2022 9th International Conference on Future Internet of Things and Cloud (FiCloud) | 978-1-6654-9350-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/FiCloud57274.2022.00035

# Improving IoT-based Smart Retrofit Model for Analog Water Meters using DL based Algorithm

A. Kumar Lall, A. Khandelwal, N. Nilesh, S. Chaudhari
*International Institute of Information Technology Hyderabad (IIIT-H), India*
Email: {ayush.lall, nitin.nilesh}@research.iiit.ac.in,
ansh.khandelwal@students.iiit.ac.in, sachin.chaudhari@iiit.ac.in

***Abstract*—This paper proposes a deep learning (DL)-based algorithm which is used for improving the performance of digit detection from internet-of-things (IoT)-based analog water meters. The DL algorithm is trained on a rich dataset of over 160,000 images collected from six water nodes deployed at locations with different environmental conditions. A detailed comparison between the proposed DL and machine learning (ML) algorithm is made based on detection accuracy, feature analysis, error analysis, and computational complexity analysis. It is observed that compared to the ML model, the proposed DL model maintained a higher detection accuracy and is more generalized in terms of feature extraction, which makes the algorithm robust.**

**Keywords– Data collection, IoT, ML and DL at edge, Smart water meter, Transfer learning**

## I. Introduction

The traditional analog water meter has a simple structure, minimal power dissipation, high durability, and high dependability. However, it still requires human effort to record meter readings manually, which is inconvenient. In order to counter this problem, there is a need for an automated system of data capturing, which can send the real-time detected meter readings to the cloud server. By analyzing the data collected, smart water meters can aid in understanding water consumption patterns and leakage detection for efficient water management. Thus it becomes essential to have a robust algorithm for meter reading detection.

There have been few works in literature for making analog water meters smart using digit recognition methods [1], [2]. In [1], gas meter readings were detected with the help of a convolution neural network (CNN) based on a Visual Geometry Group (VGG) architecture. An end-to-end performance of 85.71% was achieved. In [2], detection of meter readings was done with the help of Raspberry-Pi and deep learning (DL) model. Authors claim to get 96% accuracy using the You Only Look Once (YOLO)-v3 model tested on 100 images of the water meter. In [3], we implemented a retrofit model to digitize the analog water meter with the help of the Random Forest (RF) model. After applying a few post processing constrains, the accuracy obtained was about 97%. It was observed that the accuracy of the ML model was reduced on other water meters in the network compared to the meter used for collecting the training dataset. Ideally, the model should perform the same for all the meters as they are of the same make and have the same design and fonts. This reduction in performance can be attributed to varying illumination arising due to conditions at the deployment location. However, DL models which train on generalized features are not susceptible to such variations in the illumination.

In this paper, our specific contributions are

- A DL-based algorithm is proposed, which performs more accurately than the ML algorithm described in [3]. The proposed DL algorithm is based on CNN with transfer learning.
- A rich dataset is collected using images captured from six IoT-based smart water meters deployed in locations having different environmental conditions. For training, approximately 160,000 images have been used, which have been collected over 20 days. The testing data contains approximately 80,000 images collected over 10 days in real-time.
- A performance comparison against the proposed DL-based algorithm is carried out with the ML-based algorithm proposed in [3]. It is shown that the DL-based algorithm performs better than the ML-based algorithm even without the need for post-processing constraints defined in [3] to correct the false detections. Moreover, the proposed DL algorithm is computationally light enough to be deployed on our retrofit model (on edge).
- To understand the performance improvement of the proposed DL model over the ML-based model in [3], the extracted features from both the models are analyzed to infer what models are observing while detection.

Note the novelty of the current work as compared to [1], and [2]. In [1], the method implemented was not made to run at the IoT node, which becomes crucial as computationally complex models will not be able to run at the edge. While in [2], analysis was done on a smaller dataset, resulting in less variation. The dataset was not created with variations due to environmental factors, such as sunlight variations that can change the device's luminance, resulting in poorer detection.

