IJIEMR Transactions, online available on 15th Nov 2021. Link

:http://www.ijiemr.org/ downloads.php?vol=Volume-I0&issue=ISSUE-11

DOI: 10.48047/IJIEMR/VI0/111/13

Title: IoT and Smart dust Architecture Integration for Network Optimization using Artificial Neural Network

Volume 10, Issue 11, Pages: 84-92

Paper Authors

Jayant kumar Dorave and Dr. Ritesh Sadiwala

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

IoT and Smart dust Architecture Integration for Network Optimization using Artificial Neural Network

[1]Jayantkumar Dorave and [2] Dr. Ritesh Sadiwala

[1]Research Scholar, Department of ECE, R. K. D. F. University, Bhopal, INDIA

[2]Associate Professor, Department of ECE, R. K. D. F. University, Bhopal, INDIA

Abstract:

IoT devices are playing a greater role in business specially in wireless communication. IoT devices are achieving higher maturity as seen in smartdust. The aim of this research is to study the functionality of MOTES in smartdust to integrate with IoT architecture and infrastructure for optimization of wireless communication specially linked with 2.4Ghz and 5Ghz band. MOTES are being modeled in MALTAB using Artificial Neural Network integrated with optimization for speed, power and frequency linked with IoT architecture. The result proves that smartdust architecture if utilized in IoT architecture, the over all performances result of IoT devices is increased specially in bandwidth and power consumption. All the modeling result were compared for general sensor data bandwidth in ESP8266 for 2.4 Ghz, and mathematical model are presented for 5Ghz using smartdust MOTES. It is been proposed that using AI optimization technique like Ant Colonization Optimization or Particle Swarm Optimization we can mathematically model smartdust Architecture.

Key Words:- *IoT, Smartdust, MOTES*

Introduction

Smart dust is a little gadget with supernatural abilities. Smart dust integrates detecting, computation, wireless communication, and self-powering capabilities in a dimension of only a few millimetres, all at a cheap cost. Such gadgets are designed to be so tiny and light that they can float around in the surroundings like a regular dust particle. Smart Dust's qualities will make it valuable for observing real-world phenomena while causing minimal disruption to the underlying procedure. Smart Dust can now be made in a 5mm cube size, however we anticipate that it will soon be as tiny as a package of dust. Due to their microscopic size, smart dust sensors are mounted alluded to it as motes. The fast integration and convergence of three fundamental technological

innovations: digital circuits, communication systems which are wireless, and Micro Electro Mechanical Systems has made this possible (MEMS). Developments in hardware science and engineering design have resulted in size, electricity consumption, and efficiency improvements in each of these areas. This has allowed for the creation of incredibly small, self-contained nodes with one or even more sensors, processing and transmission capabilities, and a power source.

Micro Electro Mechanical Systems (MEMS) production enables compact, low-cost, increased performance sensors and actuators, making MEMS an enabling technological innovation for the Internet of Things (IoT). This section provides an overview at some of the most helpful and beneficial characteristics of MEMS technology for IoT. MEMS technology that has the potential to propel IoT devices to new heights. Human senses have developed to be fairly effective at certain functional capacities, but in comparison to MEMS sensing technologies, they are severely constrained.

Whereas the Internet of Things is still taking shape, its impacts have already begun to make significant progress as a global alternative media for the linked world. The construction of associated fields is always paved by specialized architectural studies. Researchers are now struggling to get through to the scope of Internet of Things-centric techniques leading to a shortage of comprehensive architectural expertise. This literature review examines Internet of Things-oriented designs that are interested in improving programmer comprehension of relevant tool, technologies, and technique.

