INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

IMPORTANT PYTHON LIBRARIES FOR MAKING AI TOOLS

Piyush Dalmia[1], Gaurav Sati[2], Jeevesh Kumar[3]

[1,2,3]Jagan Institute of Management Studies G.G.S.I.P.U New Delhi, India.

ABSTRACT

Python has become the go-to programming language in the expansive field of artificial intelligence (AI), playing a crucial role in the development, research, and implementation of modern technology. Its prominence within the AI community can be attributed to several factors, including a rich array of libraries and frameworks, its simplicity and readability, versatility, strong community support, and seamless integration capabilities.

The Python ecosystem offers a wide range of libraries tailored for various AI tasks. NumPy and SciPy provide robust support for numerical and scientific computing, enabling complex mathematical operations and statistical analyses that serve as the foundation of AI algorithms. Pandas, another essential library, facilitates data manipulation and analysis, offering efficient data structures like Data frames. Additionally, Matplotlib and Seaborn aid in data visualization, assisting in the interpretation of patterns and trends within extensive datasets.

For machine learning endeavors, Scikit-learn stands out as a comprehensive library with a diverse range of algorithms for classification, regression, clustering, and more. Its user-friendly interface and tools for model selection and evaluation make it invaluable for practitioners. In the realm of deep learning, TensorFlow and PyTorch have emerged as dominant frameworks, providing high-level abstractions for building and training neural networks. The integration of Keras with TensorFlow further simplifies the construction and training of deep learning models.

Keywords: Python, artificial intelligence, libraries, NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, PyTorch

1. INTRODUCTION

Python has emerged as a central pillar in the dynamic landscape of artificial intelligence (AI), driving a significant transformation in how we conceive, develop, and deploy intelligent systems. Its dominance in the realm of AI stems from a convergence of factors, ranging from its adaptable and user-friendly nature to a robust ecosystem of libraries and frameworks. This introduction seeks to explore the symbiotic relationship between Python and AI, examining the key factors that position Python as a formidable catalyst in shaping the future of intelligent technologies.

Python's Versatility:

The ascendancy of Python in the AI domain is rooted in its inherent versatility. As a general- purpose programming language, Python transcends traditional barriers, seamlessly adapting to a wide array of applications within AI. Whether the task involves machine learning, natural language processing, computer vision, or reinforcement learning, Python emerges as a ubiquitous and adaptable tool. This versatility extends beyond the breadth of AI applications to encompass the entire spectrum of development, from rapid prototyping to the deployment of complex AI systems.

Ecosystem of Libraries and Frameworks

At the core of Python's prowess in AI lies its extensive ecosystem of libraries and frameworks, serving as the backbone of AI development. NumPy and SciPy provide a solid foundation for numerical and scientific computing, enabling the intricate mathematical operations essential for AI algorithms. The Pandas library empowers data scientists and engineers with flexible data structures, while Matplotlib and Seaborn offer sophisticated tools for data visualization.

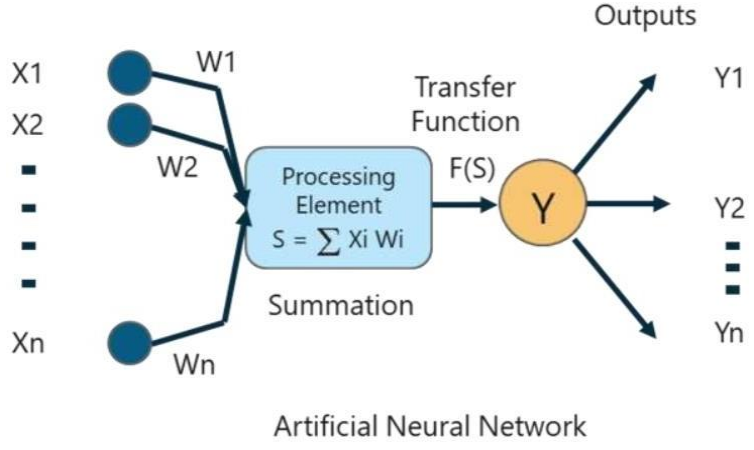

Figure 1: Artificial Neural Network

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

PYTHON LIBRARIES : AN OVERVIEW

NumPy and SciPy: Foundational Libraries for Numerical Computing

NumPy and SciPy form the backbone of numerical and scientific computing in Python. NumPy provides efficient data structures and functions for array manipulation, enabling complex mathematical operations essential for AI algorithms. SciPy extends NumPy's capabilities with additional functionalities for optimization, integration, and linear algebra, making it indispensable for AI researchers and developers.