The rest of the paper is organized as follows. In Section II, we go over the hardware description. Section III delves into dataset preparation, divided into three subsections, consisting of Pre-dataset collection, Training dataset, and Testing data, whereas Section IV details the suggested algorithm's methodology. In Section V, the results of the experiments and obser-

978-1-6654-9350-5/22/$31.00 ©2022 IEEE
DOI 10.1109/FiCloud57274.2022.00035

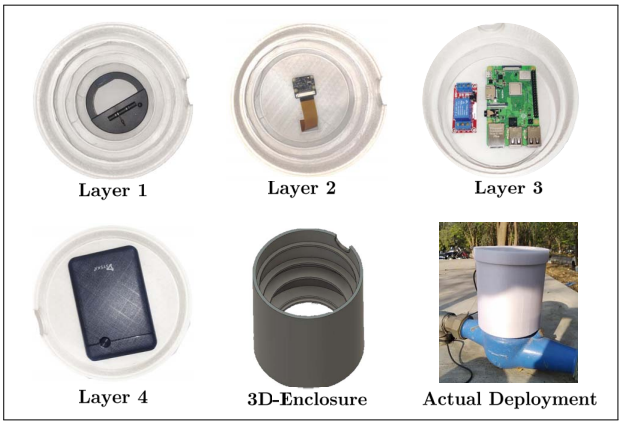



Fig. 1. The retrofit model is being deployed at an analog water meter on the campus [3].

vations are presented, which are further divided into four sections: Detection accuracy, Feature analysis, Error rate analysis, and Complexity Analysis. Section VI concludes the paper.

## II. Hardware Description

Fig. 1 shows the structure of the developed enclosure in [3]. It is a 3D-printed multi-layer structure made up of Poly-lactic Acid (PLA) material which offers protection against various weather conditions in outdoor deployment. It consists of a four-layered stack separating the various hardware components for comfortable placement. The first layer (bottom most) consists of an LED ring for providing adequate illumination to capture good quality images during nighttime. In the second layer, the camera module is placed facing the meter's dial. The Raspberry-Pi micro-controller is placed at the third layer. It is interfaced with an active high relay module using the Raspberry-Pi GPIO pins to control LED switching. At the fourth layer (topmost), the power bank is placed. The whole setup is mounted on top of the analog water meter without altering or tempering the analog meter or deployed water pipelines in any sense. The Raspberry-Pi detects the required water meter readings and transmits them in real-time to *ThingSpeak* using an LTE-based portable WiFi hotspot [4]. The setup can sense the reading at a high frequency, even when the water flow is at a peak. Data collected from the device can be used to derive valuable insights into the consumption patterns and timely detection of leaks.

## III. Dataset

This section is divided into three parts. The first part demonstrates the variability in the meter images based on the environmental factors and the need of collecting data from nodes deployed in different locations. The second part presents the dataset collected to train both (ML [3] and the proposed DL) models. Finally, the last subsection will discuss the dataset on which the proposed DL and ML model was tested in real time.

### A. Pre-dataset collection

In [3] a dataset was created using only one deployed water meter and the ML model showed the best digit detection accuracy on that deployed node. However, it failed to maintain similar detection accuracy on the other water meters of the same model deployed at different locations. This is due to environmental factors like sunlight, moisture, etc. For instance, one of our water meters was present inside a small warehouse, where it was exposed to minimal sunlight and external environmental factors. In contrast, one water meter was present in the rooftop of a building, where it was exposed to direct sunlight and other external factors. These factors will influence the illumination in the retrofit model while capturing the image. To demonstrate this, histogram of captured images' pixels is plotted for all the six nodes in Fig. 2. For this, 50 images were randomly chosen from each node, and their region of interest (RoI) was extracted on which the average histogram of pixels was plotted. The observations were intuitive as the average histogram of pixels differed for all the nodes, as shown in Fig. 2, which helped us understand how the environmental factor can influence the data collected from the IoT devices. Based on the above observations, data was collected from all the six water meters for a month at the rate of 1 image per minute.