The designs discussed attempt to tackle real-world issues by constructing and deploying strong Internet of Things concepts, either explicitly or implicitly.
Inventory control, smart corporate offices, fingertip inertial sensors virtual computer monitors, defence networks that could be deployed quickly by unmanned aerial vehicles (UAV), nodes something which track the movement patterns of birds, small animals, and now even insects, as well as environmental infrastructures which also supervise farming and livestock conditions are just a few examples.
The Internet of Things (IoT), Wireless Sensor Networks (WSNs), Ubiquitous Sensor Networks (USNs), Machine to Machine (M2M), and the Internet of Everything (IoE) have all developed into Cyber Physical Systems (CPSs), which were recently formed [1]. These technologies are collectively known as the Internet of Things (IoT), and they make use of wireless telecommunication technologies to link individuals to things and gadgets to each other in order to deliver smart technology and products [2].
Sensor network dynamic configuration will allow the use of the field gateway and sensor data transfer through IoT devices in almost real-time. This work presents an open smart dust architecture of the IoT virtualized platform integrated with sensor networks, IoT sensors, smart dust and cloud computing technologies using Artificial Neural Network [3].Because of the product's intricacy and variety, several alternatives and possibilities for platform connectivity must be defined. Artificially intelligent techniques may be used at the IoT and Cloud computing level to develop better computations independent of the network system.

The Smart Dust Architecture: -

Figure 1 shows a conceptual architecture diagram of a Smart Dust mote.

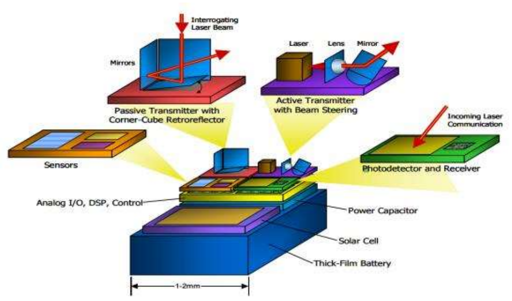

Interrogating Laser Beam
Mirrors
Laser
Lens
Mirror
Passive Transmitter with Corner-Cube Retroreflector
Active Transmitter with Beam Steering
Incoming Laser Communication
Sensors
Photodetector and Receiver
Analog I/O, DSP, Control
Power Capacitor
Solar Cell
Thick-Film Battery
1-2mm

Figure 1.:- Showcasing the Smart-Dust Architecture with different MEMS

As previously stated, a good smartdust system would include the following characteristics:

1. A high level of energy density

2. A high ratio of active volume to packaging volume

3. Low cell potential (0.5-1.0 V) to allow digital circuits to operate.

4. Compactness of sensors and its data flows

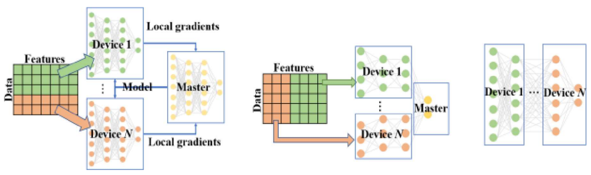
5. Less consumption of power

6. Functional with low data bandwidth

Researchers were able to compress several different types of sensors into small sizes while retaining or even exceeding the performance of the original transducers [4, 5]. Due to the obvious reductions in parasitic capacitance and space wasted to pads, homogeneously merging sensors altogether or with other equipment such as electronics might be desirable, although putting them on disparate chips can be useful for stacking and packing considerations of micro fabricated architectures.

The Internet Of Thing (Iot) Architecture: -

Distributed detection, monitoring, and classification are important tasks in wireless sensor networks or the Internet of Things. Sensor nodes take measurements and transmit them to a base station/gateway, which then sends them to that of an database server in the Internet protocol, which can subsequently make a judgement or inference based on the nodes' measurements. Digital forensics to identify breaches or questionable occurrences, environmental monitoring to identify and record pollution problems, and structure health management to avoid system failures are just a few of the uses available.



(a) datasets that are spread (b) characteristics that are spread c) models that are spread

- Figure 1.2 As per state of the art, the three most prominent frameworks for distributing machine learning to many devices are [9], [10], and [11].

If quick inference is required and the data is multidimensional, the nodes may be required to communicate significant amounts of data towards the core network, which is bandwidth intensive and may be infeasible with current IoT networks and design systems. Furthermore, allowing the nodes to exchange the entire measurements is wasteful since they generally contain redundant and associated data. The amount of transmissions which can be supplied and associated information bits will be severely constrained unless we minimise the amount of data transferred.

