© Universiti Tun Hussein Onn Malaysia Publisher's Office

JSCDM

Journal homepage: http://penerbit.uthm.edu.my/ojs/index.php/jscdm

Journal of Soft Computing and Data Mining

A Comprehensive Survey of Deep Learning Models Based on Keras Framework

Bahzad Taha Chicho[1*], Amira Bibo Sallow[2]

[1]IT Department, Technical College of Informatics Akre,
Duhok Polytechnic University, Duhok, Kurdistan Region, IRAQ

[2]Technical College of Administration,
Duhok Polytechnic University, Duhok, Kurdistan Region, IRAQ

*Corresponding Author

DOI: https://doi.org/10.30880/jscdm.2021.02.02.005
Received 09 May 2021; Accepted 07 July 2021; Available online 15 October 2021

Abstract: Python is one of the most widely adopted programming languages, having replaced a number of those in the field. Python is popular with developers for a variety of reasons, one of which is because it has an incredibly diverse collection of libraries that users can run. The most compelling reasons for adopting Keras come from its guiding principles, particularly those related to usability. Aside from the simplicity of learning and model construction, Keras has a wide variety of production deployment options and robust support for multiple GPUs and distributed training. A strong and easy-to-use free, open-source Python library is the most important tool for developing and evaluating deep learning models. The aim of this paper is to provide the most current survey of Keras in different aspects, which is a Python-based deep learning Application Programming Interface (API) that runs on top of the machine learning framework, TensorFlow. The mentioned library is used in conjunction with TensorFlow, PyTorch, CODEEPNEATM, and Pygame to allow integration of deep learning models such as cardiovascular disease diagnostics, graph neural networks, identifying health issues, COVID-19 recognition, skin tumors, image detection, and so on, in the applied area. Furthermore, the author used Keras's details, goals, challenges, significant outcomes, and the findings obtained using this method.

Keywords: Deep Learning (DL), Artificial Neural Network (ANN), Python, Keras, TensorFlow

1. Introduction

The emergence of machine learning and deep learning in recent years has resulted in popular implementation in a number of fields. A new generation of applications has resulted from significant advances in the scale of training sets and usable processing resources. As a result of this progress, the application and propagation of deep learning have expanded [1]. Several programming languages present tools and libraries for filling up Artificial Neural Networks. However, Python has appeared as the strong frontrunner in this domain [2]. Python is a basic and straightforward computational language that includes many science libraries such as NumPy8, SciPy9, Matplotlib10, Keras, and others that carry out specific simulations, regression analysis, and the determination of ordinary differential equations [3].

A Python library is a reusable piece of programming that you can use in your programs or projects. In contrast to languages such as C++ or C, Python libraries are not context-specific. In addition, the term 'library' refers to a set of key modules. A library is, in essence, a set of modules. A module is a library that can be built with the help of a package manager. Python libraries are important in developing machine learning, data science, data visualization, image and object editing applications, and other applications. There are many libraries in Python, but for this paper, the

Corresponding author: bahzad.taha@dpu.edu.krd authors only relied on the Keras library. Keras is by far the most popular deep learning library among practitioners [4]. Keras is a TensorFlow-based Application Programming Interface (API) that allows users to introduce easily, train, and analyze neural networks [5]. Its convenience encapsulates a lot of the low-level difficulty that comes with constructing deep networks from the ground up. Keras abstracts many of TensorFlow's more complex features while also allowing for customizability and ease of use [6]. Keras is many people's first choice for deep learning apps because of this mix. Keras is the obvious alternative at one end of the two-way bridge because of its prominence and ease of use [7]. This study aims to provide an overall review of Keras based on several factors (see Table 3), and the survey is completed by reviewing the latest research in various areas.

In this paper, researchers conducted a comprehensive review of the latest and most efficient methods of the Keras library in various fields over the last three years, and the details of this method, such as objectives, technique/tool, datasets, issues, important findings, and accuracy, are summarized. Moreover, the study highlighted the most commonly used methods.

The rest of this article is structured as follows. In Section 2, researchers describe the background theory that contains an overview of the Kiras library. In Section 3, a literature review of many aspects of Keras recent research studies is included. Section 4 contains a comparison and discussion of Keras, and Section 5 contains our concluding remarks.

2. Keras

Keras is a high-level neural network API written in Python that can run on top of TensorFlow and other lower-level frameworks. It was designed with a focus on allowing quick experimentation and enabling simple and fast prototyping (through user-friendliness, modularity, and extensibility) (through user-friendliness, modularity, and extensibility) [8]. Keras accepts various forms of neural network components, such as thick layers, convolutional layers, recurrent layers, dropout layers, and supports variations of them. The code dynamically handles tools such as the Central Processing Unit (CPU) and the Graphics Processing Unit (GPU), allowing optimal use of them. It also has implementations of activation functions 8, optimizers 9, metric formulas 10, and procedures required to handle training sessions with ease [9]. A Deep Learning plugin integrates functionality from Keras libraries, which in turn integrates functions from TensorFlow into Python.

2.1 Deep Learning with Keras

Keras is a deep learning library that allows one to create and train models quickly. Layers in the library are bound to one another like Lego blocks, resulting in a tidy and easy-to-understand model. Model testing is simple, needing just details, a number of training epochs, and metrics to track [10]. As a consequence, most deep learning models can be applied with considerably fewer lines of code. By utilizing Keras, we can increase efficiency by spending time on technology execution, which can then be spent on more important tasks, including developing improved deep learning algorithms. The Sequential Model API helps you to build typical models with a few lines of code. But don't be tricked by its beauty. Keras can also generate more sophisticated and complicated models by using its API and Model and Layer classes, which can be tailored to suit particular needs. The Functional API helps you to construct graph-based structures, reuse layers, and create models that act like Python functions [6]. Meanwhile, the Concept and Layer classes act as a basis for integrating novel or experimental deep learning frameworks and layers. Keras is the product of one of these recent advances that allow neural network models to be defined and created with a few lines of code. Fig. 1 shows Deep Learning (DL) in Keras - Building a DL Model.