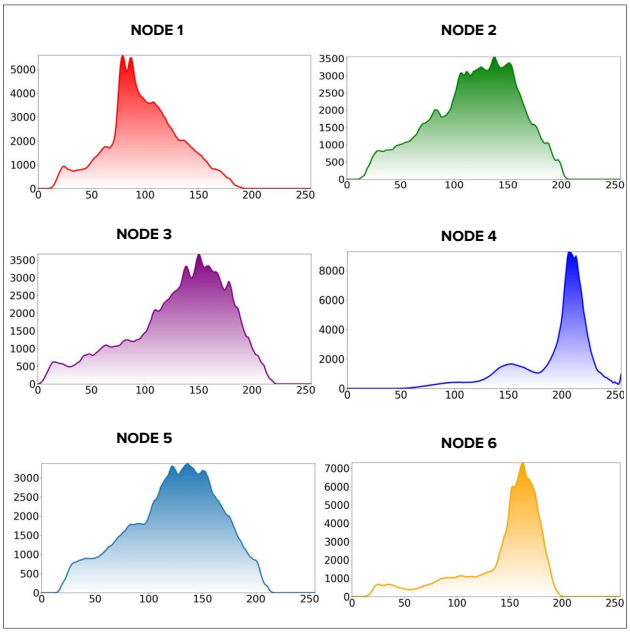



Fig. 2. Histograms of pixels for the images of respective nodes are displayed here. About 50 images were chosen randomly from each node, and their RoI was extracted on which the average histogram of pixels was plotted. The x-axis represents the pixel intensity values in the plots, and the y-axis represents the number of pixels having that particular intensity value. (Best viewed in color)

TABLE I
FREQUENCY OF THE INDIVIDUAL DIGIT IN THE TRAINING DATASET AFTER PREPROCESSING

| Digit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Node 1 | 274 | 75 | 27 | 42 | 142 | 195 | 339 | 57 | 33 | 103 | 1287 |
| Node 2 | 254 | 505 | 208 | 329 | 386 | 658 | 451 | 382 | 436 | 481 | 4090 |
| Node 3 | 497 | 551 | 588 | 226 | 434 | 743 | 472 | 403 | 524 | 726 | 5164 |
| Node 4 | 1903 | 1336 | 1491 | 1871 | 1325 | 570 | 1002 | 1845 | 1035 | 703 | 13081 |
| Node 5 | 103 | 134 | 125 | 76 | 150 | 130 | 103 | 144 | 134 | 131 | 1230 |
| Node 6 | 151 | 31 | 116 | 154 | 125 | 156 | 131 | 103 | 248 | 215 | 1430 |
| Total | 3182 | 2632 | 2555 | 2698 | 2562 | 2452 | 2498 | 2934 | 2410 | 2359 | 26282 |

### B. Training dataset

The data collected in first twenty days from all six nodes (approximately 160,000 images) was used to create the dataset. The captured images were transformed by manually inputting the selected coordinates, width, and height to get the transformation matrix. This matrix was used to apply warp perspective to get the RoI. After RoI detection, the digits were extracted by segmentation. These digits range from 0 to 9 and are saved in different folders to make a labeled dataset for supervised learning.

An ideal dataset should have all data points which are independent and identically distributed to decrease bias in the dataset. It was noticed that only the right-most digit (see water meter image in Fig. 3) was used to change for the time-series data, and the other digits moved seldomly due to the low water flow rate. This resulted in duplication of digits in the dataset. To remove duplicate images, images were first translated into tensors made by combining the respective RGB pixels and were resized to a pixel size of 50 to keep the comparison process computationally less expensive. These resized tensors of the two images were compared using mean square error (MSE). The lower the MSE, closer (similar) the images are to each other. Ideally, two images with MSE equal to 0 should be duplicates; however, due to resizing the images to 50 pixels, there is a slight variation in the original image. Therefore, we set a maximum MSE threshold of 20 (chosen based on trial and error) to find and remove the duplicate images.