Some standard approaches for Iot network, including such global source coding [6], distribution estimate [7], and widespread detection [8,] might be utilised to decrease the quantity of data to communicate. Nevertheless, these strategies often need a thorough understanding of underlying relationship model between associated with a particular at various sensors. Consolidated source coding as well as distribution detection, for instance, rely upon node correlations, and distribution estimation need understanding of the assessment model and system status, that may be difficult to get in complicated systems.
Deep Learning (DL), which is now the highly advanced in machine learning (ML) and is extensively employed in multiple surveillance applications using IoT [12], [13], is an alternatives to the model-based method discussed above. The collect-then-infer method of DL somehow doesn't expand well with growing numbers of nodes and applications in IoT systems, nevertheless, because transferring enormous amounts of high-dimensional original information from vast IoT nodes would overcrowd the IoT network (massive IoT). Distributed learning is a possible answer to the scalability and flexibility. In the IoT, there seem to be two hurdles to employing distributed machine learning. One seems to be IoT networks' restricted capacity, which necessitates reducing data transfer both in strength and conditioning and interpretation stages. The other one is the IoT nodes' limited processing, storage, and thermodynamic efficiency, which necessitates tailoring the ML model just at IoT nodes to actual capabilities.

ANALYSIS OF IEEE 802.11 WLANS FOR IOT COMMUNICATIONS:-

The AP on a Wi-Fi equipment can operate as a Network interface, forming a star topology. Wi-Fi has a higher output power than previous wireless local area network implementations. For Wi-Fi networks, comprehensive insurance of Broadband internet is required, therefore dead spots are avoided by using many antennas in the access point. The 2.4 and 5 GHz frequencies are used for Wi-Fi. Its 5 GHz functioning allows it to utilise more bandwidth and deliver better data speeds. Indoors (e.g., within buildings), nevertheless, the bandwidth of a 5 GHz transmitter is smaller than that of a 2.4 GHz transmitter. IEEE 802.11b and IEEE 802.11g use the 2.4 GHz ISM frequency to communicate. Numerous inputs and various output techniques (MIMO) are included in IEEE 802.11n, which improves on prior versions of the standard [14]. It has a high bandwidth range of 54 to 600 megabits per second [15]. IEEE 802.11ac is indeed an upgraded version of IEEE 802.11n that gradual implementation wireless personal area networks (WLANs) in the 5 GHz band with more system platforms and greater transmission with MIMO, leading to data rates of more to 433.33 Mbps [16].

Overview Of Dl Frameworks

In networks with multiple nodes, there are three major strategies to distribute ML jobs (see Fig. 1.2): (a) to distribute data samples among the nodes, (b) to distribute distinct data characteristics among the nodes, and (c) to distribute the model among the nodes. We'll take a look at these techniques today. We depict the data as a matrix in Fig. 1.2 for demonstration purposes, with rows representing samples and columns representing characteristics. The fundamental distinction between the three frameworks is how the distributed nodes learn an ANN. Our major goal is to overcome the limitations of the IoT framework by incorporating the benefits of the Smart-dust framework into a machine learning network for positive outcomes.

A. Learning with distributed datasets

Developing a unified theory with dispersed datasets necessitates training an unified framework with many nodes, each one with local data points in the same features domain. Cellphones, for example, might use their personal portfolios to train and improve the ML model for photo classification, as well as communicate some model information with one another to improve the model's accuracy. It minimises the amount of raw data sent between nodes, which is particularly important during the learning phase. As a result, this architecture is particularly suited to situations where deductions required just information from a single point and several nodes are using the same comprehensive model.

Worker nodes and a master are the most popular distributed structure. Federated Learning [17], [20], a promising example of the framework, tries to achieve a universal ML model for all employees in a collaborative manner without the requirement to acquire raw data from them all at once. Before sending information back to the master node, each worker node obtains a global model and changes it with local data. The master node then updates the global model by averaging the local models. Because each node has a complete ML model on hand, all it takes to draw an inference is for the node to enter its measures into the ML model. For completely distributed ML (without a master node), where the worker nodes coordinate through communication networks with any topologies, similar algorithmic principles have been applied (see [10] and references therein). So will see that the design parameters are the only thing that various entities have in common. Because learning on a distributed dataset necessitates data instances with the same feature space on all nodes, it is not suitable for large-scale monitoring that employs various IoT nodes to gather data with varied characteristics. Similar distributed data-sets may be established for MOTES in smart-dust and modelled for IoT networks using the same parameter.

