Babylonian Journal of Networking
Vol.2023, **pp**. 94–100
DOI: https://doi.org/10.58496/BJN/2023/012; ISSN: 3006-5372
https://mesopotamian.press/journals/index.php/BJN

Review Article

A Comprehensive Analysis of Keras: Enhancing Deep Learning Applications in Network Engineering

Noor Abdalkareem Lafta,[1,*]

[1] *Computer and Communication Engineering dept., Faculty of Engineering, Islamic University of Lebanon (IUL), Lebanon.*

ARTICLE INFO

Article History

Received 05 Sep 2023
Accepted 05 Nov 2023
Published 26 Nov 2023

Keywords

Keras
Convolutional Neural Networks (CNNs)
Face detection
Deep learning
Data Scalability
TensorFlow
Model deployment
Distributed training

ABSTRACT

Python is currently one of the most popular programming languages that is used in the sector; it has surpassed many of its predecessors. There can be numerous reasons that make this programming language chosen by many developers – and much attention is paid to the availability of numerous libraries for various purposes. One of the major differences that Keras has when compared to other such libraries is the fact that it emphasises on usability according to the basic principles. First, Keras provides many options to choose from for deploying models in production, second, good performance is supported by multi-GPU, and third, distributed training is possible. Furthermore, it is very friendly to use and easy to comprehend as well as apply especially during model development. This makes it an excellent, highly interactive, open source program for using python in creating and analyzing deep learning models. This research aims at presenting a multiple perspective analysis of Keras. Keras, a deep Learning API in Python is built on TensorFlow. It smoothly integrates with PyTorch, TensorFlow, CODEEPNEATM, and Pygame so that the efficiency of deep learning models' deployment across different sectors of application is enhanced. Some domains falling under this discipline include; Diagnosis of heart diseases, identification of Health problems, Training Graph Neural Networks, Identification of COVID-19, Skin Tumor Detection and Image recognition. This paper also seeks to refine the understanding of the specificity of Keras fueling reflection on its goals, challenges, milestones and lessons learned within its application.

1. INTRODUCTION

Due to high growth achieved in the past few years, machine learning and deep learning have spread widely in many fields. This growth is however attributed to the enhancement of the processing power, and increased access to large datasets for training, making developers to come up with new applications that improve the dopl of deep learning [1]. Currently, Python can be considered the industry standard for building ANNs as it offers better possibilities than other programming languages that also have this capability [2]. Python [3] offers a variety of scientific libraries that can in diverse ways be used in programmes such as simulations, regression analysis, solving of normal differential equations. They are strong and at the same time simple to use Said Another way Libraries referred to these libraries are both comprehensive and friendly another way.

Python libraries can be incorporated into other applications or projects very easily due to the fact that they are more or less free from constraints, as they are not specific to particular contexts like C++/C libraries. In the most general form, a 'library' is understood as a set of necessary modules stored together with the help of package managers. Python libraries are essential in different sectors including machine learning, data science and analytics, data visualization, image and object manipulation among others. As mentioned before, this article is devoted only to one of the libraries existing in the programming language called Python, namely Keras. Deep learning experts often select Keras [4] to implement their work. This is an application programming interface that is based on TensorFlow and which makes it easier to implement, train as well as evaluate neural networks [5]. When employed in building deep networks Keras leverages the features of TensorFlow framework and offers customization flexibility while being highly user-friendly at the same time [6]. These characteristics are the reasons that make Keras most suitable for the utilization in deep learning.

*Corresponding author. Email: nn4304616@gmail.com

Therefore, the scope of the study is as follows: The objective of this paper is to conduct a multi-dimensional assessment of Keras by aggregating the most up to date research from across disciplines in the last three years. All relevant areas such as goals and scopes, methods and tools, data, issues and dilemmas, valuable findings and results, and measurement and validation are discussed. Further, the work showcases the main method utilized in Keras architecture and the most utilized techniques [7].

