A review on a machine learning approach of an intelligent irrigation monitoring system with edge computing and the internet of things

L R Loua1*, M A Budihardjo2, S Sudarno2

1Department of Environmental Engineering, Diponegoro University, Jl. Prof. Sudarto No. 13, Tembalang, Kota Semarang, 50275, Jawa Tengah, Indonesia
reneloua1@students.undip.ac.id

Abstract: Water consumption during irrigation has been a much-researched area in agricultural activities, and due to the frugal nature of different practiced irrigation systems, quite a sufficient amount of water is wasted. As a result, intelligent systems have been designed to integrate water-saving techniques and climatic data collection to improve irrigation. An innovative decision-making system was developed that used Ontology to make 50% of the decision while sensor values make the remaining 50%. Collectively, the system bases its decision on a KNN machine learning algorithm for irrigation scheduling. It also uses two different database servers, an edge and an IoT server, along with a GSM module to reduce the burden of the data transmission while also reducing the latency rate. With this method, the sensors could trace and analyze the data within the network using the edge server before transferring it to the IoT server for future watering requirements. The water-saving technique ensured that the crops obtained the required amount of water to ensure crop growth and prevent the soil from reaching its wilting point. Furthermore, the reduced irrigation water also limits the potential runoff events. The results were displayed using an android application.

1. Introduction
Agriculture has always been the driving force of every nation in attaining food self-sufficiency and based on the type of farming practiced, small- or large-scale agriculture. There is a growing demand to meet the food supply of the ever-growing global population. Agriculture covers a wide variety of areas such as fisheries, poultry, horticulture etc. and in most developed and developing countries, it covers a significant portion of their Gross Domestic Product (GDP). Agricultural practices have evolved over the years with modern farming techniques to obtain greater crop yield. It has rendered traditional farming obsolete in many developed countries, but there is a continued dependence on these practices in developing and third-world countries, which requires a lot of water and energy to execute. As a result, improving and optimizing old farming techniques and technologies is crucial in increasing crop yield, which will help sustain humanity’s development.

Over the past three decades, intelligent systems have revolutionized the processes in every industry, and their application can be found in the agricultural industry in self-driving tractors, GPS field mapping, uses of sensors on the farm, field-level weather forecast and machine optimization [1]. It has been used in irrigation management, precision farming, microclimatic monitoring, water management and disease
evaluation [2]. It has seen a tremendous improvement in crop yield as more crop data is made available and better farming techniques are thus implemented for sustainable and increased crop growth. Its application in agriculture aims to connect different sensors, flow meters, actuators to the internet to measure and record different environmental parameters like soil moisture, temperature, humidity, soil pH etc. It is advantageous in many ways as it makes enhanced monitoring and evaluation possible and observes plants’ growth response concerning water needs. Traditional farming is one of those techniques that intelligent systems are trying to make obsolete as it uses far too much water, which is often wasted.

This paper [3] designed an intelligent system that efficiently uses water for tunnel farming and combines machine learning and IOT semantics to control climate, soil and crop type parameters with sensors for different outputs like temperature, humidity and soil moisture. The method used focused on Machine learning and edge computing for a reliable system with the Ontology aspect used for the plant species data, different soil and climatic types.

2. Methodology
The system's architectural design can make intelligent decisions for the irrigation of plants by considering different factors and climatic conditions, including crop/soil type, temperature, humidity, and soil moisture. The IoT architecture was divided into four parts: the application, processing, transport, and perception layers. The perception layer contains the system's hardware, such as the sensors, actuators and microcontroller, which assembles the data sent through the transport layer. This layer contains wireless/2G/3G or LAN networks that allow sensed data to be collected and processed. It also provides the schedule for watering the plants, supervising and providing necessary suggestions. The processing layer is then used to store, scrutinize and process the data coming through from the transport layer, which uses databases, cloud and edge computing. The last layer, the application layer, provides application-specific services to the end user [3]. Figure 1 shows the proposed architecture of the watering system.