B. Learning with distributed features

FL is fundamentally distinct from learning utilizing dispersed features. When many nodes detect distinct aspects, it naturally occurs. In surveillance video networking, for illustration, each camera takes footage from several perspectives, resulting in diverse characteristics. The readings of various types of sensors at different places yield diverse aspects in large-scale environmental surveillance, such as forecasting the weather. The simplest approach to dealing with dispersed features seems to be to collect all of the data on a particular node, from which the ML model may be trained centrally. When the amount of the original information at each node is big, nevertheless, this is expensive. Distributed learning methods are required to solve such challenges. The disadvantage of IoT systems is that nodes cannot manage massive amounts of data; nevertheless, the system may be improved using the smart-dust architecture and decentralized intelligence.

In the literature, distributed learning algorithms that allow feature decomposition across nodes, such as those based on the dynamic diffusion approach [21] and the adaptive direction multiplier technique [22], have been examined. The nodes in these systems compute the inner product of the data and the model and exchange the findings with others. In terms of privacy, the work in [18] offered a features distribution machine learning (FDML) system in which each data owner performs ML locally using local information and communicates the local prediction to a master node. The master node computes the weighted total among these localized predictions and sends it to an activation function for final inference. In comparison, the suggested system comprises numerous layers at the master node to provide improved inference accuracy, with maximum data flow and minimal power utilisation. Furthermore, rather than making a local inference, the proposed MOTES in our proposed framework aim to condense the information.

C. Learning distributed models

This paradigm is concerned with how the model is distributed rather than how data is disseminated. Once the MOTES are setup, the model of MOTES or distributed nodes are fully concentrated The framework distributes the ML model among numerous nodes once the ML model is created on a single node. A node is particularly responsible for learning a subset of model parameters. Because ANN has numerous layers, the most common technique to distribute the model parameters is to divide the ANN into layers. More layers might be allocated to a node with superior storage and computing capabilities. Inference is performed progressively from one node to the next, with intermediate findings shared across nodes until the last node when final inference is performed.

A system[19] describes that translates an ANN onto heterogeneous nodes. During training, each node represents a layer of the ANN and conducts a local parameter update. This spreads the computational strain over numerous nodes, resulting in speedier training and inference.

Oursmart-dust framework uses the idea of learning with distributed features and the learning of distributed models, but we propose to greatly improve the communication efficiency of the ML in IoT networks by integrating the smart-dust architecture. Briefly speaking, on the IoT nodes optimization, it is a feature distributed framework such that each IoT node makes some preliminary compression and inference based on its local measurement with decision making ability with respect to node's dataflow bandwidth.

Proposed Communication Efficient ANN for IoT with Smart-Dust Framework

To allow ANN to grow with huge IoT, we need a new communication-efficient ANN framework/architecture. In this part, we will present our suggested IoT and Smart-Dust architectural framework for network optimization.

A. System architecture

Consider an IoT network of N distant IoT nodes connected to a cloud application server through base stations, as is characteristic of cellular IoT structures [23]. The system's purpose is to produce an inference based on a feature vector x of dimension d, which comprises of the feature vectors xi, xj of dimension di,dj recorded by nodes I and j. The nodes are utilised for a monitoring task that identifies events that correspond to one of the various classifications. ML is used to develop an inference rule that, given an input data measured by the nodes as created by the event, yields the true class to which the event belongs. We must remember that xj and dj are independent variables that rely on the configuration of the motes and their interactions with the nodes.

Each node sends some information ci to the base station in order to execute the inference. We may make ci = xi, which means that each node sends its whole feature vector. This, however, is inefficient. The size of ci should be as minimal as feasible to ensure optimal communication.Let ci = Ci(xi) be a compressed version of xi, where Ci() is a compressible function. compress function will be created for the smart-dust architecture parameter.