- Inbuild Functions used for AI in NumPy Library

numpy.array():	Create arrays from Python lists or tuples.
numpy.zeros():	Create an array filled with zeros.
numpy.ones():	Create an array filled with ones.
numpy.random.rand():	Generate random numbers from a uniform distribution.
numpy.random.randn():	Generate random numbers from a normal distribution.
numpy.dot():	Compute dot product of two arrays.
numpy.linalg.inv():	Compute the inverse of a matrix.
numpy.mean():	Compute the mean of an array.

- Inbuild Functions used for AI in SciPy Library:

scipy.stats.norm():	Probability density function, cumulative distribution function, and random variates for a normal distribution.
scipy.optimize.minimize():	Minimize a function using various optimization algorithms.
scipy.interpolate.interp1d():	1-dimensional interpolation.
scipy.signal.convolve():	Compute the convolution of two arrays.
scipy.linalg.solve():	Solve a system of linear equations.
scipy.sparse.linalg.eigs():	Compute some eigenvalues of a sparse matrix.
scipy.cluster.vq.kmeans():	Perform K-means clustering.

Pandas: Data Manipulation Made Easy

Pandas simplifies data manipulation and analysis in Python, offering powerful data structures like Data frames for handling structured data. Its intuitive interface facilitates tasks such as data cleaning, transformation, and aggregation, making it an essential tool for preprocessing datasets in AI projects.

Inbuild Functions used for AI in Pandas Library:

pandas.DataFrame():	Create a DataFrame from data like ndarray, lists, dictionaries, or another DataFrame.
DataFrame.head():	Return the first n rows.
DataFrame.tail():	Return the last n rows.
DataFrame.describe():	Generate descriptive statistics of DataFrame columns.
DataFrame.merge():	Merge DataFrame objects by performing a database-style join operation.
DataFrame.pivot_table():	Create a spreadsheet-style pivot table.
DataFrame.plot():	Plot data.

Matplotlib and Seaborn: Visualizing Insights

Matplotlib and Seaborn are indispensable libraries for data visualization in Python. They provide a wide range of plotting functions and customization options, enabling users to create insightful visualizations to interpret patterns and trends within datasets. These libraries play a crucial role in communicating findings and insights derived from AI models.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

- Inbuild Functions used for AI in Matplotlib Library:

matplotlib.pyplot.plot():	Plot lines and/or markers.
matplotlib.pyplot.scatter():	Create a scatter plot.
matplotlib.pyplot.hist():	Plot a histogram.
matplotlib.pyplot.imshow():	Display an image.
matplotlib.pyplot.bar():	Plot a bar chart.
matplotlib.pyplot.boxplot():	Make a box and whisker plot.
matplotlib.pyplot.contour():	Plot contours.
matplotlib.pyplot.annotate():	Annotate points with labels.

- Inbuild Functions used for AI in Seaborn Library:

seaborn.scatterplot()	Draw a scatter plot with possibility of several semantic groupings.
seaborn.lineplot()	Draw a line plot with possibility of several semantic groupings.
seaborn.barplot()	Show point estimates and confidence intervals as rectangular bars.
seaborn.distplot()	Flexibly plot a univariate distribution of observations.
seaborn.pairplot()	Plot pairwise relationships in a dataset.
seaborn.jointplot()	Draw a scatter or hexbin plot with marginal histograms.

Scikit-learn: Machine Learning Made Accessible

Scikit-learn is a comprehensive machine learning library in Python, offering a plethora of algorithms and utilities for classification, regression, clustering, and more. Its user-friendly interface and extensive documentation make it accessible to both novice and experienced practitioners, facilitating the development and evaluation of AI models.