Table I gives a statistical review of the amount of individual digit data collected from all the nodes. It can be observed that the frequency of the digits collected is not the same for all the nodes. The reason for this is that water flow through different meters differs depending on the water usage in corresponding regions. Because of this, more water flows through some meters than others. In some meters, water may rarely flow resulting in large number of duplicate images, which are getting removed. Therefore, we got different numbers of digit images from the six nodes.

### C. Testing data

Both the models (ML model [3] and the proposed DL model) were tested in real-time using the data collected in the last ten days (approximately 80,000 images in total) from all the six nodes. Using this data, both the models were compared based on various factors which will be discussed in the results section.

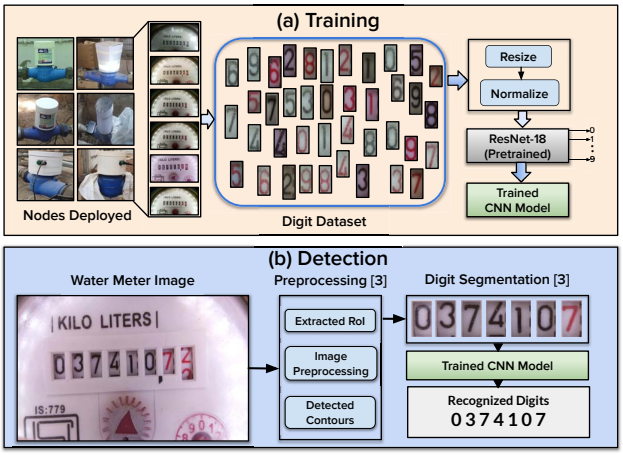



Fig. 3. Algorithmic pipeline for the DL methodology proposed. The dataset collected from different nodes mentioned in section III is used to train the ResNet-18 model (last layer having 10 nodes) using transfer learning mechanism. At the detection phase, preprocessing used in [3] was used to segment the digits from water meter image. Finally the digits are recognized using the trained CNN model. (Best viewed in color)

## IV. METHODOLOGY OF PROPOSED DL ALGORITHM

### A. Training

The problem at hand is defined as follows: Given a water meter image, extract the digit images and recognize the digits present. To solve the problem, the training dataset defined in the section III is used. As this is a classification-based supervised learning problem, DL based CNN algorithm is proposed to train and classify the digit images. In recent years, CNN-based algorithms have been extensively used in object detection and outperform image-processing-based feature extracted ML models. In our case of water meter digit recognition, usage of CNN based model is even more favorable due to the following conditions:

1) The trained CNN model extracts a rich feature set: scale, illumination, rotation, and translation invariant. As the water meter digit images contain all these variations in real-time, the CNN features-based trained model is generalized and more robust than image-processing based extracted features.
2) DL models like CNN can leverage the capability of transfer learning which allows the knowledge transferability of pretrained models on a huge dataset for similar tasks. Using this mechanism in the case of water meter images helps faster and more accurate training.

The training procedure of the CNN model is as follows:

**CNN Model Description**: Although there are many CNN models available for the object detection task, our use case required a CNN model with less computational complexity and yet is accurate enough to perform the digit recognition task.

The main reason for being computational complexity aware model is to perform the inference on the hardware node itself defined in section II. Following [5], a popular CNN model ResNet-18 [6] was a suitable choice in terms of computational complexity and accuracy.

As evident from the name, ResNet-18 is 18 layers deep network, and the model overall contains almost 11 million trainable parameters. Training the Resnet-18 model, where the weights are randomly initialized, directly on the digit dataset is an extremely challenging task. To solve this problem, the transfer learning mechanism is used. [6] has trained ResNet-18 model on a huge dataset named ImageNet [7] which contains around 14 million images ranging 20000 categories. We used this pretrained model to further train on the digit dataset as mentioned in section III-B. The final fully connected layer of the pretrained ResNet-18 model was replaced by ten neurons to match the number of outputs of digit recognition.