Despite Keras's simple syntax, the API, the Model and the Layer classes are highly configurable and thus can be used to create more complicated models. The Functional API helps to build models that are similar to Python functions, graph-based models, and layer reuse [8]. Using such approach one can incorporate experimental or a new developing deep learning frameworks and layers on the base of mastered in the Concept and Layer modules knowledge. Keras is a major improvement since it allows the definition and creation of neural network models without writing a single line of code. In the research work, the procedures for constructing a deep-learning model using Keras as the base model are described in Figure 1:

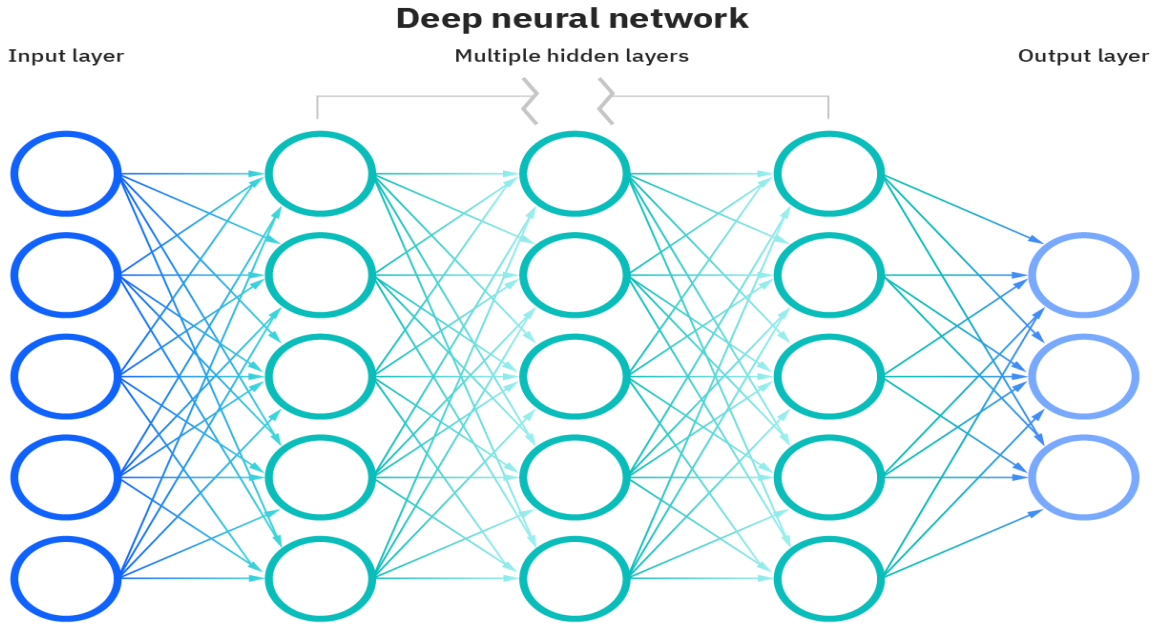

Fig .1. DL in Keras - Building a DL Model [9].

The following are Advantages of Keras over Other Deep Learning Frameworks

a. User-friendly interface for overall usage cases.
b. Basic API building suitable for together beginners as well as experts.
c. Keras does allow for the creation of custom blocks for expansion.
d. Most common deep learning system after TensorFlow.
e. Tensorflow's module facilitates Keras implementation.
f. Links to all Keras features provided by Tensorflow.
g. Utilizes a variety of neural network architectures, including auto-encoders, convolutional, recurrent, long short-term memory, deep belief nets, and deep boltzmann machines.

In addition, Figure. 2 displays the usage of automatic encoders to decrease noise through Keras, TensorFlow, and DL.