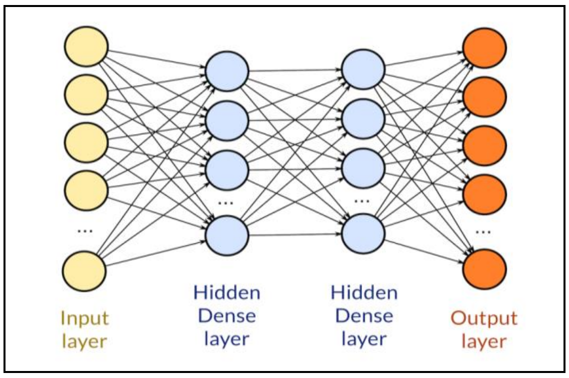


Fig. 1 - DL in Keras - Building a DL Model [11]

Keras is not a stand-alone deep learning library. It is installed on top of another deep learning library or backend, as seen in Fig. 2. This may be TensorFlow from Google, Theano from MILA, or CNTK from Microsoft. Keras is operated by a CPU, GPU, and Google's TPU [12].

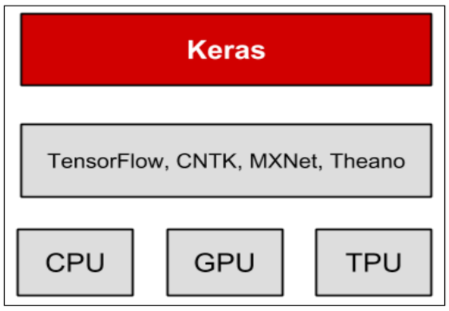


Fig. 2 - Keras is an advanced repository that lies on top of other deep learning models [13]

The following are the primary advantages of utilizing Keras over other deep learning frameworks: The Keras interface is user-friendly and well-optimized for general use cases. Its basic API structure is intended for both new developers and experts. Custom blocks for expansion can be written in Keras. Keras is the second most common deep learning system after TensorFlow. Tensorflow's (tf. Keras) module also facilitates Keras implementation. Tensorflow provides links to all Keras features (tf.KERAS). Several algorithms of DL that are used with Keras are Auto-Encoders, Convolution Neural Nets, Recurrent Neural Nets, Long Short-Term Memory Nets, Deep Boltzmann Machine (DBM), Deep Belief Nets (DBN). Fig. 3 shows the use of automatic encoders to reduce noise with Keras, TensorFlow, and Deep Learning.

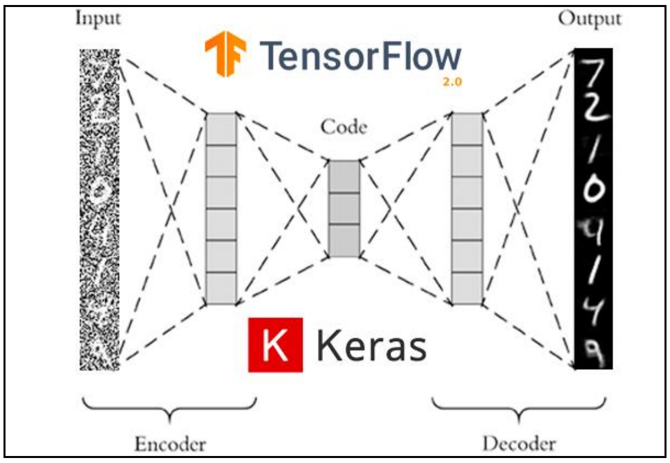


Fig. 3 - Keras, TensorFlow and DL automatic encoders to remove noise from images [14]

2.2 Reasons to Learn and Use Keras

There are multiple reasons for learning and using Keras. Some of the reasons are as follows.

- Keras enables us to move between backends based on the needs of our applications. It functions as a shell allowing one to use TensorFlow, Theano, or other structures [12].
- Keras is extremely easy to use and enjoyable to use. It adheres to excellent guiding standards such as extensibility, Python nativeness, and modularity.
- Keras' ability to build cutting-edge applications of common deep neural networks. It is short and simple to set up with Keras [15].
- Keras users will be faster and more productive and able to experiment with more ideas.
- Keras offers multi-GPU and robust distributed support. DL models can be run on large GPU clusters.
- Keras DL frameworks can be deployed on a range of systems.
- Keras has a wide ecosystem of products to help grow DL. Tensorflow Cloud, Keras Tuner, Tensorflow Lite, Tensorflow.js, and Tensorflow Model Optimization are some of the more common products.

2.2 Python Keras Advantages and Limitations

Keras is a great starting point for those who want to learn about neural networks. It is a high-level structure that masks the backend computation and enables one to build a neural network model easily. Table 1 shows the main advantages and limitations of Keras [16].

Table 1 - Advantages and Limitations of Keras

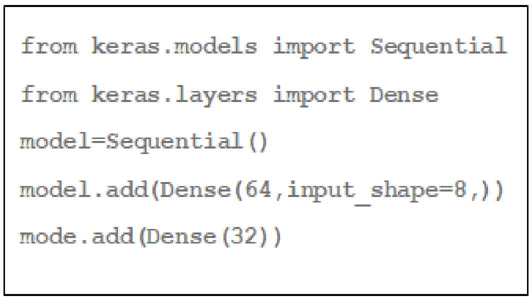
Advantages	Limitations
1. Easy and fast implementation.	1. Issues with a low-level API
2. High-quality documents and extensive mutual support.	2. Some features require improvement.
3. Modularity and several backends.	3. It is slower than its backend.
4. Pre-trained models.	
5. Help with several GPUs.	

2.3 Types of Keras Models

Keras provides two modes for building models: a simple and easy-to-use Sequential API and a more complex and sophisticated Functional API:

2.3.1 Sequential Model in Keras

The Sequential Model enables the development of models layer by layer in a sequential sequence. However, it does not permit the development of models of several inputs or outputs. It fits well with a basic stack of layers with one input and one output tensor. When any of the layers in the stack has several inputs or outputs, this model is ineffective. It is not ideal even though the authors choose a non-linear topology [17]. An example of a sequential model is as in Fig. 4.

