Volume 5 Issue 1(Page 1-9)
ISSN: 2709-0248

Top Python-Based Deep Learning Packages: A Comprehensive Review

Yasmin Makki Mohialden[1], Raed Waheed Kadhim[1], Nadia Mahmood Hussien[1], Samira Abdul Kader Hussain[1]

Corresponding Email: ymmiraq2009@uomustansiriyah.edu.iq

[1]Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq

Abstract

Deep learning has transformed artificial intelligence (AI) by empowering machines to execute intricate functions with unparalleled precision. The field claims an array of robust packages and libraries, among which Python, a prominent and celebrated programming language, has emerged as a pivotal choice for deep learning study and development. Python has become a leading language in deep learning due to its simplicity and the vast array of libraries available for developers and researchers. This article thoroughly examines the most broadly adopted deep learning packages within the Python system. The packages under scrutiny include TensorFlow, PyTorch, Keras, Theano, and Caffe. We exactly assess each of these packages to establish their typical features and capabilities. Moreover, the review explores into an in-depth analysis of the assets and weaknesses inherent in each package. This detailed exploration prepares readers with the information necessary to make informed decisions regarding the variety of the most suitable packages custom-made to their specific needs. This comprehensive review aims to propose a nuanced understanding of the landscape of popular deep learning packages and support practitioners and researchers in creation strategic and well-informed choices for their deep learning actions.

Keywords: Python-Based Deep Learning, Deep Learning Packages, Python Deep Learning Libraries

Received : December 17, 2023
Received in Revised: January 07, 2024
Accepted: January 20, 2024

Introduction

Deep learning emerged from artificial neural networks, suggesting 1940s roots. These networks have constant processing units called artificial neurons that inaccurately mimic brain neurites. Dendrites receive input impulses from neighbouring neurons, frequently over 1000. The neuron's cell body receives and integrates the modified signals before transmitting them to the neurite. The neurite sends a signal to neighbouring neurons' dendrites if input exceeds a threshold (Chen et al., 1993). Deep learning can solve complicated problems in voice, NLP, and picture identification. This increase in applications has led to the proliferation of deep learning libraries and packages, each with pros and cons. Choosing the right framework and library for an application is crucial for accuracy and performance (Pattanayak, 2023). This study compares deep learning tools and frameworks on flexibility, accessibility, computational efficiency, and performance. It is essential for deep learning and AI practitioners. Problem Statement: As deep learning libraries grow, practitioners and academics must choose the best framework. Understanding each library's pros and cons is key to optimizing accuracy and performance for certain applications. This paper reviews and compares projecting Python-based deep learning programs to help readers choose the latest options based on flexibility,

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

Volume 5 Issue 1(Page 1-9)
ISSN: 2709-0248

usability, computational efficiency, and performance. The insights presented are crucial for researchers and practitioners navigating deep learning and AI applications. The outline of the paper includes Section 1 which provides an introduction, Section 2 which offers a Review of Top Python-Based Deep Learning Packages, Section 3 Comparative Analysis, and Section 4 presenting the conclusion.

Review of Top Python-Based Deep Learning Packages

TensorFlow

TensorFlow is one of the most important libraries, it is an open-source deep-learning library made by the "Google Brain Team". it works with CNNs, RNNs, and GANs, all of which are deep-learning models.

Table 1. shows some of the most powerful and important features of TensorFlow (Raschka et al., 2020; Singh et al., 2022; Mohammadi et al., 2018)

Specification	**TensorFlow**
Framework	Google open-source machine learning framework
Tasks	Covers a wide range of machine learning tasks, including deep learning, reinforcement learning, and neural networks.
APIs	Provides both high-level and low-level APIs, allowing developers to select the degree of abstraction that works best for their use case.
Computer Languages	Supports a variety of computer languages, including Python, C++, and Java.
Performance	Allows developers to run models on numerous machines to improve performance.
Tools	Includes tools for visualizing and fixing problems, such as TensorBoard, which displays training and testing findings.
Systems	Works on a variety of systems, including Windows, Linux, macOS, iOS, and Android.
Supporting Libraries	Includes pre-trained models and libraries for image recognition, natural language processing, and voice recognition.
Hardware	Can be used with a variety of hardware processors, including GPUs, TPUs, and custom ASICs.
Applications	Is extensively used in industry and research, including self-driving cars, recommender systems, and healthcare.