- Inbuild Functions used for AI in Scikit-learn Library:
- Various supervised and unsupervised learning algorithms such as Linear Regression, Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, k-Nearest Neighbors, K-Means Clustering, etc.
- Model evaluation functions like cross_val_score(), accuracy_score(), confusion_matrix(), etc.
- Preprocessing functions like StandardScaler(), MinMaxScaler(), OneHotEncoder(), etc.
- Feature selection methods like SelectKBest(), RFE (Recursive Feature Elimination), etc.

TensorFlow and PyTorch: Powering Deep Learning

TensorFlow and PyTorch are leading deep learning frameworks that have revolutionized AI research and development. TensorFlow, developed by Google, provides a flexible ecosystem for building and training deep neural networks, with support for distributed computing and production deployment. PyTorch, backed by Facebook, offers dynamic computation graphs and a pythonic API, making it popular among researchers for its ease of use and flexibility.

- Inbuild Functions used for AI in TensorFlow Library:

tf.constant()	Create a constant tensor.
tf.Variable()	Create a variable tensor.
tf.add()	Add two tensors element-wise.
tf.matmul()	Multiply two tensors.
tf.nn.relu()	Apply rectified linear activation function.
tf.keras.layers.Dense()	Create a fully connected layer.
tf.keras.optimizers.Adam()	Create an Adam optimizer.
tf.keras.losses.BinaryCrossentropy()	Binary cross-entropy loss function.
tf.keras.metrics.Accuracy()	Compute classification accuracy.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

- Inbuild Functions used for AI in TensorFlow Library:

torch.tensor()	Create a tensor.
torch.zeros()	Create a tensor filled with zeros.
torch.ones()	Create a tensor filled with ones.
torch.rand()	Generate random numbers from a uniform distribution.
torch.randn()	Generate random numbers from a normal distribution.
torch.mm()	Matrix multiplication.
torch.nn.ReLU()	Rectified Linear Unit activation function.
torch.nn.Linear()	Fully connected layer.
torch.optim.Adam()	Adam optimizer.
torch.nn.CrossEntropyLoss()	Cross-entropy loss function for classification.

2. METHODOLOGY

Problem Identification and Understanding: In the realm of artificial intelligence (AI), utilizing Python involves a structured approach to creating, implementing, and deploying intelligent systems. Leveraging Python's flexibility, extensive library ecosystem, and collaborative community, this methodology navigates the complexities of AI development. This comprehensive approach delves into the core aspects of Python's role in AI.

The initial step in this methodology revolves around clearly defining the problem at hand, requiring a profound understanding of the domain and the specific challenges AI can address. Whether tackling image recognition, natural language processing, or predictive analytics, Python's adaptability empowers developers and data scientists to articulate the problem and its constraints in alignment with AI capabilities.

Data Collection and Preparation: At the foundation of AI lies data, and Python excels in managing data-related tasks. This stage involves acquiring pertinent datasets from either public repositories or proprietary sources.

Python's data science libraries, such as Pandas, facilitate data manipulation, cleansing, and preprocessing. This phase is critical for ensuring that the data is formatted suitably for training and testing AI models.

Exploratory Data Analysis (EDA): Python's data visualization libraries, such as Matplotlib and Seaborn, are instrumental in the exploratory data analysis phase.

EDA encompasses visualizing and scrutinizing the dataset to uncover patterns, trends, and potential relationships. This step is pivotal for making informed decisions regarding feature engineering and model selection as the process progresses.

Methodology: Within the domain of machine learning, Scikit-learn stands out as a prominent toolkit, offering a comprehensive suite of algorithms and tools for model development, evaluation, and deployment.

3. LITERATURE-REVIEW

Literature Review: Python's Role in Artificial Intelligence

The convergence of Python programming and artificial intelligence (AI) has triggered a significant transformation in the development, deployment, and study of intelligent systems. This literature review aims to explore seminal research and articles elucidating Python's significance within the domain of AI, highlighting its contributions, challenges, and the evolving landscape of this dynamic relationship.

The Ubiquity of Python in AI Development: Numerous studies underscore Python's pervasive presence in AI development. Géron's (2019) "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" emphasizes Python's dominance in machine learning (ML) through tools like Scikit-Learn and its seamless integration with deep learning frameworks like Keras and TensorFlow.

This work serves as a foundational resource, emphasizing Python's role in constructing end-to-end ML pipelines. Müller and Guido's (2017) "Introduction to Machine Learning with Python" further solidifies Python's status as a preferred language for ML.