It's important to mention that we use the term compression to describe the process of lowering the amount of data from original data to information sent from a node. The procedure might include extraction, synthesis, coding, and other steps. When the IoT node is waiting for its transmission window, it can perform this compression process. The base stations and the cloud perform the inference T after collecting ci from all nodes as c. (c). The compression method minimises communication latency and consequently the total time of the interpretation by reducing the quantity of data to be transferred. Once The main challenge in creating this communication-efficient ANN is creating appropriate compression functions Ci() and Cj(.) as well as the inference function T(). We then show how we can build them as ANNs and then train the ANNs to get the function coefficients.

B. IoT as ANN

The ANN inference network and the IoT communication network both have a multi-layered structure.sigmoid.Thus, it is natural to think of IoT networks as ANNs, with IoT nodes representing the shallow layers of the entire ANN and responsible for extracting information to reduce the amount of data to transmit, and base stations and cloud representing the deep layers of the ANN and responsible for making inferences based on the information sent from the IoT nodes. As a result, we can use ML to jointly learn the compression functions Ci(.), Cj(.), and the inference function T(.). We create Ci(.) Cj(.) and T(.) to represent the layers structure of a DNN, including the types of layers and the number of neurons at each layer. The compression and inference functions are then used to create the model parameters for the DNN to be learned. A fully connected layer, for example, corresponds to a function (Wxin + b), where xin is the input of this layer. A fully connected layer, for instance, corresponds to the function (Wxin + b), where xin represents the layer's origin and end, W and b represent the gradient form of the effective to be learned, and () represents an activation function, such as a sigmoid function. Wwhere is predIf W is a fatty matrices, the result has less dimensions than the source, and the level relates to compressing; else, the layer projections a lesser input to a higher-dimensional vectors, that can be used for interpretation. Once the model parameters of each ANN layer have been trained, as discussed in Section III.C, the compression and inference functions are composed of multiple functions, each of which corresponds to a layer in the ANN. The entire system can be thought of as an ANN, with the nodes representing the shallow disjoint parts of the ANN and the base station and cloud representing the deep part.

We can then find the optimal functions Ci(.), Cj(.) and T(.) by minimizing a loss function that captures the inference accuracy and complexity of the functions Ci(), Cj(.) and T(.) using the training data collected by the IoT nodes. The compress and inference functions of the ANN may be done by layering the levels in the ANN after the model parameters have been trained. Layers include things like convolutional layers, pooling layers, fully linked layers, and recurrent layers. At the output, differentiated security may be employed to secure data privacy throughout the inference step. To reduce extraneous material, the summarising idea [24] might be employed. To cut down on the amount of bits transferred, compression might be applied at the nodes' terminals. Because we want a communication-efficient framework, the nodes should compress the information by reducing the size of their outputs, and the base station should expand the inputs at the first hidden layer to extract the compressed information.

We don't require deep ANNs on the nodes or ground station sides in our architecture because we focus on performance efficiency. Instead, we demand that the nodes' output be as minimal as possible without compromising inference efficiency.

C. Training of the model parameters

Because we design the compression and inference functions as layers of a DNN, the coefficients of these functions are the DNN's model parameters. We can use a centralized method to obtain the coefficients. A centralized cloud center, in particular, collects all training data and the corresponding desired output. Then, it centrally trains the parameters of the entire DNN and returns the models to the nodes and base station. During inference, the IoT nodes and base station use the trained model. More specifically, denote by $\bar{x}_I \in \mathrm{R}^{di\times m}$ the training data of IoT node I, where m is the sample size. Then, the training data samples of the system $\bar{x} \in \mathrm{R}^{di\times m}$ is achieved by stacking $\bar{x}_I$ for all i. We denote $\bar{y}$the corresponding output. Let $\mathbf{w} = \{\mathbf{w}_0,...,\mathbf{w}_N\}$ be the ML model parameters of the entire DNN, where $\mathbf{w}_0$ is the model parameters of $T(\cdot)$, and $\mathbf{w}_i,(i = 1,...,N)$ is the model parameters of $C_i(\cdot)$. Then, the loss function for the training can be written as

$$L(\bar{x},\bar{y};\mathbf{w}) = \sum_{i=0}^{N} R_i(Wi) + D(T(c_1\,(\bar{x}_1;w1)\,.....,C_N(\bar{x}_N;\,W_N);\,w_0),\bar{y})$$

where the first term R(.) is the regularisation that corresponds to the ML model's complexity and the second term D(.,.) is the loss term that corresponds to the inference's accuracy. The model w is learned by minimizing L(, ;w), which can be accomplished through iterative updating.