2.1 Sensor data and semantic knowledge base
Sensing data from the sensors was one of the most critical layers in the system, and this had to be done through the use of physical components which form the brain of the design architecture. An Arduino board was used to sense the data from the humidity, temperature and soil moisture sensor through its analog pins, and the data was recorded every 30secs before it was transferred through a SIM808 GSM model. This GSM model allowed the data to be sent through the internet to a server developed solely for the system. After a decision-making process has been completed, the final results are easily visualized through an android application. Through this application, the user can control the valves of the actuators and can release water whenever it has needed.

Figure 1. Proposed model for the innovative irrigation system.

The machine learning decision-making algorithm was deployed at the IoT server to make schedules for irrigating the plants quickly. A semantic data model (SDM) was made to incorporate real-time data
supervision whereby different logical stages were functional for categorizing the perceptions and assessment of available data. These concepts, which make up the ontology decision-making for the estimation of the amount of water needed, were based on three parameters, *crop, climatic and soil type*, and together, they established the structured data for the query decision.

![Diagram of hardware design for the integrated system](image)

**Figure 2.** Hardware design for the integrated system.

Figure 2 shows the design hardware where the decision to be made comprises SPARQL (RDF query language) vital in making the watering system work. The data obtained by the sensors are taken from vast farmland, and different properties were sensed at field level, and this includes the temperature/humidity and luminance, whereas the soil moisture (deep) was measured at the quadrant level. The data also contained all the information needed to train the ML algorithm before sending it to a control agent, which also receives water in a specific soil texture.

As already discussed, some of the parameters tend to influence plants water needs. Another feature that also influences the level of water is the type of soil. Depending on the farming area and different climatic conditions, the soil type will be fundamental in determining the amount of water needed for different crops, as some crops need more water while others require a sufficient amount for growth.

The framework of the decision-support-system shown in figure 3 shows the flowchart, highlighting the ontology and data needed to extract the decision which will be used in irrigating the plants. After that, data sensed from pasture/cropland, soil, and climate type would be used to determine the required amount of water needed on the field. Watering commands is then displayed on the mobile application as recommendations for the end-user who can operate the whole system by sending commands to execute the required directives on the field, which also involves the actuation of the valves.
2.2. KNN algorithm and analysis technique

The KNN machine learning approach was used as a supervised tool to make the necessary water level requirement prediction. Other machine learning approaches that could be used include random forest, Naïve Bayes, decision trees and support vector machines. The KNN \((k=5)\) modelling practice used the whole dataset to predict the future data instance by searching the available data to find the "\(k\)" number of neighbors closest to the dataset. In an instance where "\(k\)" is given as 3, the algorithm uses the three most similar neighbors in assigning a class label to the instant data, which would then be given to the most common class label (i.e., among the \(k\)-training instance). Table 1 shows the five classes used to train Machine learning with the different sensor characteristics.

| Soil Moisture (%) | Temp. (°C) | HDT (%) | Class               |
|-------------------|------------|---------|---------------------|
| < 30              | >45        | <30     | Highly Needed (HN)  |
| 30-45             | 35-45      | 30-45   | Needed (N)          |
| 46-60             | 24-34      | 46-60   | Average (A)         |
| 61-80             | 20-24      | 61-80   | Not Needed (NN)     |
| 81-100            | < 20       | >80     | Highly Not Needed (HNN) |

In addition, the algorithm for the KNN was divided into several steps as highlighted below.

Step 1: Calculating the Euclidean distance of the new data \(X\), which features the prediction of the resultant classes A, B, C, D and each present point \(P_n\) in the input dataset \(S\) and the equation gives this:
Euclidean Distance =
\[ \sqrt{(XA - PA)^2 + (XB - PB)^2 + (XC - PC)^2 + (XD - PD)^2} \]  
(2.1)

Step 2: Choosing the value of \( k \) to the closest neighbour on the new data \( X \):
\[ k = 5. \]
(2.2)

Step 3: Count the ballots of all the \( k \) neighbours to predict the class of test data \( X \)

Step 4: Assigning the new data \( X \) to the class with more votes.