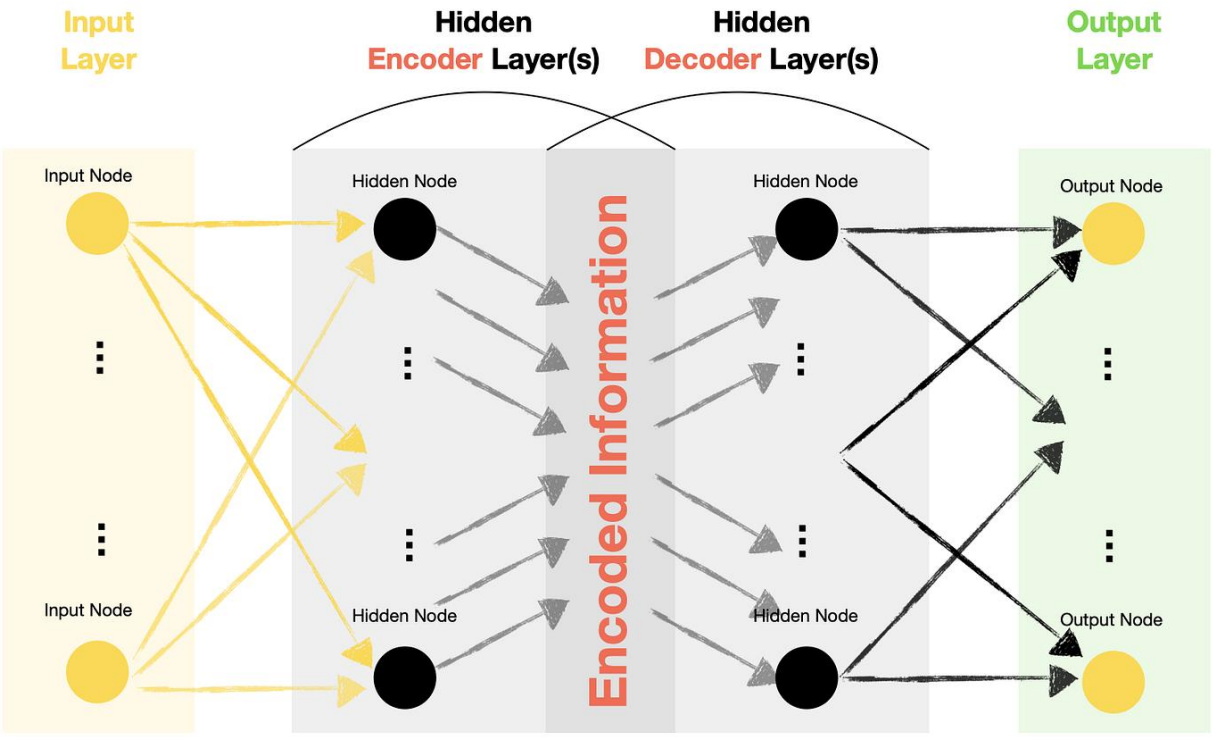

Fig .2. Automatic encoders to decrease noise through Keras, TensorFlow, and DL [10]

2. ADVANTAGES OF LEARNING AND UTILIZAING KERAS

Keras is a widely preferred choice for studying and implementing deep learning algorithms. Its popularity is attributed to its versatility in switching between different computational backends, its user-friendly interface, its capability to construct sophisticated deep neural networks, and its efficiency and effectiveness. Additionally, Keras can utilize multiple GPUs, is compatible with various operating systems, and offers an extensive range of associated products, such as Tensorflow.js, Tensorflow Cloud, Keras Tuner, Tensorflow Lite, and Tensorflow Model Optimization.

2.1. Python Keras Pros and Cons

Keras is an exceptionally effective framework for individuals seeking to gain expertise in neural networks. It provides an advanced conceptual framework that abstracts intricate computational procedures, thereby simplifying the seamless creation of neural network models. Table 1 outlines the advantages and disadvantages of Keras [11].

TABLE I. PROS BESIDES CONS OF KERAS

Pros	Cons
1. Simple, quick to implement.	1. Limitations with low-level API.
2. Comprehensive documentation and strong community support.	2. Certain features need enhancement.
3. Modularity and support for multiple backends.	3. Slower performance compared to backend frameworks.
4. Availability of pre-trained models.	
5. Support for multiple GPUs.	