The authors explore Scikit-Learn and showcase Python's adaptability in handling diverse ML tasks, highlighting the language's simplicity and readability, which fosters collaboration among interdisciplinary teams.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

Frameworks and Libraries in Python: Python's success in AI is closely intertwined with its rich ecosystem of libraries and frameworks. A comprehensive review by Brownlee (2020) in "Machine Learning Mastery" discusses the pivotal role of Python libraries such as NumPy, Pandas, Matplotlib, and Scikit-Learn in streamlining complex AI tasks. The author underscores the efficiency and productivity enhancements derived from Python's extensive toolkit.

4. RESULT

The influence of Python within the realm of artificial intelligence (AI) is profound and extensive, yielding substantial outcomes across various facets of AI development, research, and deployment. Python's primary role as a programming language in this context has ushered in a new era characterized by accessibility, innovation, and collaboration, thereby becoming a pivotal force in shaping the landscape of intelligent technologies.

Accessibility and Versatility- Python's adoption in AI has significantly enhanced accessibility for developers, data scientists, and researchers. It's clear, concise, and human-readable syntax makes it ideal for individuals at all skill levels, democratizing AI development and enabling contributions from diverse backgrounds. Python's versatility is another crucial aspect of its impact, providing a unified and adaptable platform for tasks ranging from machine learning to natural language processing. This versatility is evident in its extensive ecosystem of libraries and frameworks catering to different facets of AI, facilitating seamless transitions between tasks and domains.

Proliferation of Libraries and Frameworks: Python's success in AI is closely tied to its rich ecosystem of specialized libraries and frameworks. Libraries like NumPy and SciPy provide comprehensive support for numerical and scientific computing, while Pandas facilitates effective data manipulation and analysis. In machine learning, Scikit-learn serves as a go-to library for various tasks, while TensorFlow and PyTorch offer high-level abstractions for building and training neural networks. This proliferation of specialized tools simplifies complex AI tasks and accelerates development cycles, contributing to rapid progress in the field.

Accelerated Development and Prototyping

Python's syntax and expressiveness significantly contribute to accelerated development and prototyping of AI solutions. Because it is so readable, researchers and developers may concentrate on the ideas and logic of their algorithms instead of being mired in complex terminology.This agility is particularly valuable in the iterative process of experimentation and prototyping, where researchers can quickly translate ideas into code, test hypotheses, and refine models. Jupyter Notebooks further enhance the prototyping experience by enabling interactive computing and collaborative exploration of AI development.

Community Collaboration and Knowledge Sharing- Python's success in AI is not only attributable to its technical capabilities but also to its vibrant and collaborative community. The Python community serves as a hub for knowledge sharing, collaboration, and collective problem-solving. Online forums, conferences, and open-source projects create an ecosystem where developers, researchers, and practitioners actively contribute to the collective understanding and advancement of AI. Platforms like GitHub facilitate collaborative coding, project collaboration, and issue resolution, fostering an environment where best practices and innovative approaches are disseminated rapidly.

Integration with Cutting-edge Technologies- Python's adaptability extends beyond AI-specific libraries to encompass integration with cutting-edge technologies. It seamlessly interfaces with databases, incorporates AI into web services, and integrates AI components into existing systems, enabling the development of holistic and efficient solutions leveraging diverse technologies. Its support for emerging technologies like edge computing and the Internet of Things (IoT) positions Python as a preferred language for deploying intelligent systems at the edge of networks.

Real-world Applications and Industry Adoption- Python's impact in AI is evident in the proliferation of real-world applications and widespread industry adoption across diverse sectors. Organizations leverage Python for developing and deploying AI solutions, thanks to its scalability and efficiency in handling large-scale data processing, real-time inference, and robust performance. Its applications span healthcare, finance, natural language processing, and various other industries, showcasing its versatility and relevance in diverse domains.