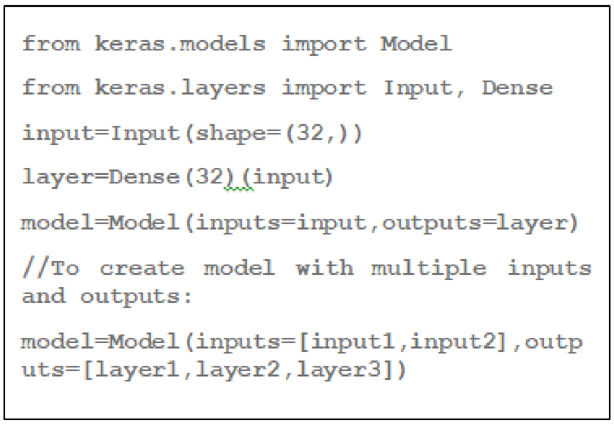


```
from keras.models import Sequential
from keras.layers import Dense
model=Sequential()
model.add(Dense(64,input_shape=8,))
mode.add(Dense(32))
```

Fig. 4 - Example of a sequential model code

2.3.2 Functional API in Keras

In Keras, the functional API provides more options for defining a model and adding layers. The practical API may be used to build templates for different inputs and outputs. It also allows for the exchange of these layers. In other terms, we can build layer graphs with Keras' functional API. Since a functioning API is a data structure, it is simple to save it as a single file that can be used to replicate the same model without the original code. It is also easy to model the graph and view its nodes here [17]. Fig, 5 shows an example of a functional API:



```
from keras.models import Model

from keras.layers import Input, Dense

input=Input(shape=(32,))

layer=Dense(32)(input)

model=Model(inputs=input,outputs=layer)

//To create model with multiple inputs
and outputs:

model=Model(inputs=[input1,input2],outp
uts=[layer1,layer2,layer3])
```

Fig. 5 - Example of functional API in Keras

2.4 Difference Between Keras and Tensorflow

Keras and Tensorflow are two well-known DL systems. DL professionals more often use Keras and Tensorflow. All of these systems have widespread mutual interests. Both of these frameworks capture a large portion of the DL output. There are several distinctions between Keras and Tensorflow [18], as shown in Table 2.

Table 2 - Difference between Keras and Tensorflow

Keras	TensorFlow
1. An API built on TensorFlow, CNTK, and Theano.	1. A system that provides both high-level and low-level APIs.
2. It is ideal for fast deployments.	2. Complex networks are ideal for DL analysis.
3. It began as a project by François Chollet and was created by a community of people.	3. It was generated by Google's brain squad.
4. Typically written in Python, it is a wrapper for Theano, TensorFlow, and CNTK.	4. The majority of the code was written in C++, CUDA, and Python.
5. For dealing with tiny datasets, this method is used.	5. High-performance models and wide datasets are typically utilized.
6. There is no community outreach.	6. It has the support of a wide group of technology companies.

3. Literature Review

Keras is a minimalist Python deep learning library that can be used for Theano or TensorFlow. It was designed to allow integrating deep learning models for research and development as quickly and simply as possible. The researchers address some recent work regarding the Keras library relevant to data mining in this portion of the review. Table 3 describes the different forms of literature reviews on Keras's approaches.

Deep Learning (DL) expands conventional Machine Learning (ML) by adding more "depth" (intricacy) to the system and transforming the data with different functions that hierarchically enable data description across multiple layers of abstraction [19], [20]. Feature learning, or the automated extraction of features from raw data, is a strong benefit of DL, with features from higher hierarchy levels being created by the configuration of lower-level features. Because of the more complicated systems used, which allow huge parallelization, DL can handle many complex issues, especially well and quickly [21], [22]. Based on the network structure utilized, DL may contain a number of elements (for example, convolutions, pooling layers, fully connected layers, gates, memory cells, activation functions, encode/decode schemes, and so on) (i.e. Unsupervised Pre-trained Networks, Convolutional Neural Networks, Recurrent Neural Networks, Recursive Neural Networks) [23], [24]. DL is a possible tool for large-scale and deep artificial neural networks.

Artificial Neural Networks (ANNs) are computational paradigms focused on mathematical models that, unlike conventional computing, have a configuration and function similar to that of a mammalian brain. Since they are made up of a collection of interconnected computing components that run in parallel, artificial neural networks, or neural networks for short, are also known as connectionist systems, parallel distributed systems, or adaptive systems [25]. Since all of the interconnected computing components change or "adapt" concurrently with the flow of information and adaptive laws, neural networks lack unified control in the conventional context [26]. ANN was developed to recognize and influence the functional features and mathematical characteristics of the brain as it executes cognitive functions, including sensorial interpretation, object categorization, concept association, and learning. Today, though, a lot of research is put into designing neural networks for applications like pattern recognition and classification, data compression, and optimization [27]. Python is one of ANN's most important programs. The Keras library's goal is to offer the building blocks for creating graph neural networks, stress usability standards, and fast prototyping, and make it easy to tackle graph-related issues using this library.

Aggarwal et al. [28] used Keras in deep learning to classify plants, which are required to differentiate a large number of flowers from various groups. Several researchers used different approaches to classify the plant genus. This is a commonly utilized tool for automatically removing features, analyzing them, and generating the best output. From among the Indian plants, five separate plant species with over 500 epochs were chosen as specimens: OcimumTenuiflorum, Sansevieriatrifasciata, Chlorophy tumcomosum, Azadirachtaindica, and Aloe Vera. These samples have a lot of oxygen. The shape, color, texture, and corners are derived from these samples. One hot is also added to the target values to improve identification results. The extracted features are fed into the sequential Keras model, which identifies plant organisms. The results showed that the accuracy of the training set was 100%, while the accuracy of the study set was 96.7%.

Haghighat and Juanes [29] presented Scientific Computing with Artificial Neural Networks (SciANN), a Python package for scientific computing and physics informed by deep learning using artificial neural networks. SciANN builds deep neural networks and optimization models using the commonly used deep learning packages Tensorflow and Keras, inheriting certain Keras features such as batch optimization and model reuse for transfer learning. SciANN is intended to abstract from neural network development for mathematical computations and the solution and discovery of Partial Differential Equations (PDE) using the Physics-Informed Neural Networks (PINN) architecture, allowing for the development of complex functional models. The results demonstrated how to use SciANN in the sense of physics-informed deep learning for curve-fitting, solving PDEs in strong and weak forms, and model inversion. Both of the explanations given here, as well as the kit itself, are open-source and publicly accessible.

Gupta et al. [30] presented a modern forecast paradigm dubbed traditional and cutting-edge technology focused on increased precision in forecasting cardiovascular disease. The data obtained from Stanford's online healthcare repository containing 304 records with 10 attributes was exposed to new-fangled techniques such as TensorFlow, PyTorch, and KERAS on the same dataset obtained from Stanford's online repository. In addition to the entire set of ML techniques, the empirical findings indicate that KERAS achieved an extraordinary prediction accuracy of 80%.