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\, \nabla_{w}\, L(\bar{x},\bar{y};\mathbf{w}),$$

where is a predefined learning step size When the training fish catches a fish, the central cloud sends wi to IoT node i. It only requires that the IoT nodes have enough storage capacity to store the model as well as the computational capacity to calculate ci based on its measurements and trained model.

Similarly

More specifically, denote by $\bar{x}_j \in \mathrm{R}^{di\times m}$ the training data of Smardust MOTES j, where m is the sample size. Then, the training data samples of the system $\bar{x} \in \mathrm{R}^{di\times m}$ is achieved by stacking $\bar{x}_j$ for all j. We denote $\bar{y}$the corresponding output. Let $\mathbf{w} = \{\mathbf{w}_0,...,\mathbf{w}_N\}$ be the ML model parameters of the entire DNN, where $\mathbf{w}_0$ is the model parameters of $T(\cdot)$, and $\mathbf{w}_j,(i = 1,...,N)$ is the model parameters of $C_j(\cdot)$. As an outcome, the training's error rate may be stated asin

$$L(\overline{x1},\overline{y1};\mathbf{w1}) = \sum_{j=0}^{N} R_j(Wj) + D(T(c_1\,(\bar{x}_1;w1)\,.....,C_N(\bar{x}_N;\,W_N);\,w_0),\bar{y})$$

where the first term $R(\cdot)$ is the regularization that corresponds to the complexity of the ML model, and the second term $D(\cdot,\cdot)$ is the loss term that corresponds to the accuracy of the inference. The model **w1** is learned by minimizing $L(\overline{x1},\overline{y1};\mathbf{w1})$, which can be done by iteratively updating

$$\mathbf{w1} \leftarrow \mathbf{w1} - \eta\, \nabla_{w1}\, L(\overline{x1},\overline{y1};\mathbf{w1}),$$

where η is a predefined step size of learning. The central cloud delivers wj to IoT motes j when the training fishes. It simply requires that the Smartdust motews have the storage capacity to hold the model and enough compute ability to calculate cj based on the measurements and trained model. We might alternatively train the ANN in a distributed fashion, in which the nodes and base station each train their own local parameters. The nodes and the base station, in particular, initially set their local settings. Each node generates its codes as the result of its compression function and delivers the codes ci = Ci(;wi) and cj = Cj(;wj) in relation to the base station using the model parameters and a batch of training data.

$$\nabla_w L \frac{\delta D}{\delta T}\frac{\delta T}{\delta w_0}|c_{1,\cdots c_N,\bar{y}} + \frac{\delta R_0}{\delta w_0}$$

and performs an update of the inference function parameters as w0 w0 – L After that, the gradients are delivered to the nodes. Based on the gradients received from the base station, each node calculates the gradients with regard to its local parameters.

$$\nabla_{wi}\, L = \frac{\delta L}{\delta C_i}|w_{0,\bar{y},c_i,(j\neq i)}|\frac{\delta c_i}{\delta w_i}|x_i + \frac{\delta R_i}{\delta w_i}$$

$$\nabla_{wj}\, L = \frac{\delta L}{\delta C_j}|w_{0,\bar{y}1,c_j,(j\neq i)}|\frac{\delta c_j}{\delta w_j}|x_j + \frac{\delta R_j}{\delta w_j}$$

where $\partial L/\partial C_i|_{w0}$ is calculated and transmitted by the base station The IoT node then updates the settings using w w– wi L. The updating of parameters continues until convergence is reached. During the training process, the nodes do not broadcast the raw training data or their local model parameters, ensuring data privacy. This networked approach implies that the Nodes have sufficient processing power and storage capabilities to compute the regional gradients and store the training data.

If the IoT nodes' local computation and storage capacity are insufficient for training, they can offload the task and training data to a trusted device, such as a private gateway or server, and retrieve the local ML model once the training is completed.

