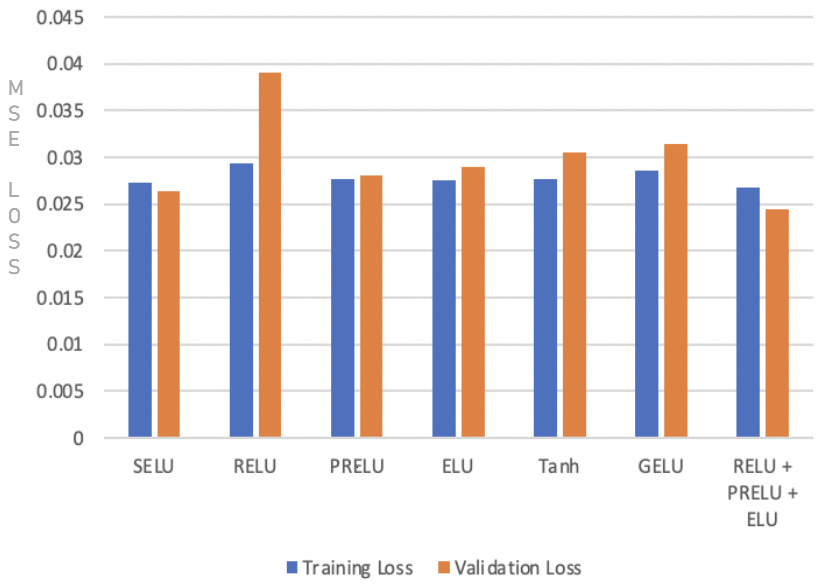

Figure 13: Comparison of different activation function.

Table 1: Result comparison of all the models

Activation Functions	Training MSE Loss	Validation MSE Loss	Time/step
Tanh	0.0277	0.0305	8sec/step
RELU	0.0294	0.039	8-7sec/step
PRELU	0.0277	0.028	9-8sec/step
ELU	0.0275	0.0289	8sec/step
SELU	0.0273	0.0264	9-8sec/step
GELU	0.0286	0.0314	9-8sec/step
RELU + PRELU + ELU	0.0268	0.0245	8-7/steps*

*The number of parameter is lager as compare to other activation function

In this study, a variety of activation function combinations, including relu+prelu+elu, prelu+relu+elu, and prelu+relu+prelu, have been tried. However, the combination of relu+prelu+elu produced the best results with the least amount of training time and the greatest number of training parameters. Consequently, only the outcome of this AF combination has been demonstrated and discussed, as shown in table 1.

8 Conclusion and Future Work

The convolutional autoencoder neural network was utilized in this research to examine and evaluate various activation functions. If no AFs are utilized in the network, the neural network can only learn linear patterns. To train the neural network with the non-linearity of the data AFs are used. In the case of an autoencoder, images are recreated from input data. It is critical that the autoencoder understand the nonlinearity of the data very effectively, which is where activation functions come in. That is why it is critical to conduct extensive research. The first section of the research emphasized how there are numerous activation functions available and how difficult it is to determine which activation function works best with their autoencoder neural network because there is less information available. So, a wheat head image classification challenge was examined to illustrate how effectively it works in order to handle this issue. After experimenting with various activation functions on the network, it was discovered that the combination of relu+prelu+elu performed the best for the convolution autoencoder when compared to other activation functions. In this study, we look at the most common AFs, such as Tanh, Relu, ELU, PRelu, as well as newly discovered AFs, such as SELU, GELU. And, any combination of the most popular activation functions has been considered as shown in section 6. The training time of the autoencoder neural network using the combination of AFs take less time (8sec/step) to train the model even with the large number of training parameter. Also the MSE loss is minimum in comparison with the other AFs. As shown in figure 13 and table 1. This study clearly shows that the combination of numerous activation functions, which were not previously employed in neural networks, may be used to provide more precise and accurate outcomes with less training time. Moreover, the most popular activation function Relu performed poorly the main reason behind this is dead neuron problem which relu always faces. Because of that during back propagation the neuron that have negative weight considered as a zero by Relu which never activate during the process which obviously effect the performance of the network. Whereas the combination of Afs can overcome from this problem as Elu and Prelu take account the negative weight during learning which increase the performance of network and Relu helps to reduce the input data which help in fast learning. As a result, by integrating all AFs with one another in different layers, the neural network is able to maximize performance and minimize training time.

As more novel AFs are produced, the combination of some other activation function may be able to perform much more successfully in the event of image anomaly. Also these combination of Afs can be performed with the new novel architecture of autoencoder.

All potential combinations of AFs with the convolutional neural network will be tested in future study. In addition, we will train the model with a bigger set of image data as well as add a larger number of layers to this neural network. In addition All of these activation functions may be done on the other neural network. And many more activation functions will be applied to this convolution neural network in order to test and compare whether new activation function or combination of those activation function can improve the efficiency of the neural network or not.

Acknowledgement

I would want to offer my heartfelt appreciation to my supervisor Dr. Giovani Estrada for his unwavering assistance and excellent suggestions throughout the research study.

