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

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 01, January 2024, pp : 143-150

e-ISSN : 2583-1062

Impact Factor : 5.725

OPTIMIZING DEEP LEARNING MODELS FOR INSTANCE SEGMENTATION IN DISTRIBUTED ENVIRONMENTS

A Srinivasa Rao[1], Koya Haritha[2], Kusuma Polanki[3], Roja D[4]

[1]Assoc. Professor, Department of CSE-AI, Chalapathi Institute of Technology, Guntur, India, 522016.

[3]Assoc. Professor, Department of CSE, Chalapathi Institute of Technology, Guntur, India, 522016.

[2,4]Asst. Professor, Department of CSE-Data Science, Chalapathi Institute of Technology, Guntur, India, 522016.

ABSTRACT

Instance segmentation, a crucial task in computer vision, involves identifying and delineating individual objects within images. In this context, leveraging the capabilities of a distributed deep learning big data cluster becomes imperative to handle the computational demands of training complex models on large datasets. This paper presents an approach to instance segmentation using distributed computing resources. The process begins with meticulous data preparation, ensuring a well-labeled dataset with instance-level segmentation masks. The distributed computing cluster is configured, equipped with GPUs, and tailored for efficient data and computation distribution. A suitable deep learning model, such as Mask R-CNN or YOLACT, is selected, with adjustments made to accommodate the characteristics of the dataset and cluster resources. Parallelism is employed at both the data and model levels. Data parallelism facilitates the distribution of training data across cluster nodes, while model parallelism addresses the challenge of large models that may exceed individual GPU memory capacities. Distributed training strategies, including gradient synchronization and parameter updates, orchestrate the collaborative training process. Optimization techniques, such as mixed-precision training and distributed batch normalization, enhance training efficiency. Validation and testing phases ensure the model's generalization and performance on unseen data. Post-training, the model is deployed on the distributed cluster for real-time or batch inference, with optimizations geared towards scalability. Monitoring tools track performance metrics and resource utilization, enabling insights into the distributed instance segmentation system. The ability to scale the cluster dynamically ensures adaptability to varying dataset sizes and model complexities.

This abstract encapsulates the key steps and considerations in the implementation of instance segmentation on a distributed deep learning big data cluster, addressing the challenges posed by large-scale datasets and intricate model architectures. The approach outlined in this paper provides a comprehensive framework for researchers and practitioners seeking to harness distributed computing for efficient and scalable instance segmentation in computer vision applications.

Keywords- Distributed deep learning, BigDl, Spark, Instance segmentation, Azure databricks.

1. INTRODUCTION

In recent years, the field of computer vision has witnessed unprecedented advancements, with instance segmentation standing out as a pivotal task in visual recognition systems. Instance segmentation involves not only classifying objects within images but also precisely delineating each individual instance. The efficacy of deep learning models in addressing this complex task has been remarkable; however, as datasets grow in scale and model architectures become increasingly intricate, the computational demands escalate. This paper addresses the imperative need to optimize deep learning models for instance segmentation within distributed computing environments. The advent of big data clusters equipped with powerful Graphics Processing Units (GPUs) provides an opportune platform to tackle the computational challenges associated with training sophisticated models on vast datasets. As we delve into the intricacies of distributed computing, we aim to develop strategies that enhance the efficiency, scalability, and performance of instance segmentation models.

The journey towards optimizing deep learning models for distributed instance segmentation unfolds in several stages. We commence with a comprehensive examination of the inherent complexities of instance segmentation and the role of deep learning in meeting these challenges. Subsequently, we explore the landscape of distributed computing, highlighting the advantages and considerations when applied to instance segmentation tasks. The significance of parallelism, both in terms of data and model, emerges as a critical aspect of our optimization strategy. Furthermore, we survey prominent instance segmentation models, selecting those that exhibit compatibility with distributed computing paradigms. Model selection considerations are discussed, with a focus on adapting architectures to the distributed environment and tailoring hyperparameters for optimal performance.