2.2. Types of Keras Models

Keras has together the Sequential API as well as the extra sophisticated Useful API, enabling users to create models using different approaches.

2.2.1. Sequential Model in Keras

The Sequential Model enables the step-by-step creation of models, where each layer is added in sequence. However, it is not suitable for constructing models with multiple inputs or outputs. This model is most effective with a straightforward stack

of layers, characterized by a single input and a single output tensor. If any layer within this stack involves multiple inputs or outputs, the Sequential Model becomes ineffective. Additionally, this model may not be ideal when a non-linear topology is required [12]. In the below an example to illustrates a sequential model. Shown Figure 3.

```
from keras.models import Sequential
from keras.layers import Dense

# Initialize Sequential model
model = Sequential()

# Add layers to the model
model.add(Dense(units=64, activation='relu', input_shape=(100,)))
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=10, activation='softmax'))

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

# Display the model architecture
model.summary()
```

Fig .3. Example of a sequential model code.

In the above example, Sequential model with three layers:

1. The first layer is a densely connected (fully connected) layer with 64 units and ReLU activation function. It expects input data of shape (batch_size, 100).
2. The second layer is another densely connected layer with 32 units and ReLU activation function.
3. The final layer is a densely connected layer with 10 units and softmax activation function, suitable for multi-class classification tasks.

This Sequential model represents a simple feedforward neural network with one input and one output tensor, following a sequential sequence of layers.

2.2.2. Functional API in Keras

In Keras, the Functional API provides a more extensive set of options for defining models and integrating layers. This API is especially advantageous for creating models with multiple inputs and outputs, allowing for the flexible exchange of layers as necessary. Essentially, the Functional API in Keras enables the construction of layer graphs. Since the Functional API represents a data structure, it can be easily saved as a single file, facilitating the replication of the same model without the original code. Moreover, it simplifies the visualization of the graph and the examination of its nodes [13]. In the below an example to illustrates the functional API. Shown Figure 4.

```
from keras.models import Model
from keras.layers import Input, Dense

# Define input layer
input_layer = Input(shape=(784,))

# Define hidden layers
hidden_layer1 = Dense(128, activation='relu')(input_layer)
hidden_layer2 = Dense(64, activation='relu')(hidden_layer1)

# Define output layer
output_layer = Dense(10, activation='softmax')(hidden_layer2)

# Create the model
model = Model(inputs=input_layer, outputs=output_layer)

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Display the model architecture
model.summary()
```

Fig. 4. Example of the functional API model code.

In the above example, first define an input layer using Input with the shape of our input data (784 features). Then, we define two hidden layers, each with a Dense layer of 128 and 64 units respectively, with ReLU activation function. Finally, we define the output layer with a Dense layer of 10 units and softmax activation function for multi-class classification.

Also, create the model using the Model by specifying the input and output layers. After compiling the model, we can display its architecture using summary (). This example demonstrates the use of the functional API to create a neural network with multiple layers in Keras.

2.3. Distinction Between Tensorflow and Keras

TensorFlow and Keras are prominent deep-learning frameworks frequently used by specialists in the field. These frameworks share significant commonalities and are widely utilized within the deep learning community, collectively representing a substantial portion of deep learning applications. Despite their shared objectives, notable differences exist between TensorFlow and Keras [14], see Table 2.

TABLE II. CONTRASTS BETWEEN TENSORFLOW AND KERAS

TensorFlow	Keras
1. A system offering both high-level and low-level APIs.	1. An API that utilizes TensorFlow, CNTK, and Theano as its backend.
2. Preferred for analyzing complex neural networks.	2. Suitable for rapid deployments of models.
3. Developed by Google's brain team.	3. Initially initiated by François Chollet and developed by a community.
4. Mainly coded in C++, CUDA, and Python.	4. Primarily written in Python, serving as a wrapper for Theano, TensorFlow, and CNTK.
5. Typically utilized for high-performance models and extensive datasets.	5. Often employed for handling small datasets.
6. Backed by support from numerous technology companies.	6. Lacks extensive community outreach.