Bohrer et al. [9] proposed an open implementation of the Coevolution Deep NEAT (CoDeepNEAT) algorithm using the commonly used and well-supported Keras platform. CoDeepNEAT is a versatile neural network topology generation technique focused on augmenting topology neuroevolution (NEAT) and module co-evolution. It uses evolutionary techniques and heuristics to investigate the vast search space of potential topological configurations for neural networks, using advanced genetic algorithm aspects to produce and test topology and hyperparameter selection solutions. It was then checked on publicly available image datasets and compared to the original version's performance, taking into account discrepancies in conditions and experimentation parameters. The findings illustrate that suitable network topologies can be accomplished with small population sizes and few generations operating in constrained hardware settings, despite the fact that massive runs and populations provide the best results.

Versaci [31] presented an efficient wavelet library that utilizes TensorFlow and Keras to exploit Graphics Processing Unit (GPU) parallelism and allows for simple integration into existing machine learning workflows, called Wavelet TensorFlow (WaveTF). WaveTF is checked for performance on image datasets by hardware and software in two ways: first, by performing raw signal transformations and storing the output data in Random Access Memory (RAM) or GPU memory. Second, as a Keras layer, it was implemented into a basic neural network for training. Since the wavelet transform is distinguished by high parallelism and low computational complexity, minimizing communication is crucial to achieving successful output. The results show that the overhead of applying wavelet features to current 5-level CNNs is less than 1% in both training and assessment, making them useful at a low cost.

Yang et al. [32] introduced Composed Neural Networks (CpNNs) and our compiler CpNN2C, which both prepare a pathway for the simultaneous design of the component neural networks of real-time systems based on the well-known Keras neural network repository. The methods also allow modular translation from a specified CpNN to C code using the established semantics. The created code can be used for Worst-Case Execution Time (WCET) analysis. Vehicle tracking data (actual human driver data) collected from human drivers was used to train neural networks. Like a platoon of cars, the nine vehicles follow a circular route. Every vehicle's spacing, velocity, and acceleration are measured with a sampling precision of 0.1 sec. So many benchmarks revealed the established methodology's supremacy over a recent method utilizing Esterel, besides the common Python package TensorFlow Lite. According to the results, the CpNNs solution outperforms Esterel with an average WCET lowering of 64.06% and TensorFlow Lite with an average evaluated WCET lowering of 62.08%.

Parisi et al. [33] prepared the 'hyper-sinh,' a variant of the m-arcsinh activation function appropriate for DL-based supervised learning techniques like CNN. Hyper-sinh is thus described and verified as an efficient and consistent activation mechanism for both deep and shallow neural networks, using the open-source Python libraries TensorFlow and Keras. Enhancements in precision and usability in image and text classification functions are presented using five (N = 5) sample data sets accessible from Keras. The experimental findings show that this novel feature improves comprehensive competitive classification performance of shallow and deep neural networks. This feature is comparable to gold standard triggering mechanisms, demonstrating its comprehensive comparability in image and text recognition accuracy and reliability.

Grattarola and Alippi [34] introduced Spektral, an open-source Python library for creating graph neural networks with TensorFlow and the Keras implementation programming interface. Spektral integrates a diverse range of deep learning techniques into graphs, such as message-passing and pooling processes, besides tools for processing graphs and loading common gauge datasets. The aim of this library is to prepare the necessary construction blocks for constructing graph neural networks, with a focus on the ease of use and rapid prototyping of concepts as guidelines that Keras is built on. As a result, Spectral is appropriate for both absolute beginners and experienced deep learning practitioners. The findings show that Spektral's strong numerical efficiency and range of available methods make it a feasible alternative even for specialized use cases.

Reiser et al. [35] established the Convolution and pooling layers for the TensorFlow-Keras technique, allowing for smooth and scalable incorporation into standard Keras layers to set up graph models in a practical way. This necessitates the use of mini-batches as the first tensor dimension, which is possible with TensorFlow's new RaggedTensor group, which is better suited for graphs (image processing). Centered on TensorFlow-Keras, the researchers built the Keras Grid Convolutional Neural Network (KGCNN) Python bundle, which includes a collection of Keras layers for graph connections. The result showed that the graph neural networks were implemented, and the clarity of the tensor representation and smooth incorporation with other Keras systems are the key goals of the KGCNN bundle.

Balan et al. [36] developed a novel approach to combining Ecology and Engineering using machine learning algorithms. They used the traditional allele frequency database "ALFRED" to demonstrate the efficiency of the built model. Also, the adaptive learning rate enhancement was used by the researchers to simplify the Hardy Weinberg prototype. For the study of growth, the neural network that is employed for training was created using Tensorflow and Keras. During the optimization process, it was discovered that the two-layer network has oscillatory behavior, whereas the four-layer network has less oscillatory behavior. As a result, automating this crucial theory of HWE in Population Genetics facilitates the analysis of allele frequencies across generations with less difficulty.

Yashwanth et al. [37] presented an innovative approach to minimizing herbicide use by classifying plant images into weeds and crops to select herbicide spraying. Excessive use of weedicides can endanger the health of agricultural workers and growers and contaminate the soil and water. The first step is to recognize the difference between crops and weeds. The Deep Learning function was used to apply the Image Classification Method. The result indicated that the maximum productivity of 96.3% was obtained with just 250 images of every plant in the features. The developed framework can be conveniently installed on the Raspberry Pi, and specific spraying can be accomplished with a linked sprayer. For real-time deployment, this system may be mounted on a tractor or a helicopter.

Manapure et al. [38] modified and adapted the current Artificial Intelligence (AI) technique and joined it with clinical knowledge to construct a framework that uses stable 2-dimensional and 3-dimensional Deep Learning (DL) models, Keras, and TensorFlow. The candidate disease areas were first divided into parts of the pulmonary Computed Tomography (CT) image collection. It includes employing a 3-dimensional DL system and utilizing a location-attention classification method. These segregated images were then grouped into COVID-19, Influenza-A viral pneumonia, and unrelated to infection types, along with acceptable reliability ratings. Finally, the Noisy-or Bayesian method determined the disease form and overall confidence rate of this CT case. The COVID-19 processors achieved 90-92% on a dataset based exclusively on X-ray images; no other data, such as geographical area, population density, or other factors, has been used to train this method.