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

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 01, January 2024, pp : 143-150

e-ISSN : 2583-1062

Impact Factor : 5.725

The paper is structured to guide researchers, practitioners, and enthusiasts through the intricate process of optimizing deep learning models for instance segmentation in distributed environments. By the end, we aim to provide a comprehensive framework that not only addresses the technical intricacies but also contributes to the broader conversation on the role of distributed computing in advancing the capabilities of computer vision systems for real-world applications.

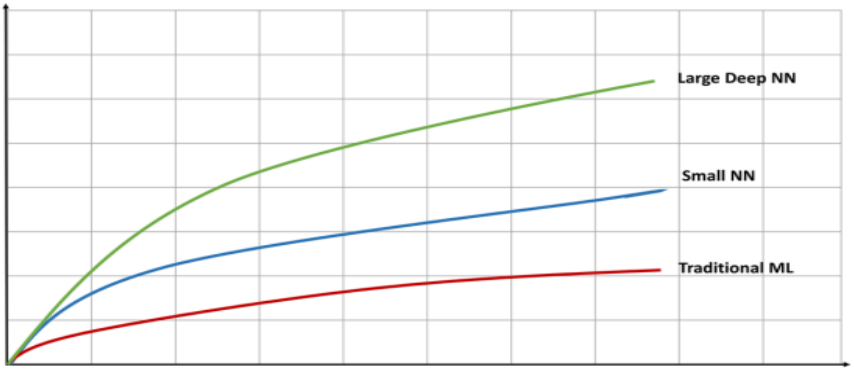

Figure 1: The relationship between model performance and the size of the trained data is generally positive, with larger amounts of training data leading to improved performance in deep learning models

A survey of distributed deep learning frameworks:

Distributed deep learning frameworks have become instrumental in addressing the ever-growing computational demands of training complex models on vast datasets. This survey navigates the landscape of distributed deep learning frameworks, providing a comprehensive overview of the existing solutions, their key features, and their applicability to diverse use cases. We explore the evolution of these frameworks, highlighting their role in enabling parallelism, scalability, and efficiency in deep learning tasks.

The survey begins by establishing the foundational concepts of distributed computing in the context of deep learning, emphasizing the necessity of these frameworks in handling large-scale datasets and intricate model architectures. We then delve into an extensive review of prominent distributed deep learning frameworks, encompassing both industry-standard and emerging solutions. Each framework is scrutinized based on its architecture, support for popular deep learning libraries, ease of use, scalability, and performance. Practical considerations, such as hardware compatibility and support for various neural network architectures, are also discussed. The survey provides insights into the strengths and limitations of each framework, facilitating informed decisions for researchers and practitioners in selecting the most suitable solution for their specific requirements.

Furthermore, we explore emerging trends and advancements in the field, including efforts towards interoperability between frameworks, support for heterogeneous computing environments, and integration with specialized hardware accelerators. The survey concludes with a discussion on future directions and challenges, paving the way for ongoing research in the dynamic and rapidly evolving landscape of distributed deep learning frameworks.

This comprehensive survey serves as a valuable resource for both novices and experts in the field, offering a holistic understanding of the state-of-the-art in distributed deep learning frameworks and guiding the community towards advancements that will shape the future of scalable and efficient deep learning.

Motivation and addressing challenges in the utilization of distributed deep learning on big data clusters: Distributed deep learning is motivated by the increasing volume and complexity of data in real-world applications, and the need to train and deploy large and complex deep learning models. DDL efficiently processes and learns from this data by distributing computational tasks across multiple machines. DDL is better than single machine deep learning, DDL offers several advantages over traditional single-machine deep learning, especially when deploying large and complex deep learning models on big data clusters these advantages include Scalability, DDL can scale to train and deploy models on very large datasets and more complex models, which is essential for many deep learning applications; Fault tolerance, Big data clusters are typically designed to be fault-tolerant, meaning that they can continue to operate even if some of the nodes fail. This makes them a good choice for deploying DDL applications, as it can help to reduce the risk of training failures Speeding Up Training Time, By distributing the computational workload across multiple machines, DDL can significantly reduce the time required to train models and inference of deep learning models, which can be critical for time-sensitive applications Cost-effectiveness, DDL can help to reduce the cost of training and deploying deep learning models by using distributed computing resources. DDL reduces the cost of training and deploying deep learning models by distributing the workload across commodity machines in a