3. Results and discussion

3.1 Machine learning, edge computer and android application

Applying the intelligent watering system for tunnel farming using different crops such as rice, maize, sugarcane, cotton, and wheat shows the extent of IoT usage in employing the much-needed decision for water management in different farming techniques. With the use of different environmental sensors deployed at the main field, real-time data was made available, which helped analyze the system's feedback and allowed the farmer to see the performance of these sensors through his/her android device.

In training the model, which was implemented using Anaconda (created for python programs), different considerations were made with regards to different water requirements for different crop types, and as such, it prioritized the water requirements for the crops based on:

- Rice > sugarcane > Maize > Cotton > Wheat.

In considering, a general rule was created, which elaborated and identified the range for the humidity—temperature and soil moisture for all the crops to identify the class. Table 2 shows this general rule for the three sensors as well as the Ontology decision and class. Similarly, regarding water conditions for different climate and soil types, the following working rules were generated:

- Sand and gravel > Clay > Silt > loam > Organic soil
- Hot and Dry > Hot and humid > Cold and Humid

| Rule No. | Temperature | Humidity | Soil Moisture | Ontology Decision | Class |
|----------|-------------|----------|---------------|--------------------|-------|
| 1        | \( T > 50 \) | \( H < 20 \) | SM < 20        | HN                 | HH    |
| 2        | \( T > 40 \) | \( H < 40 \) | SM < 40        | HN                 | N     |
| 3        | \( T > 30 \) | \( H < 60 \) | SM < 60        | HN                 | A     |
| 4        | \( T > 20 \) | \( H < 80 \) | SM < 80        | HN                 | NN    |
| 5        | \( T < 20 \) | \( H > 80 \) | SM >80         | HN                 | HNN   |
| 6        | \( T > 57 \) | \( H < 20 \) | SM <10         | N                  | HN    |
| 7        | \( T > 40 \) | \( H < 30 \) | SM <30         | N                  | N     |
| 8        | \( T > 35 \) | \( H < 40 \) | SM <40         | N                  | A     |
| 9        | \( T > 30 \) | \( H < 60 \) | SM <60         | N                  | NN    |
| 10       | \( T < 30 \) | \( H > 60 \) | SM >60         | N                  | HNN   |
| 11       | \( T > 57 \) | \( H < 30 \) | SM <60         | A                  | HN    |
| 12       | \( T > 40 \) | \( H < 60 \) | SM <60         | A                  | N     |
| 13       | \( T > 35 \) | \( H < 40 \) | SM <40         | A                  | A     |
| 14       | \( T > 30 \) | \( H < 60 \) | SM <60         | A                  | NN    |
| 15       | \( T < 30 \) | \( H > 60 \) | SM >60         | A                  | HNN   |
| 16       | \( T > 50 \) | \( H < 30 \) | SM <40         | NN                 | HN    |
| 17       | \( T > 40 \) | \( H < 40 \) | SM <60         | NN                 | N     |
| 18       | \( T > 30 \) | \( H < 60 \) | SM <80         | NN                 | A     |
| 19       | \( T > 20 \) | \( H < 90 \) | SM <100        | NN                 | NN    |
| 20       | \( T < 20 \) | \( H > 90 \) | SM >100        | NN                 | HNN   |
| 21       | \( T > 50 \) | \( H < 30 \) | SM < 30        | HNN                | HN    |
| 22       | \( T > 40 \) | \( H < 50 \) | SM < 60        | HNN                | N     |
| Rule No. | Temperature | Humidity | Soil Moisture | Ontology Decision | Class |
|---------|-------------|----------|---------------|-------------------|-------|
| 23      | T > 30      | H < 60   | SM < 80       | HNN               | A     |
| 24      | T > 20      | H < 80   | SM < 90       | HNN               | NN    |
| 25      | T < 20      | H > 80   | SM > 90       | HNN               | HNN   |

With the use of the predictive data analysis tool, Scikit-learn, the model recognized the general rules before the data was made available to the end-user to extract through the help of a cloud server, Heroku cloud application platform. The web service was set up in this stage using the Flash API, which defined routes for the HTTP request handles that allowed data to travel from the perception layer to the edge server Firebase through the sensor-Arduino side.