PyTorch

Facebook's AI Research Lab (FAIR) initially developed PyTorch, an open-source machine learning framework primarily serving applications like deep learning and natural language processing. It is now part of the Linux Foundation. It is known for its dynamic computational graph, which provides flexibility and real-time testing of code.

PyTorch models are defined by inheriting from the torch class. Class Module. This class defines the model's layers and components in __init__() and discusses their relationships in forward(). Multiple loss functions assess the difference between a model's predictions and accurate responses.

Tensor computing in PyTorch is similar to NumPy's array calculation but incorporates GPU acceleration and automated differentiation for neural network training. For autonomous differentiation, PyTorch tensors track a computational graph and gradients. PyTorch is versatile

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

and supports over 200 mathematical operations, allowing builders to employ low-level APIs for complex applications. The framework also supports data loading, preprocessing, model parameter optimization, and model saving and loading. Developed by Facebook AI Research, PyTorch is an open-source deep learning platform with a dynamic computational graph and a user-friendly programming paradigm. It supports various deep learning models and includes numerous visualization and debugging tools, making it highly adaptable for more advanced use cases (Deng, 2019; Xu et al., 2021). The PyTorch specifications are shown in Table 2 (Rasch et al., 2021; Gugger & Howard, 2020).

Table 2. PyTorch specifications

Specification	**TensorFlow**
Framework	Google's open-source machine learning framework
Tasks	Encompasses a broad spectrum of machine learning tasks, including deep learning, reinforcement learning, and neural networks.
APIs	Supplies both high-level and low-level APIs, enabling developers to choose the level of abstraction that suits their use case best.
Programming Languages	Supports a variety of programming languages, including Python, C++, and Java.
Performance	Empowers developers to execute models on numerous machines to enhance overall performance.
Tools	Incorporates utilities for visualizing and addressing issues, such as TensorBoard, which exhibits training and testing findings.
Systems	Functions on a diverse range of systems, spanning Windows, Linux, macOS, iOS, and Android.
Auxiliary Libraries	Encompasses pre-trained models and libraries for tasks like image recognition, natural language processing, and voice recognition.
Hardware Compatibility	Can be utilized with various hardware processors, including GPUs, TPUs, and custom ASICs.
Applications	Widely employed in industry and research, playing a role in self-driving cars, recommender systems, and healthcare.

Keras

Keras, a Python-based deep learning API, was created by Google software engineer and AI researcher François Chollet. It uses JAX, TensorFlow, or PyTorch. Keras handles data processing, hyperparameter tweaking, and deployment within the machine learning workflow to speed up experimentation and simplify deep learning model building and training. A consistent interface, few user actions for typical use scenarios, and clear error messages make the API user-friendly. Excellent documentation and development instructions are prioritized.

To simplify layer stacking and setup, Keras models are defined using the Sequential class or Functional API. For neural network construction, configuration, training, and evaluation, it supports several layers and parameters, including loss functions and optimizers. Pure TensorFlow tensors work with Keras, making it a useful model definition add-on. With different data preparation methods, Keras supports 1D and 2D convolutions and recurrent neural networks. Long-term deployment and reuse need model saving and loading tools.

Keras, known for its simplicity and versatility in academia and business, is a fundamental component of YouTube Discovery's new modelling infrastructure, showing its practical applicability in huge environments. Keras API supports several architectures and facilitates deep learning model construction with an easy-to-use interface. Key features make it a flexible

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

and frequently used open-source Python-based artificial neural network toolkit. According to Table 3, François Chollet's API is used in real-world systems like YouTube Discovery's modeling infrastructure (Shobha et al., 2023; Ahmadibeni, 2020; Harjoseputro, 2020).

Table 3. Key Specifications of Keras

Specification	Keras
Performance	A high-level neural networks API written in Python that operates on top of TensorFlow, Theano, or CNTK (Microsoft Cognitive Toolkit).
Applications	A popular choice for deep learning beginners due to its intuitive, easy-to-use, and user-friendly nature.
Model	Provides many pre-built layers, including convolutional, recurrent, and pooling layers, enabling the creation of complex models.
API	Offers both sequential and functional API styles for building models, providing users with flexibility in designing models.
Syntax	Utilizes a simple syntax and methods for compiling, training, and evaluating models.
Tools	Includes a built-in data preprocessing tool that simplifies training data loading.
Hardware	Compatible with GPUs, TPUs, and can seamlessly work with Keras.
Applications	Widely utilized in business and research for tasks such as image recognition, natural language processing, and recommender systems.