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 01, January 2024, pp : 143-150

cluster, which improves utilization of available resources; and Resource sharing, by sharing resources among multiple applications, organizations can make better use of their existing resources and avoid the need for additional hardware investments. DDL on Big Data clusters provides a scalable, efficient, and robust solution for training deep learning models on large datasets As the demand for deep learning continues to grow, DDL is expected to become even more widely adopted in the future. DDL offers significant benefits for training machine and deep learning models, including faster training times, improved accuracy, and increased scalability. DDL works by distributing computational tasks across multiple machines in a big data cluster. This allows DDL applications to scale to very large datasets and complex models, such as VGG networks or Inception Resent network. DLL significantly reduces the training time and improves the accuracy of very large models on very large datasets. While GPUs are preferred for training due to their high performance, CPUs are sufficient and more attractive for data preprocessing and inference, which are less resource-intensive. There is a growing interest in developing CPU-optimized deep learning frameworks and algorithms, as CPUs are more widely available and less expensive.

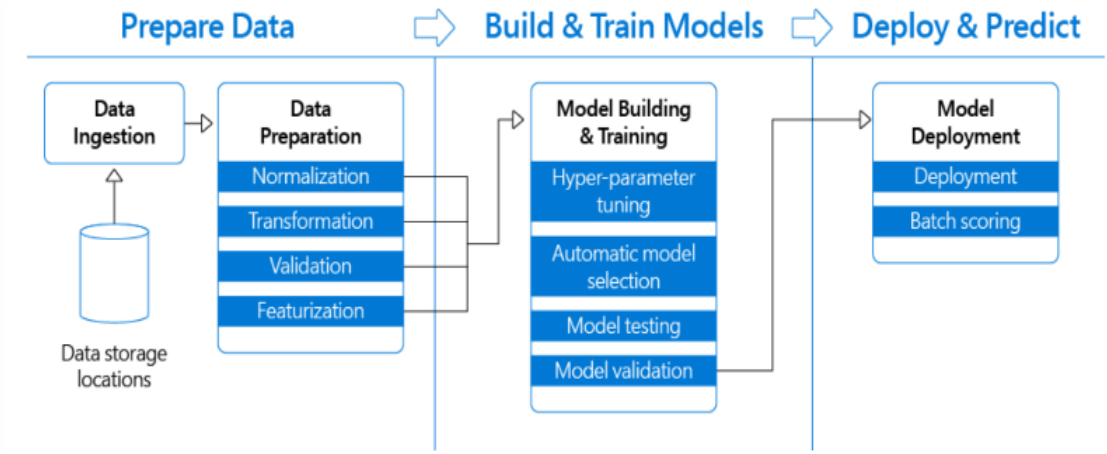

Figure 2: Hyper-Parameter Tuning

HDFS is designed to be a distributed file system that can handle large-scale data across a cluster of machines.It breaks down large files into smaller blocks (typically 128 MB or 256 MB in size) and distributes these blocks across multiple nodes in the cluster. HDFS follows a decentralized metadata architecture, where file metadata (information about files such as their location, size, and access permissions) is distributed across the nodes in the cluster. The metadata is managed by the NameNode, which stores information about the structure and organization of the file system, and DataNodes, which store the actual data blocks.

The NameNode is a central point of coordination for the file system but does not store the data itself. It maintains metadata about files and directories. DataNodes are responsible for storing and retrieving the actual data blocks. They communicate with the NameNode for metadata operations. In summary, HDFS is indeed a distributed file system, and its metadata architecture is decentralized, with metadata distributed across multiple nodes in the cluster. This design enhances scalability and fault tolerance in handling large volumes of data in distributed computing environments.