Benbrahim et al. [39] presented a system for classifying seven forms of skin cancer utilizing the Deep Convolution Neural Network (CNN), TensorFlow, and Keras structure. The system was put into action by using an image classification scheme on the HAM10000 database. Understanding that there are seven potential disease types, the above includes a large number of skin cancer images in the Jpeg file. The number of cases of skin cancer is rising, and it is getting more dangerous as a result of a pause in diagnosis or an inability to predict the disease. The aim is to create a system that can recognize skin tumors in photographs and identify them. This technology will assist doctors in properly diagnosing, detecting, and easily identifying skin cancer patients and speed up workflow. The classification experiment's findings indicate that the model's precision was accomplished, with 94.06% and 93.93% in the validity and test sets, respectively, in its optimal configuration.

Harjoseputro [40] introduced a Javanese character classification system utilizing the Convolutional Neural Network (CNN) approach and KERAS for Image Classification, as deep learning techniques have recently become one of the most common approaches for working with image classification issues. The modeling approach and dataset included in this study are simple, resulting in a reduction in computation weight and time. The results showed that the method had the highest precision and time of 86.68% and 639.85 seconds, employing 1000 datasets and 50 eras. Furthermore, the benefit of this process is that it is quick to compute and has a high level of accuracy, providing a mechanism for further development, especially when applied to a device that recognizes Javanese characters on a mobile or web platform.

Kumar et al. [41] presented a real-time identification of handwritten digits approach for handwritten digit recognition. The image is classified using CNN as the model, and the classifier is the Keras Sequential model. Pygame is in charge of the interface. The interface's architecture is kept plain, with two frames for input and output. The most significant move, which was accomplished with the aid of OpenCV and Scipy, was image pre-processing. The teaching and research dataset is MNIST. Handwritten Digit Identification can be found in a variety of circumstances in real life. It is used to detect car numbers, read checks in accounts, sort letters in post offices, and perform a host of other activities. According to the findings, the consistency of the test results was tested for various CNN layers, including two-layered, three-layered, and four-layered CNNs. Of all the layers compared, the four-layered CNN produced the best accuracy of 99.25% with the least failure. In the case of 15 eras, this was the case.

Nagisetty and Gupta [42] offered a system for detecting malicious behaviors in IoT Backbone Networks by using the Keras Deep Learning Library. The primary goal is to detect malicious network traffic in the IoT backbone network utilizing quick and reliable big data techniques such as TensorFlow and Keras. To distinguish network traffic, they used four separate deep learning techniques: Multi-Layer Perceptron (MLP), CNN, Deep Neural networks (DNN), and Autoencoder. To distinguish attackers, two well-known datasets, UNSW-NB15 and NSLKDD99, are used for research and output comparison of an identification of malicious behaviors using machine learning algorithms such as Logistic Regression (LR) and Vector Machine Support (SVM). The experimental findings show that the DNN outperforms all other systems in terms of precision (99.24%).

Mondal et al. [43] presented a new deep learning technique for the prediction of chest diseases based on X-ray images using Xception deep convolutional neural network structure. They used the Keras library to run the experiment and recognize the chest disease images, which is installed on top of the Tensorflow backend. Patients who do not have access to radiologists to interpret their chest X-rays will profit from the classification of precision. According to the findings, the produced model had an accuracy of 88.76%, and the model failure was also at a level that was satisfactory in terms of performance. Furthermore, our model had higher AUC scores for most illnesses, with hernia having the maximum AUC score of 0.83.

Refianti et al. [44] presented a model for Melanoma Skin Cancer Classification Utilizing Convolutional Neural Networks and. The classification of melanoma cancer images is divided into two stages: the first is the training of the dataset to create a standard, and the second is the classification of the images. The second stage is the classification method, in which the machine analyzes the image details. In addition, the dataset contained 220 images of dermoscopy examination and was obtained from the ISIC (International Skin Imaging Collaboration) website. For sorting, the Python programming language was used and the Keras library as a Tensorflow backend. The method of classifying the outcomes of the dermatoscopy analysis in this device, on the other hand, was more precise and the deduction period was comparatively short. This application classifies image data using deep learning technology, including the Convolutional Neural Network method and a multi-layer network based on CNN, as introduced by the Yann LeCun (LeNet-5) architecture. The results found that the maximum performance percentage was 93% in preparation and 100% in research, with 176 photographs and 100 epochs of training data used.

Tseng and Lee [45] provided a supervised learning method for designing a Digital Differentiator (DD) on the Keras framework. To begin, the DD design issue is transformed into a supervised learning problem, allowing Keras' rich learning resources to be applied directly to the problem. The coefficients of the DD transfer function are then calculated by minimizing loss functions, including mean squared or mean absolute errors. Following that, numerous Keras optimizers such as Stochastic Gradient Descent (SGD) or Adaptive Moments (Adam) algorithms are used to find the best answer to the DD architecture challenge. Finally, a number of numerical illustrations are provided to demonstrate the efficacy of the proposed design process. The results showed that the DD approach produces fewer approximation errors than the traditional method.

Faker and Dogdu [46] incorporated Big Data and Deep Learning mechanisms to improve the performance of intrusion detection systems. The Deep Feed-Forward Neural Network (DNN), as well as two ensemble methods, RF and Gradient Boosting Tree, are used to identify network traffic datasets (GBT). They used a homogeneity index to determine characteristics in order to pick the most important attributes from the datasets. The suggested approach was evaluated using two recently released databases, UNSW-NB15, and CICIDS2017. The machine learning models were evaluated using 5-fold cross-validation in this study. They introduced the approach using the distributed computing framework Apache Spark, combined with Keras Deep Learning Library to apply the deep learning methodology while the ensemble techniques are implemented using Apache Spark Machine Learning Library. The findings revealed that DNN had good precision for binary and multiclass classification on the UNSW-NB15 dataset, with accuracies of 99.16% for binary classification and 97.01% for multiclass classification. While the GBT classifier obtained the maximum accuracy for binary classification with the CICIDS2017 dataset at 99.99%, DNN had the highest accuracy at 99.56% for multiclass classification.