MXNet

Amazon's open-source deep learning framework, Apache MXNet, supports CNNs, RNNs, and generative adversarial networks (GANs) and is strong and adaptable. MXNet supports several languages and trains models quickly because of its scalability. Portable and lightweight, it scales across GPUs on computers. MXNet uses Gluon and other high-level APIs to facilitate development. Gluon is a high-level interface that combines API flexibility with simplicity of use. MXNet supports imperative Python model training and symbolic graph deployment. Researchers like MXNet for speedy prototyping and outcome assessment. With imperative programming, researchers can control computation. MXNet accommodates most systems and supports several data types, including Amazon S3 cloud storage. In 2019, MXNet's support for Horovod improved. Uber created the distributed learning framework Horovod. Two high-level MXNet Python API packages are the Gluon API and the Module API. For beginners, the Gluon API is more versatile and user-friendly. MXNetProcessor in the Amazon SageMaker Python SDK runs processing operations in Docker containers with a controlled MXNet environment for neural network training and deployment. Amazon's open-source deep learning framework, MXNet, supports several architectures and is scalable, flexible, and easy to program. High-level APIs like Gluon and imperative and symbolic programming make it adaptable for academics and developers (Guo et al., 2020; Nguyen et al., 2019; Chen et al., 2015).

Table 4. MXNet Specifications

Specification	MXNet
Framework	An open-source deep learning system.
Performance	Enables developers to execute models on multiple machines, enhancing efficiency.

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

API	Provides both high-level and low-level APIs, allowing users to select the best abstraction for their specific use case.
Languages	Supports Python, C++, Julia, R, and Matlab.
Tools	Incorporates built-in and distinctive neural network layers.
Hardware	Compatible with GPUs and TPUs.
Applications	Widely utilized in both industry and research for tasks such as computer vision, natural language processing, and speech detection.

Theano

The Theano package is an open-source Python tool from the Montreal Institute for Learning Algorithms (MILA) that handles mathematical expressions, particularly multi-dimensional arrays. It was designed by developers as a popular deep learning package for computing complex neural network algorithms. Theano supports various deep learning activities such as image classification, speech recognition, and NLP. It also supports CNNs, RNNs, and autoencoders. Optimizing mathematical computation is a crucial aspect of Theano. It automatically optimizes CPU and GPU utilization to accelerate computations. GPUs can perform data-intensive operations faster than CPUs, making Theano an efficient tool. Theano also provides visualization and debugging tools for model creation and validation. It represents and manipulates computation graphs well. The programming model of Theano optimizes, evaluates, and defines mathematical expressions, especially machine learning model-building parameters.The Python scientific computing tool NumPy works well with Theano. It creates symbolic graphs for gradient computation through symbolic differentiation. The development of Theano stopped in 2017 due to heavy industrial rivalry. The PyMC development team was renamed Aesara. When not using mini-batches and on models with at least one hidden layer, Theano outperforms Torch 5. Major Theano parameters are listed in bullet points in Table 5 (Shatnawi et al., 2018; Niraula & El, 2022).

Caffe

Berkeley AI Research (BAIR) created Caffe, an open-source deep learning system called Convolutional Architecture for Fast Feature Embedding. Caffe, licensed under the BSD 2-Clause, is fast, modular, and expressive. Caffe is compatible with several deep learning architectures, such as CNNs and RNNs. It is highly efficient for image classification and convolutional models, processing over 60 million photos daily with a single NVIDIA K40 GPU. Caffe's programming model is simple. Configuration-based models and optimisation enable CPU-GPU switching with a single flag. This simplifies GPU training and deployment to commodity clusters or mobile devices. Caffe supports high-level APIs, including Caffe2, introduced by Facebook in April 2017 and incorporated into PyTorch. Caffe's drawbacks include restricted interaction with other deep learning frameworks and the need to define models in configuration files, which can be difficultn (Mathew et al., 2021; Sarkar, 2018).Table 6 contains bullet-pointed Caffe specs.