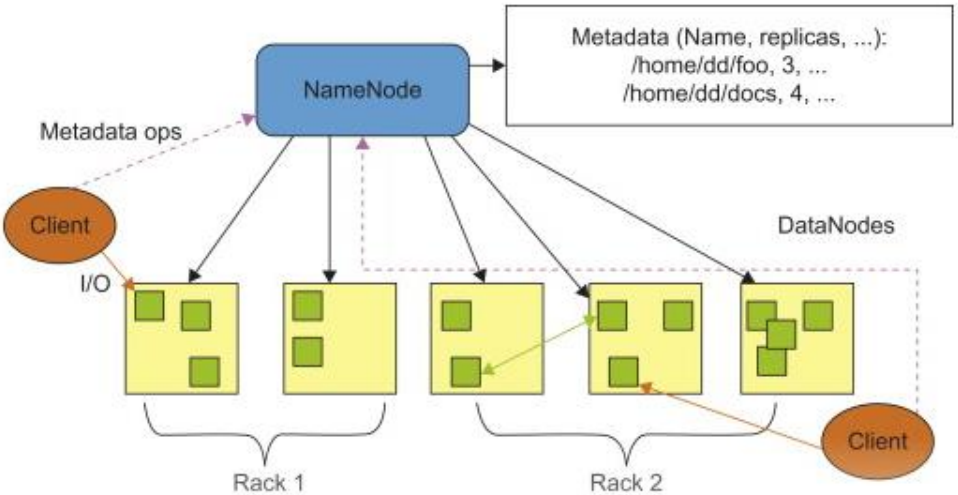

Figure 3: HDFS is a centralized metadata distributed file system

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 01, January 2024, pp : 143-150

Algorithm 1: Synchronous SGD with Parameter Server Pseudocode for Distributed Training

Require: Training data D, batch size B, number of workers N, learning rate γ
Initialize model parameters θ randomly
Distribute θ to all worker nodes
for $t = 1$ to maximum number of iterations **do**
  for $i = 1$ to N in parallel **do**
    Worker i samples a mini-batch of size B from D
    Worker i computes gradients g_i using mini-batch and current θ
    Worker i sends g_i to the parameter server
  end for
  Parameter server aggregates gradients from all workers:
  $\hat{g} = \frac{1}{N} \sum_{i=1}^{N} g_i$
  Parameter server updates θ:
  $\theta \leftarrow \theta - \frac{\gamma}{N} \hat{g}$
  Distribute updated θ to all worker nodes
end for

Instance segmentation deep learning model Instance segmentation is a complex computer vision task that assigns distinct labels to separate instances of objects belonging to the same class, providing pixel-specific object instance masks. Unlike semantic segmentation which classifies pixels based on classes, instance segmentation models classify pixels based on "instances". These algorithms can distinguish overlapping or very similar object regions based on their boundaries, regardless of the class a classified region belongs to. In semantic segmentation, smaller objects (with fewer pixels) are less significant. However, in instance segmentation, all objects, regardless of size, are equally important. This distinction makes instance segmentation applicable in various real-world scenarios. For example, in self-driving car technology, a vehicle navigating complex street scenarios such as crowded pedestrian areas or construction sites needs a detailed understanding of its surroundings. Instance segmentation plays a crucial role in this context. In the medical domain, instance segmentation has a wide range of applications. For instance, histopathology images usually contain numerous nuclei of various shapes surrounded by cytoplasm. Recognizing and segmenting these nuclei using instance segmentation can aid in detecting severe diseases like cancer. It's also used for detecting tumors in MRI brain scans. Satellite imagery is another area where instance segmentation is highly useful. Major applications include identifying and counting cars, detecting ships for maritime security, preventing oil spills monitoring marine pollution, and segmenting buildings for geospatial analysis. Given that objects in satellite imagery are typically small and closely spaced concerning the image's resolution, pixel-wise methods are not very effective. Therefore, instance segmentation network architecture can provide better separation between objects by understanding each object as a separate instance.