Application and Numerical Outcomes

In this part, we'll use simulations and smart-dust motes to test the performance of proposed networked ML framework for IoT node optimization. We assume that the data transfer is perfect, in the sense that the receiving and decoding data is equal to the data transmission, which can be achieved by error - correcting encoding and repetition. The whole simulation of the models where done in MATLAB, the numerical data was all randomly generated for test purpose. ESP8266 microcontroller was modeled and consider as the nodes or optimized MOTES in MATLAB

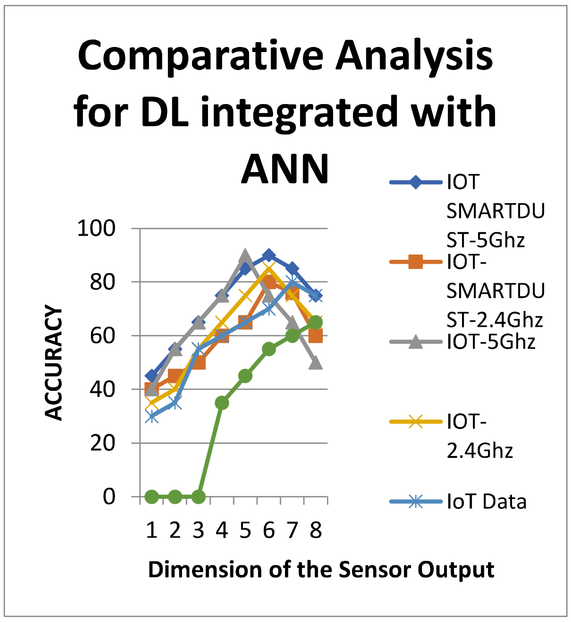


Fig. 3. Comparison of the classification accuracy achieved by our framework with different quantization's.

Summary And Future Directions

The aim of this research is to study the functionality of MOTES in Smartdust to integrate with IoT architecture and infrastructure for optimization of wireless communication specially linked with 2.4Gz and 5Ghz band. MOTES are modeled in MALTAB utilising an Artificial Neural Network with performance, energy, and frequency optimization coupled to an Iot ecosystem. The result proves that smartdust architecture if utilized in IoT architecture, the over all performances result of IoT devices is increased specially in bandwidth and power consumption. Modeling the Internet of Things as an efficient Artificial Neural combined with the Smart-Dust architecture variable is also suggested in this paper. We suggested a distributed ML framework that combines compression at the nodes and motes (FOR SMART-DUST) with reasoning at the ground stations, allowing nodes to send just a minimal quantity of data to the base station. When compared to the scheme that sends all raw data, the numerical results showed that our framework only loses 1.52 percent in inference accuracy while saving 91 percent in data transmission. Using the smart-dust framework, one can create extremely efficient IoT systems for monitoring and inference.
There are numerous interesting topics that should be researched further in order to improve system performance. One alternative is to look into the trade-

off between transmission speed and average accuracy. Is there a way for calculating the contact characteristics between such a collection of Nodes and Motes that is more efficient and systematic? Which layers have the potential to increase communication while minimising the impact on inference performance? It will be vital to develop the ML layer upon layer and having to learn in a way that the codes generated by the nodes and motes result in better inferential showings for numerous monitoring applications: this process is achieved using computer methods based such as Particle Swarm Optimization and Ant Colonization Enhancement.