References

Basha, Cmak Zeelan et al. (2020). “Rainfall prediction using machine learning & deep learning techniques”. In: 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, pp. 92–97.

Bawa, Vivek Singh and Vinay Kumar (2019). “Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability”. In: Expert systems with applications 120, pp. 346–356.

Bedi, Parminder Pal Singh et al. (2022). “Increasing the Versatility of Leaky ReLU Using a Nonlinear Function”. In: Advanced Machine Intelligence and Signal Processing. Springer, pp. 433–442.1

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). “Learning long-term dependencies with gradient descent is difficult”. In: IEEE transactions on neural networks 5.2, pp. 157–166.

Carvalho, Marcio and Teresa B Ludermir (2007). “Particle swarm optimization of neural network architectures andweights”. In: 7th International Conference on Hybrid Intelligent Systems (HIS 2007). IEEE, pp. 336–339.

Clevert, Djork-Arn ́e, Thomas Unterthiner, and Sepp Hochreiter (2015). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). doi: 10.48550/ARXIV.1511.07289. url: https://arxiv.org/abs/1511.07289.

David, Etienne et al. (2020). “Global Wheat Head Detection (GWHD) dataset: a large and diverse dataset of high-resolution RGB-labelled images to develop and benchmark wheat head detection methods”. In: Plant Phenomics 2020.

Ding, Bin, Huimin Qian, and Jun Zhou (2018). “Activation functions and their characteristics in deep neural networks”. In: 2018 Chinese control and decision conference (CCDC). IEEE, pp. 1836–1841.

Hayou, Soufiane, Arnaud Doucet, and Judith Rousseau (2019). “On the impact of the activation function on deep neural networks training”. In: International conference on machine learning. PMLR, pp. 2672–2680.

He, Kaiming et al. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. doi: 10.48550/ARXIV.1502.01852. url: https://arxiv.org/abs/1502.01852.

Hou, Borui and Ruqiang Yan (2020). “Convolutional Autoencoder Model for

Finger-Vein Verification”. In: IEEE Transactions on Instrumentation and Measurement 69.5, pp. 2067–2074. doi: 10.1109/TIM.2019.2921135.

KILIC¸ARSLAN, Serhat, ADEM Kemal, and Mete C¸elik (2021). “An overview of the activation functions used in deep learning algorithms”. In: Journal of New Results in Science 10.3, pp. 75–88.

Kim, Taewook and Ha Young Kim (2019). “Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data”. In: PloS one 14.2, e0212320.

Klambauer, G¨unter et al. (2017). “Self-Normalizing Neural Networks”. In: Advances in Neural Information Processing Systems. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. url: https://proceedings.neurips.cc/paper/2017/file/5d44ee6f2c3f71b73125876103c8f6c4-Paper.pdf.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: Advances in neural information processing systems 25.

Kukreja, Harsh et al. (2016). “An introduction to artificial neural network”. In: Int J Adv Res Innov Ideas Educ 1, pp. 27–30.

Li, Ding, Yuefei Zhu, and Wei Lin (2017). “Traffic identification of mobile apps based on variational autoencoder network”. In: 2017 13th International conference on computational intelligence and security (CIS). IEEE, pp. 287–291.

Li, Guanqing, Zhiyong Song, and Qiang Fu (2018). “A New Method of Image Detection for Small Datasets under the Framework of YOLO Network”. In: 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 1031–1035. doi: 10.1109/IAEAC.2018.8577214.

Maind, Sonali B, Priyanka Wankar, et al. (2014). “Research paper on basic of artificial neural network”. In: International Journal on Recent and Innovation Trends in Computing and Communication 2.1, pp. 96–100.

Mijwel, Maad M (2018). “Artificial neural networks advantages and disadvantages”. In: Retrieved from LinkedIn https//www. linkedin. com/pulse/artificial-neuralnet Work.

Nguyen, Anh et al. (2021). “An analysis of state-of-the-art activation functions for supervised deep neural network”. In: 2021 International Conference on System Science and Engineering (ICSSE). IEEE, pp. 215–220.

Nicolson, Aaron and Kuldip K Paliwal (2019). “Deep learning for minimum mean-square error approaches to speech enhancement”. In: Speech Communication 111, pp. 44–55.

Raghavendra, U et al. (2019). “A two layer sparse autoencoder for glaucoma identification with fundus images”. In: Journal of medical systems 43.9, pp. 1–9.

Theckedath, Dhananjay and RR Sedamkar (2020). “Detecting affect states using VGG16, ResNet50 and SE-ResNet50 networks”. In: SN Computer Science 1.2, pp. 1–7.

Wang, Zi et al. (Jan. 2022). “Interval universal approximation for neural networks”. In: Proceedings of the ACM on Programming Languages 6.POPL, pp. 1–29. doi: 10 . 1145 / 3498675. url: https : / / doi . org / 10 . 1145 % 2F3498675.

Yu, Jiabao et al. (2019). “Radio frequency fingerprint identification based on denoising autoencoders”. In: 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). IEEE, pp. 1–6.