Ethical AI and Responsible Practices- Python's impact in AI extends to ethical considerations and responsible AI practices. As AI applications become integral to decision-making processes, transparency, fairness, and accountability become paramount. Python, with its emphasis on readable and understandable code, supports the implementation of ethical guidelines and the development of fair and transparent AI models. Frameworks and libraries within the Python ecosystem, such as Fairness Indicators in TensorFlow and Fairlearn in Scikit-learn, underscore the commitment to addressing biases and promoting fairness in AI systems. Researchers and practitioners actively contribute to the discourse on ethical AI, advocating for responsible practices and the mitigation of unintended consequences.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

PYTHON'S PIVOTAL ROLE IN SHAPING THE LANDSCAPE OF ARTIFICIAL INTELLIGENCE

The pivotal role of Python in shaping the landscape of artificial intelligence (AI) cannot be overstated. Its ascent within the realm of AI represents a significant paradigm shift, fundamentally altering how we conceive, develop, and deploy intelligent systems. Reflecting on Python's journey in the context of AI reveals its transformation from a preferred tool to a fundamental force driving innovation, collaboration, and responsible practices in the field.

Versatility and Accessibility

At the core of Python's impact on AI lies its unparalleled versatility and accessibility. Python's syntax serves as a gateway that welcomes individuals from diverse backgrounds into the world of AI. Its simplicity and readability have democratized AI, empowering a broad range of individuals to contribute meaningfully to the development of intelligent systems. This accessibility has broken down barriers and fostered a more inclusive AI community.

The versatility of Python is evident in its adaptability to a wide array of AI applications. Whether the task involves traditional machine learning, deep learning, natural language processing, computer vision, or reinforcement learning, Python offers a unified platform. This versatility encourages a holistic approach to problem-solving and system development, allowing practitioners to seamlessly transition between different facets of AI.

Ecosystem of Libraries and Frameworks

Python's impact on AI is further exemplified by its robust ecosystem of libraries and frameworks tailored to meet the specific demands of the field. Libraries like NumPy, SciPy, and Pandas provide a solid foundation for numerical and data-centric operations, while Scikit- learn simplifies the development process with its comprehensive suite of machine learning algorithms. TensorFlow and PyTorch cater to the ever-growing demand for deep learning capabilities. The proliferation of specialized tools within the Python ecosystem has significantly accelerated AI development. Researchers and developers no longer face the daunting task of building algorithms from scratch; instead, they can leverage these libraries to focus on the nuances of their specific AI tasks. This accelerates the pace of innovation and encourages a collaborative environment where practitioners can build upon existing work, contributing to the collective knowledge of the AI community.

Collaboration and Community Dynamics

Python's impact extends beyond its technical capabilities to the collaborative and dynamic nature of its community. The open-source ethos and collaborative spirit of the Python community have created an environment where knowledge sharing and collective problem- solving thrive. Online forums, conferences, and collaborative platforms such as GitHub have become arenas for the exchange of ideas, solutions to challenges, and the development of shared resources. The collaborative nature of Python is especially evident in platforms like Jupyter Notebooks. These interactive environments facilitate collaborative research and allow researchers to share not only code but also visualizations and narratives. The transparency and openness inherent in Python's community dynamics contribute to a culture of continuous learning and improvement, shaping the trajectory of AI development.

Real-world Impact and Industry Adoption

Python's impact in the context of AI is not confined to the theoretical realm but is tangibly felt in real-world applications across diverse industries. From healthcare to finance, from natural language processing to computer vision, Python has permeated various sectors, enabling the development of AI solutions that address real-world challenges. The language's scalability and efficiency make it a preferred choice for industry applications, where large-scale data processing, real-time inference, and robust performance are imperative. Python has become the backbone of AI systems powering recommendation engines, autonomous vehicles, predictive analytics, and a myriad of other applications, contributing to the digital transformation of industries globally.

Rapid Prototyping and Iterative Development

Python's impact is particularly pronounced in the realm of rapid prototyping and iterative development. The language's expressive syntax and the availability of tools like Jupyter Notebooks allow researchers and developers to quickly translate ideas into code, experiment with different algorithms, and iterate on models. This agility has catalyzed breakthroughs and innovation, fostering an environment where ideas can be tested, refined, and implemented in a short span.