2. RESULT ANALYSIS

Optimizing model inference is vital for real-time deep learning applications like autonomous driving, focusing on throughput, memory, and energy. The process starts with converting the YOLACT PyTorch model to the ONNX format, a universal format for deep learning models that enhances interoperability between various AI frameworks. ONNX, an open-source project co-developed by Microsoft, Amazon, and Face book, facilitates model conversion between any framework and the ONNX format. It supports faster inference using the ONNX model on the supported ONNX Runtime and is compatible with numerous machine learning frameworks like Tensor Flow, PyTorch, and Scikit-learn. ONNX Runtime is designed for high performance and supports both CPU and GPU hardware. It also supports distributed training and inference across multiple devices and machines for improved performance and scalability [39]. The conversion of the YOLACT PyTorch model to ONNX format faced challenges due to unsupported layers in YOLACT, such as those in the Feature Pyramid Networks (FPN) class. The FPN class, which is optimized using torch script mode, allows for parallel multi-threading through the torch.jit.script module. This Just-In-Time (JIT) compiler improves the efficiency of PyTorch models in production. The FPN class extracts multi-level features for topdown up-sampling and fusion, creating multi-scale depth image features. However, this class inherits from the torch.jit.ScriptModule class, which often lacks tensor shape information, leading to potential size mismatch errors between ONNX and TorchScript definitions. PyTorch operates in two modes: eager mode for immediate operation execution and intuitive programming, and script mode, which compiles PyTorch code into an optimized TorchScript format for efficient execution in repeated prediction scenarios. PyTorch's JIT compilation optimizes

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 01, January 2024, pp : 143-150

TorchScript modules using runtime information, capturing the structure of a PyTorch model and saving it in a serialized format for use in different environments or devices. This is done without unifying the input dimensions of tensors in the FPN. When converting a TorchScript-saved PyTorch model to ONNX, issues such as unsupported operations can occur. However, all YOLACT layers are supported in ONNX OpsetVersion 11. Another issue is related to tensor shapes - while PyTorch allows dynamic tensor shapes, ONNX requires static shapes. To address this issue in the detect transformation part used in the Backbone Network. The input and output shapes were set before converting the model to ONNX in five FPN layers dimension sizes as [(69, 69), (35, 35), (18, 18), (9,9), (5,5)]. This made it compatible with the ONNX format. Modifications were made to the YOLACT model to return prediction outputs directly from the forward method before post-processing. The Just-In-Time (JIT) compiler was disabled and the model was exported in ONNX format using OpsetVersion=11.

The model was then visualized using the Netron App. The exported ONNX model was simplified using ONNX Simplifier, which removes redundant operators. The simplified model was then converted into OpenVINO format using the OpenVINO toolkit.

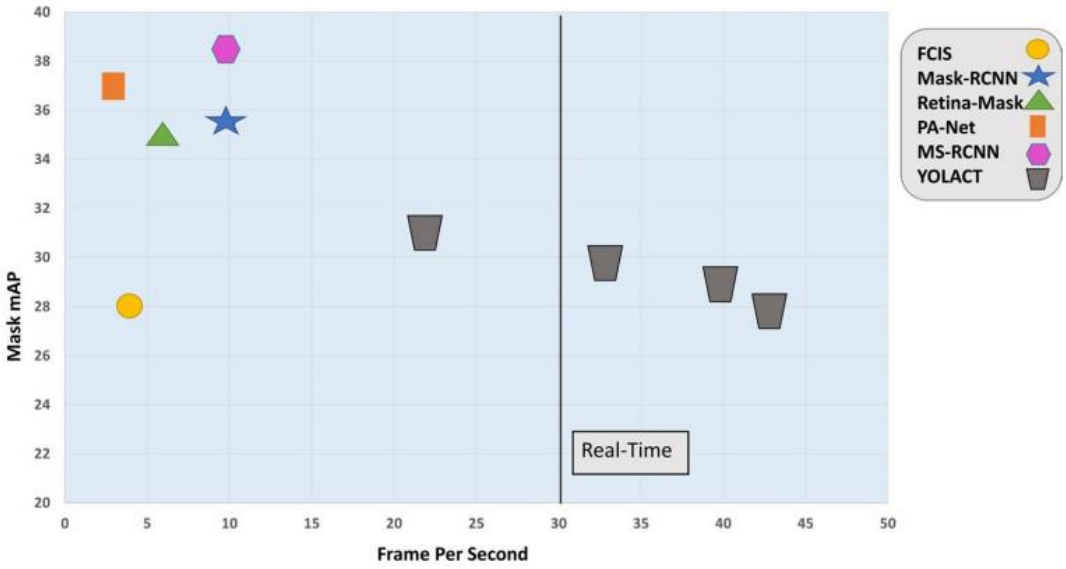

Figure 4: Speed-performance trade-of for various instance segmentation methods

Figure 5: Ground truth mask coefficients for reference to mask coefficient predictions from YOLACT PyTorch, ONNX, and OpenVINO-FP16