Table 6. Caffe Characteristics

Characteristic	**Caffe**
Tools	Caffe was developed for computer vision applications by the Vision and Learning Center (BVLC).
Support	Facilitates the creation and training of neural networks with support for convolutional and pooling layers.
Hardware	Allows high-performance training on both CPU and GPU hardware.

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

Models	Provides Python and command-line interfaces for model creation and execution.
Applications	Widely utilized in industry and research for tasks such as image recognition, object detection, and segmentation.
Update	The most recent update occurred in 2018.

Comparative Analysis

The choice of a deep learning framework depends on the specific needs and requirements of a project. TensorFlow, PyTorch, Keras, Theano, and Caffe each have unique strengths well-suited for different types of deep learning tasks. Understanding the capabilities and limitations of each framework helps developers make informed decisions about their projects. Table 7 is a comparison of some factors between TensorFlow, PyTorch, Keras, Theano, and Caffe.

Table 7. Comparison of some factors between TensorFlow, PyTorch, Keras, Theano, and Caffe.

Factor	**TensorFlow**	**PyTorch**	**Keras**	**Theano**	**Caffe**
Popularity	Most widely adopted.	Experienced a decline recently.	Not as widely embraced recently.	Diminished popularity recently.	Famous in deep learning.
Language	Implemented in Python.	Utilizes Python and C++.	Coded in Python.	Crafted using C++.	Developed using C++.
Ease of Use	Recognized for user-friendliness.	Features dynamic computational graph for flexibility.	More intricate but offers increased control.	Relatively user-friendly with some limitations.	Known for ease with speed focus.
Performance	Acknowledged as one of the fastest.	Demonstrates high-speed performance.	Exhibits rapid processing.	Efficient computation, ensuring swift performance.	Optimized for speed on CPU and GPU.
Documentation	Extensive and well-documented.	Comprehensive with numerous examples.	Good documentation.	All good but less extensive.	Well-documented with speed focus.
Community	Large and vibrant community.	Sizable and active communities.	Smaller communities.	Potential challenges in finding support.	Smaller community, may have limited support.
Flexibility	Offers simplicity with limitations.	Highly adaptable and customizable.	Intricate but offers significant control.	Relatively user-friendly but with	Highly flexible and customizable.

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

				limited flexibility.	
Compatibility	Compatible with a wide array of systems.	Compatible with most systems but with some limitations.			
Integration	Can be integrated with various tools.	Excels in compatibility with additional tools.	Can be seamlessly integrated with other tools.		Can be combined with other tools.

Which deep learning framework is best for small datasets?

Keras has been suggested for small-data deep learning. Keras, a high-level neural network API, works with TensorFlow, Theano, and the Microsoft Cognitive Toolkit. It is ideal for rapid deep neural network development and validation on small datasets. Transfer learning refines pre-trained models for tiny datasets. This approach uses deep learning with restricted data by modifying a model learned on a large set of data to perform well on an inferior dataset. Deep learning-based frameworks like GenerativeMTD resolve limited data restrictions. Generative MTD creates synthetic data from small datasets to train deep learning models with pseudo-real data. When choosing a deep learning framework, project needs such task difficulty, speed, efficiency, and computational resources must be considered.

Conclusion

In this comprehensive analysis of the major Python-based deep learning packages, we examined TensorFlow, PyTorch, Keras, MXNet, Theano, and Caffe. Programmers may choose a deep learning package depending on use case and requirements, allowing them to customize their project. Our comparison showed that each library has pros and cons. Flexibility and support for the most popular languages make TensorFlow and PyTorch stand out. Keras streamlines model construction and training with its efficient structure. MXNet is a strong and scalable deep learning model developer and deployer. Although rare, Theano and Caffe make model construction and deployment fast. After considering these programs' usability, Keras is the most straightforward. Caffeine and theano are less common yet effective for certain uses.

Deep learning trends and Python-based package improvements are important as it evolves. Continuous research and development may improve these libraries, solve restrictions, and increase their capabilities. By adding reinforcement learning and generative adversarial networks, these tools may be useful in new domains. Future research may include in-depth case studies to demonstrate these packages' real-world applications. This would reveal their performance and appropriateness for certain sectors or applications. These packages will evolve with open-source community cooperation and input as deep learning advances. Regular updates and contributions from developers worldwide will improve and innovate the Python-based deep learning environment.