**Preprocessing**: As the ResNet-18 model input is of fixed size, all the digit images are first resized to a fixed resolution of $224 \times 224$ to match the compatibility of the pretrained ResNet-18 model. Secondly, the standard normalization (zero mean and unit standard deviation) is applied to the input image. This step ensures that all the inputs to the CNN model follow a similar data distribution and helps converge faster while training the network.

**Training**: The dataset was split into training and validation sets with 90% and 10% randomly selected data points, respectively. This step helps generalize the model as it is only trained on the training dataset and tested on the validation dataset. While training the model, instead of only training the last fully connected layer, all the 18 layers of the ResNet-18 model were finetuned, which helped achieve better accuracy. The model was trained for 5 epochs where it hit the convergence. The choice of *optimizer* was *Adam* with a learning rate of $0.001$ scheduled using a linear scheduler having *gamma* set to $0.1$. For all the DL implementation, popular python-based framework *Pytorch* [8] was used.

### B. Detection

To detect given water meter image, detection methods till segmentation (RoI extraction, image preprocessing, and digit image segmentation) defined in the detection section of [3] is used. After obtaining the segmented digit images, the preprocessing step similar to training (image normalization and resize) was applied to the digit images. Finally, each of the digit image is passed through the trained CNN model to recognize the digit present between 0 to 9.

## V. Results

This section is divided into four subsections. First subsection compares the detection accuracy of both the ML model in [3] and the proposed DL model (CNN with transfer learning) on the testing data as mentioned in dataset section III. Note that ML model in [3] is the RF model with postprocessing techniques. Second subsection comprises of analysis of features extracted by the models. The third subsection does a detailed error analysis of both the models. Finally, the last subsection presents the comparison of the models based on computation complexity.

### A. Detection accuracy

Table II presents the detection accuracy of the trained models on the testing data. The proposed DL model has more than 99% detection accuracy for all the nodes, whereas the detection accuracy of the ML model fluctuates from 91% to 97% depending on the node. It is essential to notice that the ML model gets this accuracy after some post-processing constraints as described in [3]. However, the proposed DL model does not require any constraints to maintain its high accuracy of more than 99%. This, in turn, shows that the proposed DL model is more generalized in terms of feature extraction and detection. Thus, it can be used with multiple nodes deployed nodes in different environmental conditions. Figs. 4(a) and 4(b) show the detected meter readings by both the RF model (with constraints) and proposed DL model of nodes 1 and 2, respectively. The time axis is set up so that every tick corresponds to the start of the day at 0000 hrs. As a result, the interval between two consecutive ticks is 24 hours. There is an increase in the meter reading records during every pumping event. For the period that the pump was not on the meter reading stayed steady. It can be observed from the plots that the ML model has more false detections than the proposed DL model. From both plots, the zoomed parts highlight the wrong detections by the ML model. The red spikes in the plots denote the wrong detections. The error analysis is done later in this section to show where the ML model fails to detect digits.

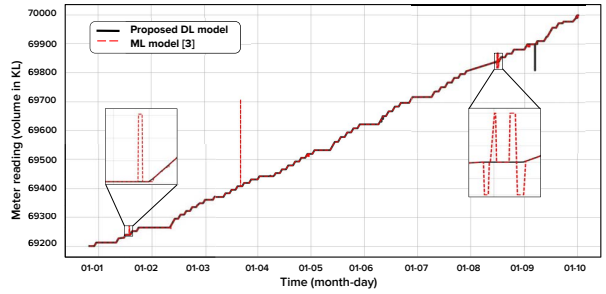
The accuracy of proposed DL model is above 99%; at very rare instance like in Fig. 4(a) at 0300 hrs on 9th January an error was observed. Here the CNN algorithm detected a tens digit 9 as 0. However, it is worth observing that the proposed DL model performs better than the ML model (which requires post-processing) with consistently higher accuracy on different nodes. The other four nodes also had a similar types of graphs but not shown here for brevity.