3. LITERATURE REVIEW

Keras, a lightweight Python deep learning library compatible with Theano or TensorFlow, was developed to streamline the rapid and straightforward integration of deep learning models for research and development purposes. This section of the review discusses recent research relevant to data mining utilizing the Keras library.

Deep learning (DL) enhances system efficiency by introducing intricate layers of data representation, enabling hierarchical structures across multiple levels of abstraction. This advancement leads to improvements in traditional machine learning (ML) techniques. DL excels at feature learning, automatically extracting features from raw data, with higher-level features derived from lower-level ones.

Presently, implementing Deep Learning layers, particularly convolutional and pooling layers, is easiest using the TensorFlow-Keras technique, including substantial graph models, as demonstrated by Reiser et al. [15]. Additionally, recent applications of Keras include disease prediction by Manapure et al. [16], skin cancer classification by Benbrahim et al. [17], and the classification of Javanese characters by Harjoseputro [18].

Lastly, [19] introduced a Keras-based supervised learning approach to digital differentiator creation, while [20] integrated Big Data and DL mechanisms to enhance intrusion detection systems. These examples highlight the broad spectrum of applications where Keras has been successfully employed, underscoring its effectiveness and versatility across diverse fields.

Similarly, in [21], Keras was employed for plant image analysis, achieving a maximum efficiency of 96.3% with a limited dataset of 250 images. In another study [22], the Deep Convolutional Neural Network, TensorFlow, and Keras structures were used to recognize skin tumors in cancer photos. On the other hand, Deep Learning, Keras, and TensorFlow were employed to identify COVID-19 instances in a previous study [23], achieving impressive accuracy rates on X-ray image datasets.

The literature review highlights both the advantages and disadvantages of the Keras library in the Python language. Notable pros include its user-friendly interface, rapid deployment capabilities, comprehensive documentation, extensive community support, modularity, and support for multiple backend structures and pre-trained systems. However, challenges such as problems with the API at the lowest level, the need for certain features to be improved, and relatively slower performance compared to backend systems are noted.

Overall, the findings suggest a strong association between Keras and TensorFlow, with Keras predominantly utilized in conjunction with TensorFlow. However, Keras demonstrates versatility beyond neural networks, being applied to diverse problem domains such as healthcare, agriculture, and plant sciences.

4. CONCLUSION

The Keras library emerges as a user-friendly and versatile Python tool for the creation and evaluation of deep learning (DL) models. Harnessing the computational capabilities of libraries such as Theano and TensorFlow, Keras simplifies the process of defining and training neural networks with minimal code requirements. This analysis endeavors to provide a comprehensive overview of the various dimensions explored in studies concerning the Keras library, particularly within the domain of deep learning. The review underscores some of the primary challenges encountered by Keras, including issues with its Low-level Application Programming Interface (API) that warrant improvement, along with concerns regarding its backend performance. Conducting surveys within this domain is pivotal for furnishing readers and users with deeper insights spanning diverse areas. Keras adheres to commendable practices aimed at minimizing cognitive load, advocating for consistent and straightforward software design principles that streamline common tasks. Moreover, it excels in delivering clear and actionable feedback in the event of errors, thereby enhancing usability. Leveraging insights from a plethora of techniques, datasets, accuracy metrics, and significant findings enriches our understanding of this indispensable library.

Conflicts Of Interest

The absence of any competing relationships or biases that could affect the research is explicitly mentioned in the paper.

Funding

The author's paper asserts that the research was conducted on a voluntary basis and without any financial backing from institutions or sponsors.

Acknowledgment

The author acknowledges the institution for their commitment to fostering a research-oriented culture and providing a platform for knowledge dissemination.