References

Ahmadibeni, A. (2020). *Aerial Vehicles Automated Target Recognition of Synthetic SAR Imagery Using Hybrid Stacked Denoising Autoencoders* (Doctoral dissertation, Tennessee State University).

Chen, S., Mulgrew, B., & Grant, P. M. (1993). A clustering technique for digital communications channel equalization using radial basis function networks. *IEEE Transactions on neural networks*, *4*(4), 570-590.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., ... & Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

Deng, Y. (2019, May). Deep learning on mobile devices: a review. In *Mobile Multimedia/Image Processing, Security, and Applications 2019* (Vol. 10993, pp. 52-66). SPIE.

Gugger, S., & Howard, J. (2020). *Deep Learning for Coders with Fastai and PyTorch*. O'Reilly Media, Incorporated..

Guo, J., He, H., He, T., Lausen, L., Li, M., Lin, H., ... & Zhu, Y. (2020). Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. *The Journal of Machine Learning Research*, *21*(1), 845-851.

Harjoseputro, Y. (2020). A classification Javanese letters model using a convolutional neural network with KERAS framework. International Journal of Advanced Computer Science and Applications, 11(10).

Mathew, A., Amudha, P., & Sivakumari, S. (2021). Deep learning techniques: an overview. *Advanced Machine Learning Technologies and Applications: Proceedings of AMLTA 2020*, 599-608.

Mohammadi, M., Al-Fuqaha, A., Sorour, S., & Guizani, M. (2018). Deep learning for IoT big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, *20*(4), 2923-2960.

Nguyen, G., Dlugolinsky, S., Bobák, M., Tran, V., López García, Á., Heredia, I., ... & Hluchý, L. (2019). Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review*, *52*, 77-124.

Niraula, D., & El Naqa, I. (2022). Software Tools for Machine and Deep Learning. In *Machine and Deep Learning in Oncology, Medical Physics and Radiology* (pp. 117-133). Cham: Springer International Publishing.

Pattanayak, S. (2023). Introduction to deep-learning concepts and TensorFlow. In *Pro Deep Learning with TensorFlow 2.0: A Mathematical Approach to Advanced Artificial Intelligence in Python* (pp. 109-197). Berkeley, CA: Apress..

Rasch, M. J., Moreda, D., Gokmen, T., Le Gallo, M., Carta, F., Goldberg, C., ... & Narayanan, V. (2021, June). A flexible and fast PyTorch toolkit for simulating training and inference on analog crossbar arrays. In *2021 IEEE 3rd international conference on artificial intelligence circuits and systems (AICAS)* (pp. 1-4). IEEE..

Raschka, S., Patterson, J., & Nolet, C. (2020). Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information*, *11*(4), 193.

Sarkar, D., Bali, R., & Ghosh, T. (2018). *Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras*. Packt Publishing Ltd.

Shatnawi, A., Al-Bdour, G., Al-Qurran, R., & Al-Ayyoub, M. (2018, April). A comparative

DOI: https://doi.org/10.47667/ijpasr.v5i1.283

Volume 5 Issue 1(Page 1-9)
ISSN: 2709-0248

study of open source deep learning frameworks. In *2018 9th international conference on information and communication systems (icics)* (pp. 72-77). IEEE.

Shobha, Y., Prasad, K. V., Anuradha, S. G., & Vaidya, H. (2023). Auto Skin Tumour Classification Using CNN Framework with Tensorflow and Keras. *Mathematical Statistician and Engineering Applications*, *72*(1), 172-184..

Singh, N. B., Singh, M. M., & Sarkar, A. (2022). Deep learning architectures, libraries and frameworks in healthcare. In *Deep learning, machine learning and IoT in biomedical and health informatics* (pp. 221-248). CRC Press.

Xu, Y., Liu, X., Cao, X., Huang, C., Liu, E., Qian, S., ... & Zhang, J. (2021). Artificial intelligence: A powerful paradigm for scientific research. *The Innovation*, *2*(4).

DOI: https://doi.org/10.47667/ijpasr.v5i1.283