### B. Feature Analysis

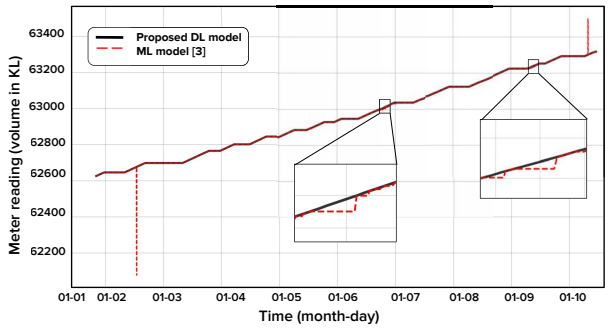
A comparison has been made between the features extracted by the ML model (RF-based model) and the proposed DL model (CNN with transfer learning). The RF model trains on HOG-based features. To analyze the HOG extracted features, the method proposed by [9] named HOGgle is used. The HOG extracted features are inverted into the corresponding natural image to understand at a human level using this method. This also helps to understand what exact features a HOG-based learning algorithm (RF model in this case) has been trained

TABLE II
Accuracy in percentage for both models

| | Node 1 | Node 2 | Node 3 | Node 4 | Node 5 | Node 6 |
|---|---|---|---|---|---|---|
| ML model [3] | 96.2 | 93.4 | 94.5 | 91.2 | 95.4 | 97.2 |
| Proposed DL Model | 99.2 | 99.5 | 99.4 | 99.3 | 99.6 | 99.2 |




(a) Node 1




(b) Node 2

Fig. 4. Meter reading (volume of water flown) w.r.t time, assuming 0000 hrs as the start of the day on time axis

on. The incorrectly detected digits by the ML model, trained on HOG features, have been visualized. On the other hand, to analyze the CNN extracted features, the method proposed by [10] is used. This method uses features extracted by a trained CNN model to reconstruct their corresponding original image. It has been claimed that features extracted by a highly accurate CNN model preserve photographically accurate information about the image and can be used to reconstruct their original image. In our case, the CNN model's first, third, and fifth convolution layer outputs are used to reconstruct the digit image from a randomly generated noisy image.

In Fig. 5, it can be observed what exactly leads the ML model trained on HOG features to make mistakes while detecting certain digits. For example, in the case of Fig. 5(a), it can be seen that the ML model is misclassifying the digit 0 as 9. It was also observed that most errors occurred for the digits where some part of other digits had a shape influence. In contrast, the CNN model was able to detect it correctly. In order to understand the reason behind it, we have analyzed the digit HOG features by inverting them following the HOGgle method. It can be seen that the inverted HOG image resembles the shape of digit 9. While there were certain cases when the CNN model misclassified the digit images, the analysis of CNN extracted features using [10] was performed to understand the reason behind it. For example, in the case of Fig. 5(c), the digit encountered was a particular case where half of the previous digit and half of the upcoming digit ('half digit') were present due to the rotating-disc type meters. In

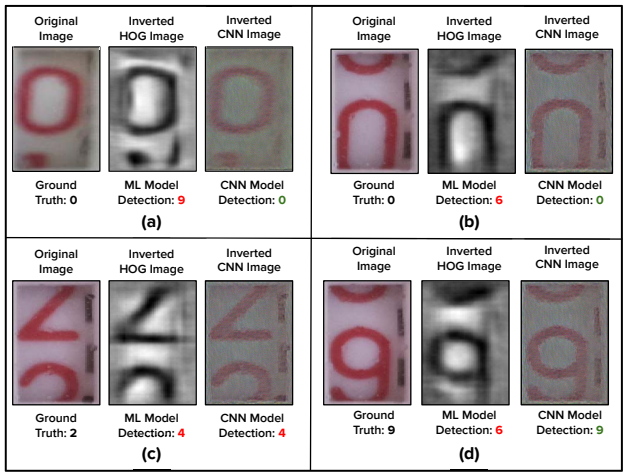



Fig. 5. Analysis of features extracted between HOG and CNN model. (Best viewed in color)

this case, both the ML and DL model misclassified the digit image. After analysis of both features (HOG-based and CNN extracted) for this particular image, it can be observed that both resemble the shape of digit 4. This was one of the primary motivations to discard the last digit of the water meter image (i.e., the least significant digit) at the detection stage. The frequency of encountering such kind of 'half digits' at the last position increases significantly compared to the rest of the digit's position and may lead to error-prone detection.