Table 1: Shows the execution time taken for the process

Model name	Total time taken	OneFrame--Video Length time (s)	Machine configuration details
YOLACT Original	45 min 4s	3.00–15 (Video Length)	Python version 3.6
ONNX Model	26 min 9 s	1.74–15	Anaconda windows 10
OpenVINO-FP16	29 min 50 s	1.98–15	With 16 GB RAM
OpenVINO-FP32	29 min 38 s	1.97–15	Torch V 1.10.2+cpu
YOLACT Original	54 min 43 s	3.64–15 (Video Length)	Ubuntu 20.04 VM
ONNX Model	32 min 48 s	2.18–15	Torch V 1.8.1+cu102
OpenVINO-FP16	30 min 10 s	2.01–15	4 GB RAM-python 3.6
OpenVINO-FP32	30 min 9 s	2.01–15	Total cores = 4
YOLACT Original	35 min 47 s	2.30–15 (Video Length)	Ubuntu 20.04 VM
ONNX Model	32 min 48 s	2.10–15	Torch version 1.2.0
OpenVINO-FP16	24 min 22 s	1.60–15	6 GB RAM-python 3.7

3. CONCLUSION

In conclusion, this paper presents a distributed system designed to deploy the optimized YOLACT instance segmentation model, a large and intricate deep learning model, both on-premises and in the cloud. The end-to-end data loading and preprocessing pipeline is detailed, and the inference time is evaluated across various frameworks, including PyTorch, ONNX, and OpenVINO. The experimental results highlight the significant acceleration in inference time achieved by increasing the number of executors across the cluster, potentially leading to cost savings for server resources. Despite the promising outcomes, the study identifies some challenges, specifically excessive memory usage issues. These challenges underline the need for further work in optimizing memory management within the distributed system. The identified issues provide valuable insights for future improvements, ensuring the scalability and efficiency of the proposed system.

3.1 Future Work:

3.1.1 Addressing Memory Usage Issues: Investigate and implement strategies to mitigate excessive memory usage observed during the experiments. This could involve optimizing data loading processes, exploring memory-efficient data structures, or employing advanced memory **management techniques.**

3.1.2 Exploring BigDL on Distributed GPU Cluster: Further investigate the feasibility of running BigDL on a distributed GPU cluster using Spark 3.x. This exploration could offer insights into leveraging GPU resources for improved model training and inference, potentially enhancing the overall performance of the distributed system.

3.2.1 Optimizing Resource Allocation: Explore dynamic resource allocation strategies to optimize the utilization of computing resources, especially in cloud environments. This includes efficient scaling based on workload demands and minimizing resource wastage during idle periods.

4. REFERENCES

[1] Vellela, S. S., &Balamanigandan, R. (2022, December). Design of Hybrid Authentication Protocol for High Secure Applications in Cloud Environments. In 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 408-414). IEEE.

[2] Madhuri, A., Jyothi, V. E., Praveen, S. P., Sindhura, S., Srinivas, V. S., & Kumar, D. L. S. (2022). A New Multi-Level Semi-Supervised Learning Approach for Network Intrusion Detection System Based on the 'GOA'. Journal of Interconnection Networks, 2143047.

[3] Vellela, S. S., Reddy, B. V., Chaitanya, K. K., &Rao, M. V. (2023, January). An Integrated Approach to Improve E-Healthcare System using Dynamic Cloud Computing Platform. In 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 776-782). IEEE.