The iterative development process supported by Python is essential for the dynamic nature of AI research. As new algorithms and models emerge, Python's flexibility allows researchers to experiment and adapt swiftly, contributing to the iterative nature of the AI development lifecycle.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 05, May 2024, pp: 1160-1166

Ethical Considerations and Responsible AI

Python's impact in the context of AI transcends technical prowess to ethical considerations and responsible AI practices. The readability of Python code promotes transparency, a crucial factor in addressing concerns related to bias, fairness, and accountability in AI systems. The language provides a platform for the implementation of ethical guidelines and the development of fair and transparent AI models. Frameworks and libraries within the Python ecosystem, such as Fairness Indicators in TensorFlow and Fairlearn in Scikit-learn, exemplify the commitment to addressing biases and promoting fairness in AI systems. The discourse on ethical AI within the Python community underscores the importance of responsible practices, ensuring that AI technologies are developed and deployed with a conscious awareness of their societal impact.

Challenges and Continuous Improvement

While Python's impact on AI is overwhelmingly positive, challenges persist. Scalability and performance optimization have been areas of concern, especially in computationally intensive AI tasks. Researchers and developers have responded by actively addressing these challenges, proposing optimizations, and exploring avenues to enhance Python's efficiency in handling large-scale data and complex computations. Additionally, as AI technologies become more integrated into everyday life, concerns related to privacy, security, and the ethical implications of AI decisions are gaining prominence. The Python community remains engaged in the ongoing discourse on these issues, emphasizing the importance of continuous improvement, responsible development practices, and ethical considerations in the AI landscape.

5. CONCLUSION

In conclusion, the impact of Python in the context of artificial intelligence is nothing short of transformative. From democratizing AI by making it accessible to a broad audience to fostering collaboration, accelerating development, and promoting ethical considerations, Python has emerged as a linchpin in the AI revolution. Its versatility, supported by a rich ecosystem of libraries and frameworks, has enabled the development of sophisticated AI systems that permeate every facet of our lives. As Python continues to evolve alongside the dynamic field of AI, its impact is poised to deepen. The collaborative spirit of the Python community, coupled with the language's versatile nature and robust ecosystem, ensures that Python will remain at the forefront of AI innovation. With ongoing efforts to address challenges, promote responsible practices, and advance ethical considerations, Python's role in shaping the landscape of artificial intelligence will only grow stronger in the years to come.

6. REFERENCES

[1] R. K. Kaushik Anjali and D. Sharma, "Analyzing the Effect of Partial Shading on Performance of Grid Connected Solar PV System", 2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE), pp. 1-4, 2018.

[2] R. Kaushik, O. P. Mahela, P. K. Bhatt, B. Khan, S. Padmanaban and F. Blaabjerg, "A Hybrid Algorithm for Recognition of Power Quality Disturbances," in IEEE Access, vol. 8, pp. 229184-229200, 2020.

[3] Purohit, A. N., Gautam, K., Kumar, S., & Verma, S. (2020). A role of AI in personalized health care and medical diagnosis. International Journal of Psychosocial Rehabilitation, 10066–10069.

[4] Kumar, R., Verma, S., & Kaushik, R. (2019). Geospatial AI for Environmental Health: Understanding the impact of the environment on public health in Jammu and Kashmir.

[5] International Journal of Psychosocial Rehabilitation, 1262–1265.

[6] Kaushik, R. K. "Pragati. Analysis and Case Study of Power Transmission and Distribution." J Adv Res Power Electro Power Sys 7.2 (2020): 1-3.

[7] Akash Rawat, Rajkumar Kaushik and Arpita Tiwari, "An Overview Of MIMO OFDM System For Wireless Communication", International Journal of Technical Research & Science, vol. VI, no. X, pp. 1-4, October 2021.

[8] R. Kaushik, O. P. Mahela, P. K. Bhatt, B. Khan, S. Padmanaban and F. Blaabjerg, "A Hybrid Algorithm for Recognition of Power Quality Disturbances," in IEEE Access, vol. 8, pp. 229184-229200, 2020.

[9] Kaushik, R. K. "Pragati. Analysis and Case Study of Power Transmission and Distribution." J Adv Res Power Electro Power Sys 7.2 (2020): 1-3.

[10] Kaushik, M. and Kumar, G. (2015) “Markovian Reliability Analysis for Software using Error Generation and Imperfect Debugging” International Multi Conference of Engineers and Computer Scientists 2015, vol. 1, pp. 507-510.