References

[1] J. Ott, M. Pritchard, N. Best, E. Linstead, M. Curcic, and P. Baldi, "A Fortran-Keras deep learning bridge for scientific computing," Sci. Program., vol. 2020, 2020.
[2] S. W. B. Tan, P. K. Naraharisetti, S. K. Chin, and L. Y. Lee, "Simple Visual-Aided Automated Titration Using the Python Programming Language," ACS Publications, 2020.
[3] L. Baptista, "Using Python and Google Colab to Teach Physical Chemistry During Pandemic," ChemRxiv, Cambridge: Cambridge Open Engage, 2021.
[4] S. H. Haji and A. B. Sallow, "IoT for Smart Environment Monitoring Based on Python: A Review," Asian Journal of Research in Computer Science, vol. 9, no. 1, pp. 57–70, 2021, doi: 10.9734/ajrcos/2021/v9i130215.
[5] H. Lee and J. Song, "Introduction to convolutional neural network using Keras; an understanding from a statistician," CSAM, vol. 26, pp. 591–610, 2019.
[6] K. Ramasubramanian and A. Singh, "Deep Learning Using Keras and TensorFlow," in Machine Learning Using R, Apress, Berkeley, CA, 2019.
[7] G. Drakopoulos and P. Mylonas, "Evaluating graph resilience with tensor stack networks: A Keras implementation," Neural Comput. Appl., pp. 1–16, 2020.
[8] J. Moolayil, "An introduction to deep learning and keras," in Learn Keras for Deep Neural Networks, Springer, pp. 1–16, 2019.
[9] J. S. Bohrer, B. I. Grisci, and M. Dorn, "Neuroevolution of Neural Network Architectures Using CoDeepNEAT and Keras," ArXiv Prepr. ArXiv200204634, 2020.
[10] V. Petra and R. Neruda, "Evolving KERAS Architectures for Sensor Data Analysis," in Federated Conference on Computer Science and Information Systems (FedCSIS), 2017, pp. 109–112.
[11] A. Alyousfi, "Deep Learning in Keras - Building a Deep Learning Model," Stack Abuse. [Online]. Available: https://stackabuse.com/deep-learning-in-keras-building-a-deep-learning-model/. [Accessed: May 06, 2021].
[12] Ö. Sahin, "Integrating Keras Models," in Develop Intelligent iOS Apps with Swift, Springer, 2021, pp. 137–164.
[13] R. Atienza, 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. [Online]. Available: https://www.pyimagesearch.com/2020/02/24/denoising-autoencoders-with-keras-tensorflow-and-deep-learning/. [Accessed: May 06, 2021].

[15] A. Peris and F. Casacuberta, "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] A. Infante and A. Bergel, "Experience in Bridging Keras for Python with Pharo," p. 7, 2018.
[17] "Python Keras Advantages and Limitations," DataFlair, Apr. 22, 2020. [Online]. Available: https://data-flair.training/blogs/python-keras-advantages-and-limitations/. [Accessed: Mar. 25, 2021].
[18] E. Bisong, 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] Z. K. Maseer et al., "DeepIoT. IDS: Hybrid Deep Learning for Enhancing IoT Network Intrusion Detection," CMC-Computers Materials & Continua, vol. 69, no. 3, pp. 3945–3966, 2021.
[20] A. Abdulazeez, Ed., "Classification Based on Decision Tree Algorithm for Machine Learning," J. Appl. Sci. Technol. Trends, vol. 2, no. 01, pp. 20–28, 2021.
[21] S. Minaee et al., "Image Segmentation Using Deep Learning: A Survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021, doi: 10.1109/TPAMI.2021.3059968.
[22] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," J. Appl. Sci. Technol. Trends, vol. 2, no. 01, pp. 20–28, 2021.
[23] Q. Yuan et al., "Deep learning in environmental remote sensing: Achievements and challenges," Remote Sens. Environ., vol. 241, p. 111716, 2020.