For all other digit locations, the CNN model's overall error rate was significantly less even when the water meter node's location differed. On the other hand, in the case of the ML model, the same cannot be said. The primary reason behind this can be inferred by the difference in richness of features extracted between both the methods. It can also be claimed that features extracted by the CNN model are more generalized than the HoG-based features.

### C. Error rate analysis

In the error analysis, we use the same performance parameters as in [3], i.e., digit error rate (DER), value error rate (VER), and root mean square error (RMSE). Table III shows the DER for all nodes on the digits that were detected using both the models. It can be observed that the proposed DL model has a very low DER values, whereas, ML model has a higher DER values for most of the digits. This again proves that the proposed DL model has more generalized features than the HOG features-based RF model. Along with DER, we also found the VER and RMSE for both the models given in Table IV. The maximum VER for RF model came out as 8.61% and for proposed DL model 0.62%. While the maximum RMSE for the ML model came out as 0.351 KL, and for the proposed DL model was 0.06 KL.

### D. Complexity analysis

Table V shows the computational complexity analysis for both the models in terms of three main parameters of computa-

TABLE III
DER (IN PERCENTAGE) OF BOTH THE ALGORITHMS ON THE NODES

| Digit | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Node 1 | ML model [3] | 6.43 | 3.05 | 0.0 | 5.0 | 0.2 | 3.07 | 0.2 | 0.5 | 6.1 | 1.05 |
| | Proposed DL model | 0.0 | 0.0 | 0.0 | 0.4 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 |
| Node 2 | ML model [3] | 0.0 | 0.78 | 0.0 | 1.41 | 0.19 | 0.65 | 0.15 | 1.47 | 0.5 | 0.0 |
| | Proposed DL model | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.2 | 0.0 |
| Node 3 | ML model [3] | 13.1 | 4.85 | 0.5 | 2.43 | 1.03 | 6.93 | 0.24 | 1.87 | 17.28 | 1.65 |
| | Proposed DL model | 0.4 | 0.0 | 0.0 | 1.221 | 0.0 | 0.0 | 0.12 | 0.0 | 0.0 | 0.0 |
| Node 4 | ML model [3] | 7.69 | 6.66 | 1.73 | 8.22 | 3.46 | 3.33 | 1.73 | 12.21 | 8.82 | 3.09 |
| | Proposed DL model | 0.0 | 0.0 | 0.86 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.13 |
| Node 5 | ML model [3] | 11.90 | 0.2 | 0.0 | 10.25 | 6.55 | 7.79 | 0.18 | 2.16 | 9.19 | 0.82 |
| | Proposed DL model | 0.0 | 0.0 | 0.0 | 2.5 | 0.0 | 0.0 | 0.09 | 0.0 | 0.0 | 0.0 |
| Node 6 | ML model [3] | 0.8 | 0.0 | 1.85 | 0.15 | 2.65 | 0.38 | 0.31 | 0.0 | 0.0 | 0.0 |
| | Proposed DL model | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.0 | 0.0 | 0.0 |

TABLE IV
VER AND RMSE OF BOTH THE ALGORITHMS ON THE NODES

| Node | | ML model [3] | Proposed DL model |
|---|---|---|---|
| Node 1 | VER (%) | 5.53 | 0.44 |
| | RMSE (KL) | 0.157 | 0.06 |
| Node 2 | VER (%) | 1.81 | 0.2 |
| | RMSE (KL) | 0.038 | 0.004 |
| Node 3 | VER (%) | 7.83 | 0.6 |
| | RMSE (KL) | 0.174 | 0.027 |
| Node 4 | VER (%) | 8.61 | 0.62 |
| | RMSE (KL) | 0.319 | 0.024 |
| Node 5 | VER (%) | 5.61 | 0.4 |
| | RMSE (KL) | 0.351 | 0.016 |
| Node 6 | VER (%) | 1.81 | 0.4 |
| | RMSE (KL) | 0.161 | 0.019 |