4. Comparison And Discussion

Many studies have been conducted in the past on deep learning libraries using various methods and their future implications. Many researchers expect that the usage of such libraries will aid programming processes significantly. It can be concluded from the preceding section that researchers have worked in various fields with various tools and algorithms. The researchers illustrated relevant aspects of the suggested methods of assessment.

A comparison of the studies explained at the end of this section is seen in Table 3. The comparison includes six main features that adopt their trends in order to verify the goals achieved via their approaches in the deep learning field, which is based on Keras. The table shows that the reference [30] used Keras with Tensorflow on the Stanford online healthcare repository dataset to solve diagnostic of cardiovascular disease issues, and the empirical findings indicate that KERAS obtained an extraordinary prediction precision of 80%. In contrast, [35] utilized Keras and TensorFlow on image processing datasets to overcome the flexible size of graphs. Moreover, based on the study [33] employed Keras and Tensorflow on accuracy and reliable function issues, the highest accuracy was achieved at 99% when using the 'MNIST' data set available.

On the other hand, the reference [37] used Keras on plant images to overcome the health problems, and maximum efficiency of 96.3% was achieved with just 250 images. The author [39] utilized the DCNN, TensorFlow, and Keras structures on cancer images to address skin tumors, while the study [38] employed the DL, Keras, and TensorFlow on COVID-19 identification problems, and the processors achieved 90-92% on a dataset based exclusively on X-ray images.

According to the studies reviewed, the Keras library in Python has advantages and drawbacks like all other libraries. The most important advantages are user-friendly and fast deployment, quality Documentation and large community support, multiple backend and modularity, pre-trained models, and multiple graphics processing units support. While its shortcomings are due to issues with the low-level Application Programming Interface (API), some features require improvement, and it is slower than its backend.

In general, based on the literature review, the authors mostly used the Keras library with TensorFlow, appearing as if the Keras library does nothing without TensorFlow. Furthermore, a survey in this study found that Keras is the most commonly used in the field of Neural Networks. However, it is also used to solve problems in other fields, such as plants, healthcare, neural networks, and agriculture. Finally, in this section, the used mechanisms are summarized in Table 3 as shown below, including the aims, techniques/tools, datasets, issues, significant results, and accuracy.

Table 3 - Summary of Literature Review Related to Keras

Ref	Year	Objectives	Problems	Dataset(s)	Techniq ue /Tool	Significant Results	Accuracy
[28]	2021	Plant Detection Sequential Model Optimization	Plant Recognition	images plant (more than 500 epochs)	KERAS	The instruction set is 100% accurate, while the research set is 96.7% accurate	instruction set: 100% research set: 96.7%
[30]	2021	improved prediction accuracy and dependability of cardiovascular disease diagnostic performance	diagnostic of cardiovascul ar disease	Stanford's online healthcare repository contained a sum of 304 records with 10 attributes	KERAS, and TensorFl ow, PyTorch	the empirical findings indicate that KERAS obtained an extraordinary prediction precision of 80%	Keras: 80%
[31]	2021	evaluate the computing overhead of introducing wavelet functionality to the network using WaveTF	the overhead of integrating WaveTF	image	Keras and TensorFl ow	The complexity of applying wavelet capabilities to the original 5-level CNN is less than 1% in both testing and assessment, making it practically free to use	reduce the complexity to less than 1%
[34]	2021	to prepare the necessary building blocks for constructing graph neural networks	Graph neural networks	Graph such as message-passing and pooling process	Keras and TensorFl ow	Spektral is a strong option also for specialized usage cases because of its good numerical efficiency and range of usable methodologies. The best accuracy is achieved when classifying node	Keras: 77%

						classification tasks, reaching 77%.	
[35]	2021	TensorFlow-Keras Technology Incorporated	flexible size of graphs	Graph (image processing)	Keras and TensorFlow	the KGCNN package is the transparency of the tensor representation and the seamless integration with other Keras models, as well as the mean absolute validation error is reduced to 0.148	mean absolute validation error is reduced to 0.148
[9]	2020	To check the complexity of the process	Neural development of neural network architects	images	Keras and CODEEPNEAT	Suitable network topologies can be accomplished with small population sizes and few generations operating in constrained hardware settings, and the network obtained a training accuracy of 86.5% and a validation accuracy of 79.5%.	Training: 86.5% validation: 79.5%.
[32]	2020	Reduce Real-time systems	Strict functional and timing constraints	Vehicle tracking data (actual human driver data)	CpNNs based on Keras	the CpNNs method outperforms Esterel with a 64.06% WCET lowering and Tensorflow Lite with a 62.08% gauged WCET lowering	CpNNs: 64.06% Tensorflow: 62.08%
[33]	2020	to develop the m-arcsinh into a new precise and dependable activation mechanism	Accuracy and Reliable Function	image and text	Keras and Tensorflow	image and text classification with robust competitive accuracy and dependability and the highest accuracy is achieved at 99% when using the 'MNIST' data set available in Keras	99%
[36]	2020	Hardy Weinberg Equilibrium Automated Evolution	Allele difference	allele frequencies "ALFRED"	Keras and Tensorflow	Automating this crucial theory of HWE in Population Genetics facilitates the analysis of allele frequencies across generations with less difficulty.	After 40 repetitions, it is apparent that the loss function has been reduced
[37]	2020	Avoid using herbicides manually	health problems	plant images	Keras	the maximum efficiency of 96.3% was achieved with just 250 images	Keras: 96.3%
[38]	2020	Build a DL-based method for automated accident zone split and quantification.	COVID-19 identification	X-ray images	DL, Keras, and TensorFlow	The COVID-19 processors achieved 90-92 % on a dataset based exclusively on X-ray images	90-92 %
[39]	2020	create a system that can recognize skin tumors in photographs	skin tumors	cancer images HAM10000	DCNN, TensorFlow and Keras	The model's precision was accomplished, with 94.06 % and 93.93 % in the validity and test sets, respectively,	validation set: 94.06%