References

[1] Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. Future Gener. Comput. Syst. 2013, 29, 1645–1660.
[2] Jun, Z.; Simplot-Ryl, D.; Bisdikian, C.; Mouftah, H.T. The internet of things. IEEE Commun. Mag. 2011, 49, 30–31.
[3]"CLOUD, FOG, DEW AND SMART DUST PLATFORM FOR ENVIRONMENTAL ANALYSIS",Rossitza GOLEVA1, Aleksandar SAVOV, Ivelin ANDREEV ,Rumen STAINOV, Jugoslav ACHKOSKI, Nikola KLETNIKOV,Igorche KARAFILOVSKI; The 13th Annual International Conference on Computer Science and Education in Computer Science, 30 June – 3 July 2017, Albena, Bulgaria
[4] Lj. Ristic [ed], Sensor Technology and Devices, Artech House, London, 1994.
[5] G.T.A. Kovacs, Micromachined Transduceers Sourcebook, WCB McGraw-Hill, San Francisco, 1998
[6] Z. Xiong, A. D. Liveris, and S. Cheng, “Distributed source coding for sensor networks,” IEEE Signal Process. Mag., vol. 21, no. 5, pp. 80–94, 2004.
[7]J.-J. Xiao, A. Ribeiro, Z.-Q. Luo, and G. B. Giannakis,“Distributed compression-estimation using wireless sensor
networks,” IEEE Signal Process. Mag., vol. 23, no. 4, pp. 27–41, 2006.
[8] J.-F. Chamberland and V. V. Veeravalli, “Wireless sensors in distributed detection applications,” IEEE Signal Process. Mag., vol. 24, no. 3, pp. 16–25, 2007.
[9] J. Konecnˇ y, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” https://arxiv.org/abs/1610.05492, 2016, [Accessed on 2019-12-01].
[10] Y. Hu, D. Niu, J. Yang, and S. Zhou, “FDML: A collaborative machine learning framework for distributed features,” in Proc. ACM SIGKDD, 2019, pp. 2232–2240.
[11] S. Teerapittayanon, B. McDanel, and H.-T. Kung, “Distributed deep neural networks over the cloud, the edge and end devices,” in Proc. IEEE ICDCS, 2017, pp. 328–339.
[12] A. Abeshu and N. Chilamkurti, “Deep learning: the frontier for distributed attack detection in fog-to-things computing,”
IEEE Comm. Mag., vol. 56, no. 2, pp. 169–175, 2018.
[13] X. Wang, X. Wang, and S. Mao, “RF sensing in the internet of things: A general deep learning framework,” IEEE Comm. Mag., vol. 56, no. 9, pp. 62–67, 2018.
[14] F. Rusek, D. Persson, B. K. Lau, E. G. Larsson, T. L. Marzetta, O. Edfors, et al., "Scaling up MIMO: Opportunities and challenges with very large arrays," IEEE Signal Processing Magazine, vol. 30, pp. 40-60, 2013.
[15] E. Perahia and R. Stacey, Next Generation Wireless LANs: 802.11 n and 802.11 ac: Cambridge university press, 2013.
[16] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey,"
Computer Networks, vol. 38, pp. 393-422, 2002
[17]J. Konecnˇ y, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for`
Improving communication efficiency,”
https://arxiv.org/abs/1610.05492, 2016, [Accessed on 2019-12-01].
[18]Y. Hu, D. Niu, J. Yang, and S. Zhou, “FDML: A collaborative machine learning framework for distributed features,” in Proc. ACM SIGKDD, 2019, pp. 2232–2240.
[19]S. Teerapittayanon, B. McDanel, and H.-T. Kung, “Distributed deep neural networks over the cloud, the edge and end
devices,” in Proc. IEEE ICDCS, 2017, pp. 328–339.
[20]H. B. McMahan, E. Moore, D. Ramage, S. Hampson, et al., “Communication-efficient learning of deep networks from decentralized data,” https://arxiv.org/abs/1602.05629, 2016, [Accessed on 2019-12-01]. optimization,” Proceedings of the IEEE, vol. 106, no. 5, pp. 953–976, 2018.
[21]B. Ying, K. Yuan, and A. H. Sayed, “Supervised learning under distributed features,” IEEE Trans. Signal Process., vol. 67, no. 4, pp. 977–992, 2018.
[22]Y. Hu, P. Liu, L. Kong, and D. Niu, “Learning privately over distributed features: An ADMM sharing approach,” https: //arxiv.org/abs/1907.07735, 2019, [Accessed on 2019-12-01].

[23]A. D. Zayas and P. Merino, "The 3GPP NB-IoT system architecture for the internet of things,"in Proc. IEEE ICC Workshops, 2017, pp. 277–282.
[24]K. Muhammad, T. Hussain, M. Tanveer, G. Sannino, and V. H. C. de Albuquerque, "Cost-effective video summarization using deep cnn with hierarchical weighted fusion for iot surveillance networks," IEEE Internet Things J., 2019,