[4] S Phani Praveen, RajeswariNakka, AnuradhaChokka, VenkataNagarajuThatha, SaiSrinivasVellela and UddagiriSirisha, "A Novel Classification Approach for Grape Leaf Disease Detection Based on Different Attention Deep Learning Techniques" International Journal of Advanced Computer Science and Applications(IJACSA), 14(6), 2023. http://dx.doi.org/10.14569/IJACSA.2023.01406128

[5] Praveen, S. P., Sarala, P., Kumar, T. K. M., Manuri, S. G., Srinivas, V. S., &Swapna, D. (2022, November). An Adaptive Load Balancing Technique for Multi SDN Controllers. In 2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS) (pp. 1403-1409). IEEE.

[6] Vellela, S. S., BashaSk, K., &Yakubreddy, K. (2023). Cloud-hosted concept-hierarchy flex-based infringement checking system. International Advanced Research Journal in Science, Engineering and Technology, 10(3). Vellela, S. S., &Balamanigandan, R. (2023).

[7] Sk, K. B., Roja, D., Priya, S. S., Dalavi, L., Vellela, S. S., & Reddy, V. (2023, March). Coronary Heart Disease Prediction and Classification using Hybrid Machine Learning Algorithms. In 2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA) (pp. 1-7). IEEE.

[8] Kiran Kummineni, Ratna Babu Pilli, K. Tejaswi, P. Venkata Siva,Attention-based Bayesian inferential imagery captioning maker,Materials Today: Proceedings,2023,ISSN 2214-7853,https://doi.org/10.1016/j.matpr.2023.05.231.

[9] VenkateswaraRao, M., Vellela, S., Reddy, V., Vullam, N., Sk, K. B., &Roja, D. (2023, March). Credit Investigation and Comprehensive Risk Management System based Big Data Analytics in Commercial Banking. In 2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS) (Vol. 1, pp. 2387-2391). IEEE.

[10] Vellela, S.S., Balamanigandan, R. Optimized clustering routing framework to maintain the optimal energy status in the wsn mobile cloud environment. Multimed Tools Appl (2023). https://doi.org/10.1007/s11042- 023-15926-5

[11] Vullam, N., Vellela, S. S., Reddy, V., Rao, M. V., SK, K. B., &Roja, D. (2023, May). Multi-Agent Personalized Recommendation System in E-Commerce based on User. In 2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 1194-1199). IEEE.

[12] K. N. Rao, B. R. Gandhi, M. V. Rao, S. Javvadi, S. S. Vellela and S. KhaderBasha, "Prediction and Classification of Alzheimer's Disease using Machine Learning Techniques in 3D MR Images," 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 2023, pp. 85-90, doi: 10.1109/ICSCSS57650.2023.10169550.

[13] Vellela, S. S., BashaSk, K., &Javvadi, S. (2023). MOBILE RFID APPLICATIONS IN LOCATION BASED SERVICES ZONE. MOBILE RFID APPLICATIONS IN LOCATION BASED SERVICES ZONE", International Journal of Emerging Technologies and Innovative Research (www. jetir. org| UGC and issn Approved), ISSN, 2349-5162.

[14] Venkateswara Reddy, B., &KhaderBashaSk, R. D. Qos-Aware Video Streaming Based Admission Control And Scheduling For Video Transcoding In Cloud Computing. In International Conference on Automation, Computing and Renewable Systems (ICACRS 2022).

[15] Madhuri, A., Praveen, S. P., Kumar, D. L. S., Sindhura, S., &Vellela, S. S. (2021). Challenges and issues of data analytics in emerging scenarios for big data, cloud and image mining. Annals of the Romanian Society for Cell Biology, 412-423.

[16] Vellela, S. S., Balamanigandan, R., & Praveen, S. P. (2022). Strategic Survey on Security and Privacy Methods of Cloud Computing Environment. Journal of Next Generation Technology, 2(1).