		and identify them			structure	in its optimal configuration.	test set: 93.93%
[40]	2020	Javanese Letters Grouping	image classification	Image	KERAS and CNN	the highest precision and time of 86.68% and 639.85 seconds employing 1000 datasets and 50 eras	86.68%
[41]	2020	handwritten digit recognition	handwritten digit	Image (handwritten digit)	Keras Sequenti al and pygame	the four-layered CNN produced the best accuracy of 99.25 %, with the least failure	99.25 %
[42]	2019	identify the malicious network traffic in the IoT backbone network	secure and reliable	UNSW-NB15 and NSLKDD99 (text)	Keras and TensorFl ow.	the DNN performs well among all the schemes in terms of accuracy (99.24%)	99.24%
[43]	2019	Automatic Classification	Classificatio n	X-ray images	Keras and TensorFl ow.	The produced model had an accuracy of 88.76%,	88.76%
[44]	2019	to develop a method that uses images to identify melanoma cancer	Classificatio n of Melanoma Skin Cancer	Image	Keras and TensorFl ow.	the maximum percentage of performance was 93%in preparation and 100%	93%
[46]	2019	enhance the effectiveness of intrusion prevention devices	intrusion prevention	UNSW-NB15 and CICIDS2017 (text)	Keras and g Apache Spark	the GBT classifier obtained the maximum accuracy for binary classification with the CICIDS2017 dataset at 99.99%. For multiclass classification, DNN had the highest accuracy at 99.56%.	GBT: 99.99% DNN: 99.56%.

5. Conclusion

Keras is a powerful and easy-to-use free, open-source Python library for developing and evaluating deep learning models. It wraps the powerful numerical computing libraries Theano and TensorFlow, allowing you to describe and train neural network models in a matter of lines of code. This analysis aimed to assist the reader in comprehending the numerous aspects presented by the study in the Keras library, which is focused on deep learning. According to the study reviewed, the most significant difficulties confronting Keras are the Low-level Application Programming Interface, which has certain features that might be enhanced, and a slower backend. Whereas surveying in this area is necessary and important to give the reader and users more detail and knowledge in various areas, Keras follows best practices to reduce cognitive load. It enables consistent and simple application programming, reducing the number of user actions required for common use cases. When an error occurs, it provides clear and actionable feedback, and this is of interest. Also, the authors used the details of the techniques/tools, datasets, accuracy, and the important findings outlined in the mentioned library.

Acknowledgement

The authors would like to thank the IT Department, Technical College of Informatics Akre and Technical College of Administration, Duhok Polytechnic University, Duhok, Kurdistan Region, IRAQ

References

[1] Ott, J., Pritchard, M, Best, N., Linstead, E., Curcic, M. and Baldi, P. (2020). A Fortran-Keras deep learning bridge for scientific computing, *Sci. Program.*, vol. 2020

[2] Tan, S. W. B., Naraharisetti, P. K., Chin, S. K. and Lee, L. Y. (2020). Simple Visual-Aided Automated Titration Using the Python Programming Language, *ACS Publications*, 2020

[3] Baptista L. (2021). Using Python and Google Colab to Teach Physical Chemistry During Pandemic. ChemRxiv. Cambridge: Cambridge Open Engage

[4] Haji, S. H., & Sallow, A. B. (2021). IoT for Smart Environment Monitoring Based on Python: A Review. Asian Journal of Research in Computer Science, 9(1), 57-70. https://doi.org/10.9734/ajrcos/2021/v9i130215

[5] Lee H, Song J. (2019). Introduction to convolutional neural network using Keras; an understanding from a statistician. CSAM 2019; 26:591-610

[6] Ramasubramanian K., Singh A. (2019) Deep Learning Using Keras and TensorFlow. In: Machine Learning Using R. Apress, Berkeley, CA

[7] Drakopoulos, G. and Mylonas, P. (2020). Evaluating graph resilience with tensor stack networks: A Keras implementation, *Neural Comput. Appl.*, pp. 1–16

[8] Moolayil, J. (2019). An introduction to deep learning and keras, *Learn Keras for Deep Neural Networks*, Springer, pp. 1–16

[9] Bohrer, J. S., Grisci, B. I. and Dorn, M. (2020). Neuroevolution of Neural Network Architectures Using CoDeepNEAT and Keras, *ArXiv Prepr. ArXiv200204634*

[10] Petra, V. and Neruda, R. (2017). Evolving KERAS Architectures for Sensor Data Analysis, Federated Conference on Computer Science and Information Systems (FedCSIS), 2017, pp. 109-112

[11] Alyousfi, A., Deep Learning in Keras - Building a Deep Learning Model, *Stack Abuse*. https://stackabuse.com/deep-learning-in-keras-building-a-deep-learning-model/ (accessed May 06, 2021)

[12] Sahin, Ö. (2021). Integrating Keras Models, in *Develop Intelligent iOS Apps with Swift*, Springer, 2021, pp. 137–164

[13] Atienza, R. (2020). Advanced Deep Learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more. Packt Publishing Ltd, 2020

[14] "Denoising autoencoders with Keras, TensorFlow, and Deep Learning," *PyImageSearch*, Feb. 24, 2020. https://www.pyimagesearch.com/2020/02/24/denoising-autoencoders-with-keras-tensorflow-and-deep-learning/ (accessed May 06, 2021)

[15] Peris, A. and Casacuberta, F. (2018). NMT-Keras: A Very Flexible Toolkit with a Focus on Interactive NMT and Online Learning, *Prague Bull. Math. Linguist.*, vol. 111, no. 1, pp. 113–124, Oct. 2018, doi: 10.2478/pralin-2018-0010

[16] Infante, A. and Bergel, A. (2018). Experience in Bridging Keras for Python with Pharo, p. 7

[17] "Python Keras Advantages and Limitations," *DataFlair*, Apr. 22, 2020. https://data-flair.training/blogs/python-keras-advantages-and-limitations/ (accessed Mar. 25, 2021)

[18] Bisong, E. (2019). *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-4470-8

[19] Maseer, Z. K., Yusof, R., Mostafa, S. A., Bahaman, N., Musa, O., & Al-rimy, B. A. S. (2021). DeepIoT. IDS: Hybrid Deep Learning for Enhancing IoT Network Intrusion Detection. *CMC-Computers Materials & Continua*, *69*(3), 3945-3966

[20] Abdulazeez, A., Ed., (2021). Classification Based on Decision Tree Algorithm for Machine Learning, *J. Appl. Sci. Technol. Trends*, vol. 2, no. 01, pp. 20–28, 2021.