tional complexity: execution memory (RAM), storage memory (ROM), and execution time recorded during the run-time. It is observed that memory and space consumed by both models do not have much of a difference. However, the execution time is approximately 3.5 seconds more in the case of the proposed DL model. As the nodes take water meter images with a frequency of one image per minute, the slight increase in the execution time does not hinder the algorithm's functioning in any way.

## VI. CONCLUSION

In this paper, DL based detection model (CNN trained with tansfer learning) was proposed. A rich dataset was collected, keeping variation in the location of deployed IoT nodes. The data set collected will also be made public later to help advance research in this field. We trained our model using the collected dataset. We analyzed the meter reading data using both the ML (RF-based) and proposed DL (CNN model trained with transfer learning) models and found their accuracies. Visual analysis was done to understand how ML models are trained using HOG features and how the proposed DL model's features are trained to understand what precisely a particular model understands from the features extracted while training itself. It was seen that the proposed DL-based model had more generalized features than the ML model while having computational complexity sufficient enough to fulfil the tasks, thus making it robust to use on different deployed nodes. In order to get a better understanding, an analysis was done on all the six nodes, and meter reading detection was performed using both the models. As a result, it was seen that the DL model had more than 99% accuracy while the ML model had accuracy between 91% to 97% for all the six nodes. Finally, an error analysis was given, including DER, VER and RMSE for each node. It was observed that the ML model had maximum VER of 8.61%, while the proposed DL model had maximum VER of 0.62%. The maximum RMSE for ML model was 0.351 KL, while for proposed DL model maximum RMSE was 0.06 KL. In future, it is intended to deploy more smart water meters on the campus to do a detailed network analysis based on water consumption patterns. The DL algorithm will also be expanded to detect 'half digits' for better precision of recorded meter readings.

TABLE V
COMPLEXITY ANALYSIS

| | Memory (MB) | Space (MB) | Execution time (s) |
|---|---|---|---|
| ML model [3] | 250 | 33.2 | $\sim 1$ |
| Proposed DL model | 280 | 44.8 | $\sim 4.5$ |


## ACKNOWLEDGEMENT

This research was supported partly by National Geospatial Programme (NGP), India, under grant no. 2073 (2020), PRIF Social Incubator Program (2019) and the Ministry of Electronics and Information Technology (MEITY), Govt. of India under grant no. 3070665 (2020), with no conflict of interests.


## REFERENCES


[1] C. Son, S. Park, J. Lee, and J. Paik, "Deep learning–based number detection and recognition for gas meter reading," *IEIE Transactions on Smart Processing & Computing*, vol. 8, pp. 367–372, 10 2019.

[2] H. J. C. Barreto, Kurtulan, S. İnci *et al.*, "Converting utility meters from analogue to smart based on deep learning models," *Innovations in Intelligent Systems and Applications Conference*, pp. 1–4, 2020.

[3] A. K. Lall, A. Khandelwal, R. Bose, N. Bawankar *et al.*, "Making analog water meter smart using ML and IoT-based low-cost retrofitting," pp. 157–162, 2021.

[4] JioFi JMR540: LTE based portable WiFi hotspot, accessed 27 Feb. 2021. [Online]. Available: https://www.jio.com/shop/en-in/router-jmr540-black-/p/491193576

[5] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li *et al.*, "Imagenet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[8] A. Paszke, S. Gross, F. Massa, A. Lerer *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *33rd Conference on Neural Information Processing Systems*, pp. 8024–8035, 2019.

[9] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba, "HOGgles: Visualizing object detection features," *IEEE International Conference on Computer Vision*, pp. 1–8, 2013.

[10] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," *IEEE conference on computer vision and pattern recognition*, pp. 5188–5196, 2015.