[17] Reddy, N.V.R.S., Chitteti, C., Yesupadam, S., Desanamukula, V.S., Vellela, S.S., Bommagani, N.J. (2023). Enhanced speckle noise reduction in breast cancer ultrasound imagery using a hybrid deep learning model. Ingénierie des Systèmesd'Information, Vol. 28, No. 4, pp. 1063-1071. https://doi.org/10.18280/isi.280426

[18] D, Roja and Dalavai, Lavanya and Javvadi, Sravanthi and Sk, KhaderBasha and Vellela, SaiSrinivas and B, Venkateswara Reddy and Vullam, Nagagopiraju, Computerised Image Processing and Pattern Recognition by Using Machine Algorithms (April 10, 2023). TIJER International Research Journal, Volume 10 Issue 4, April 2023, Available at SSRN: https://ssrn.com/abstract=4428667

[19] Vellela, S.S., Balamanigandan, R. An intelligent sleep-awake energy management system for wireless sensor network. Peer-to-Peer Netw. Appl. (2023). https://doi.org/10.1007/s12083-023-01558-x

[20] Rao, D. M. V., Vellela, S. S., Sk, K. B., &Dalavai, L. (2023). Stematic Review on Software Application Under-distributed Denial of Service Attacks for Group Website. DogoRangsang Research Journal, UGC Care Group I Journal, 13.

[21] S. S. Priya, S. SrinivasVellela, V. R. B, S. Javvadi, K. B. Sk and R. D, "Design And Implementation of An Integrated IOT Blockchain Framework for Drone Communication," 2023 3rd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2023, pp. 1-5, doi: 10.1109/CONIT59222.2023.10205659.

[22] N. Vullam, K. Yakubreddy, S. S. Vellela, K. BashaSk, V. R. B and S. SanthiPriya, "Prediction And Analysis Using A Hybrid Model For Stock Market," 2023 3rd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2023, pp. 1-5, doi: 10.1109/CONIT59222.2023.10205638.

[23] K. K. Kumar, S. G. B. Kumar, S. G. R. Rao and S. S. J. Sydulu, "Safe and high secured ranked keyword searchover an outsourced cloud data," 2017 International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, India, 2017, pp. 20-25, doi: 10.1109/ICICI.2017.8365348.

[24] Sk, K. B., Vellela, S. S., Yakubreddy, K., &Rao, M. V. (2023). Novel and Secure Protocol for Trusted Wireless Ad-hoc Network Creation. KhaderBashaSk, Venkateswara Reddy B, SaiSrinivasVellela, KancharakuntYakub Reddy, M VenkateswaraRao, Novel and Secure Protocol for Trusted Wireless Ad-hoc Network Creation, 10(3).

[25] Vellela, S. S., & Krishna, A. M. (2020). On Board Artificial Intelligence With Service Aggregation for Edge Computing in Industrial Applications. Journal of Critical Reviews, 7(07).

[26] Venkateswara Reddy, B., Vellela, S. S., Sk, K. B., Roja, D., Yakubreddy, K., &Rao, M. V. Conceptual Hierarchies for Efficient Query Results Navigation. International Journal of All Research Education and Scientific Methods (IJARESM), ISSN, 2455-6211.

[27] Sk, K. B., &Vellela, S. S. (2019). Diamond Search by Using Block Matching Algorithm. DIAMOND SEARCH BY USING BLOCK MATCHING ALGORITHM. International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN, 2349-5162.

[28] Vellela, S. S., Sk, K. B., Dalavai, L., Javvadi, S., &Rao, D. M. V. (2023). Introducing the Nano Cars Into the Robotics for the Realistic Movements. International Journal of Progressive Research in Engineering Management and Science (IJPREMS) Vol, 3, 235-240.

[29] Kumar, K. & Babu, B. & Rekha, Y.. (2015). Leverage your data efficiently: Following new trends of information and data security. International Journal of Applied Engineering Research. 10. 33415-33418.

[30] S. S. Vellela, V. L. Reddy, R. D, G. R. Rao, K. B. Sk and K. K. Kumar, "A Cloud-Based Smart IoT Platform for Personalized Healthcare Data Gathering and Monitoring System," 2023 3rd Asian Conference on Innovation in Technology (ASIANCON), Ravet IN, India, 2023, pp. 1-5, doi: 10.1109/ASIANCON58793.2023.10270407.