[21] Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N. and Terzopoulos, D. (2021). Image Segmentation Using Deep Learning: A Survey,IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2021.3059968

[22] Charbuty, B. and Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning, *J. Appl. Sci. Technol. Trends*, vol. 2, no. 01, pp. 20–28, 2021

[23] Yuan, Q. Shen, H., Li, T., Li, Z., Li, S., Jiang, Y., Xu, H., Tan, W., Yang, Q., Wang, J., Gao, J., Zhang, L. (2020). Deep learning in environmental remote sensing: Achievements and challenges, *Remote Sens. Environ.*, vol. 241, p. 111716, 2020

[24] Chicho, B. T., Abdulazeez, A. M., Zeebaree, D. Q., and Zebari, D. A. (2021). Machine Learning Classifiers Based Classification for IRIS Recognition, *Qubahan Acad. J.*, vol. 1, no. 2, pp. 106–118

[25] Ab Aziz, M. F., Mostafa, S. A., Foozy, C. F. M., Mohammed, M. A., Elhoseny, M., & Abualkishik, A. (2021). Integrating Elman Recurrent Neural Network with Particle Swarm Optimization Algorithms for an Improved Hybrid Training of Multidisciplinary Datasets. *Expert Systems with Applications*, 115441

[26] Abdullah, D. M., Abdulazeez, A. M. and Sallow, A. B. (2021). Lung cancer prediction and classification based on correlation selection method using machine learning techniques, *Qubahan Acad. J.*, vol. 1, no. 2, pp. 141–149, 2021

[27] Susanty, Sahrul, M., Rahman, A. F., Normansyah, M. D., and Irawan, A. (2019). Offensive Language Detection using Artificial Neural Network, in *2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT)*, Mar. 2019, pp. 350–353. doi: 10.1109/ICAIIT.2019.8834452

[28] Aggarwal, S., Bhatia, M., Madaan, R., and Pandey, H. M. (2021). Optimized Sequential model for Plant Recognition in Keras, in *IOP Conference Series: Materials Science and Engineering*, vol. 1022, no. 1, p. 012118.

[29] Haghighat, E. and Juanes, R. (2020). SciANN: A Keras wrapper for scientific computations and physics-informed deep learning using artificial neural networks, *ArXiv Prepr. ArXiv200508803*, 2020

[30] Gupta, D., Khanna, A., Bhattacharyya, S., Hassanien, A. E. Anand, S. and Jaiswal, A. (2021). Eds., *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2020, Volume 1*, vol. 1165. Singapore: Springer Singapore, 2021. doi: 10.1007/978-981-15-5113-0

[31] Versaci, F. (2020). WaveTF: a fast 2D wavelet transform for machine learning in Keras, ICPR Workshops.

[32] Yang, X., Roop, P., Pearce, H. and Ro, J. W. (2020). A compositional approach using Keras for neural networks in real-time systems, in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020, pp. 1109–1114

[33] Parisi, L., Ma, R., RaviChandran, N. and Lanzillotta, M. (2020). Hyper-sinh: An Accurate and Reliable Function from Shallow to Deep Learning in TensorFlow and Keras, *ArXiv Prepr. ArXiv201107661*, 2020

[34] Grattarola, D. and Alippi, C. (2021), Graph Neural Networks in TensorFlow and Keras with Spektral [Application Notes], *IEEE Comput. Intell. Mag.*, vol. 16, no. 1, pp. 99–106, Feb. 2021, doi: 10.1109/MCI.2020.3039072

[35] Reiser, P., Eberhard, A. and Friederich, P. (2021). Implementing graph neural networks with TensorFlow-Keras, *ArXiv Prepr. ArXiv210304318*, 2021

[36] Balan, K., Santora, M., Faied, M. and Carmona-Galindo, V. D. (2020). Study of Evolution by Automating Hardy Weinberg Equilibrium with Machine Learning Techniques in TensorFlow and Keras, in *2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)*, Jul. 2020, pp. 14–19. doi: 10.1109/ACCTHPA49271.2020.9213202

[37] Yashwanth, M., Chandra, M. L., Pallavi, K., Showkat, D. and Kumar, P. S. (2020). Agriculture Automation using Deep Learning Methods Implemented using Keras," in *2020 IEEE International Conference for Innovation in Technology (INOCON)*, Nov. 2020, pp. 1–6. doi: 10.1109/INOCON50539.2020.9298415.

[38] Manapure, P., Likhar, K. and Kosare, H. (2020). Detecting covid-19 in x-ray images with keras, tensor flow, and deep learning, *assessment*, vol. 2, no. 3, 2020.

[39] Benbrahim, H., Hachimi, H. and Amine, A. (2020). Deep Convolutional Neural Network with TensorFlow and Keras to Classify Skin Cancer Images, *Scalable Comput. Pract. Exp.*, vol. 21, no. 3, pp. 379–390, 2020

[40] Harjoseputro, Y. (2020). A Classification Javanese Letters Model using a Convolutional Neural Network with KERAS Framework, International Journal of Advanced Computer Science and Applications (IJACSA), vol. 11, pp. 106–111, 2020

[41] Senthil Kumar K., Kumar S., Tiwari A. (2021) Realtime Handwritten Digit Recognition Using Keras Sequential Model and Pygame. In: Mahapatra R.P., Panigrahi B.K., Kaushik B.K., Roy S. (eds) Proceedings of 6th International Conference on Recent Trends in Computing. Lecture Notes in Networks and Systems, vol 177. Springer, Singapore

[42] Nagisetty, A. and Gupta, G. P. (2019). Framework for detection of malicious activities in IoT networks using Keras deep learning library, in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 633–637

[43] Mondal, S., Agarwal, K. and Rashid, M. (2019). Deep learning approach for automatic classification of X-ray images using convolutional neural network, in *2019 Fifth International Conference on Image Information Processing (ICIIP)*, pp. 326–331

[44] Refianti, R., Mutiara, A. B. and Priyandini, R. P. (2019). Classification of melanoma skin cancer using convolutional neural network, *IJACSA*, vol. 10, no. 3, pp. 409–417

[45] Tseng, C. and Lee, S. (2019). Design of Digital Differentiator Using Supervised Learning on Keras Framework, in *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, Oct. 2019, pp. 162–163. doi: 10.1109/GCCE46687.2019.9014634

[46] Faker, O. and Dogdu, E. (2019). Intrusion detection using big data and deep learning techniques, in *Proceedings of the 2019 ACM Southeast Conference*, pp. 86–93