Furthermore, the end-user application gave the individual (farmer) the option to choose from the three input parameters (crop, climate and soil type), which can be accessed using a dropdown button. Input parameters have been assigned a unique code based on the class label, which allows the algorithm to read the requirement for each label before it is transferred to the server where the ontology section is found. The decision sent to this section can then be extracted along with other data from the different sensors, made available to the central IOT server where the ML algorithm is located. The dataset meant for the training of the machine learning algorithm contains encoded values for each of the labels and hence the front-end user, the farmer, sees the converted code in text form. Figure 4 (a, b, c) shows the different interfaces available to the user whereas, table 3 highlights the conversion codes.

**Table 3. Codes for each class labels.**

| Class               | Code |
|---------------------|------|
| Highly Needed       | 3    |
| Needed              | 2    |
| Average             | 1    |
| Not Needed          | 0    |
| Highly Not Needed   | -1   |

**Figure 4.** Front-end user interface (a, b and c).
3.2 Performance evaluation

Tests were conducted on the sample data collected randomly from about 500 instances used to train the KNN model to forecast class labels. Precision and recall were used to estimate the model’s performance where \( k \) was taken to be equal. The results shown in figure 5 shows an accuracy rate of 0.45 and a macro average of 0.48 precision, 0.51 recall, and 0.48 the F1-score. The “\( k \)” value has to be chosen precisely to increase the accuracy and based on the general working rules of the algorithm, the \( k \)-value for any two classes should be an odd value whereas, for more than two classes, the value should not be a multiple of the resultant classes.

In this model, the predicted results contain 5-five labelled classes, and as such, the \( k \)-value had to be chosen accordingly. In choosing a suitable “\( k \)”-value, a graph of “\( k \) value versus mean error” had to be plotted to identify the error trend and hence “matplotlib-pyplot” was used to graph it, as shown in figure 6.

Furthermore, the mean error increased at first up to 0.5 as the "\( k \)" value increases but later decreased to 0.3 when the “\( k \)” value reaches 10-11. After that, the error rate started to rise and continued with an increase in the \( k \)-value. It means that to obtain the lowest mean error, a maximum value of 11 for the “\( k \)” has to be obtained, which would significantly improve the model's performance, as the accuracy report rightly highlighted.

In figure 5, the predicted, “Needed” class label lacks precision due to the performance dependency of the value "\( k \)” with the nearest neighbor. It causes a graphical representation of the matrix of confusion with and without normalization, as shown in figure 7.

Significant improvements in the model's performance were noticed in the accuracy rate when the maximum “\( k \)” value was set to 11. Whereas, there was also a massive increment in the precision value of the “Not Needed” class from 0.33 to 1.0, which also shows an accuracy rate from 33% to 100%.

![Figure 5. Accuracy report with “k = 5”](image)

![Figure 6. Error rate concerning the k value.](image)
Likewise, the “Average” class also had an increased precision value from 0.25 to 0.33, which explains the increased accuracy rate from 25% to 33%. Based on these results, there is a high possibility of turning the model by simply adjusting the value to obtain a better system performance. Figure 8 showed the results when the value of “k” was set to 11.

![Confusion Matrices with and without normalization for "k = 5".](image)

![“k = 11” accuracy report.](image)

4. Conclusion
The intelligent irrigation watering system was implemented using a sensor network deployed in three phases. This sensor data phase includes collecting data using the DHT22 temperature/humidity sensor, the HL-69 hygrometer to measure the soil moisture content of the soil and the BH1750 light sensor. The second phase provides the difference in the system’s architectural design, which integrates the KNN (k=5) machine learning algorithm on a sample dataset to train the model in making efficient irrigation decision-making requirements for different crop, soil and climatic types. The third phase connects the system to the edge and main IoT servers, allowing data to be transferred and received through HTTP requests. The system was implemented using Anaconda, a Python data science platform that allows the classification of the input values into five; Highly Needed, Needed, Average, Not Needed, and Highly Not Needed.

Furthermore, the results show that the performance of the KNN (k=5) had a reasonable accuracy rate of 0.45, which could be increased by increasing the value of k to 11. It shows the tuning possibility of the model for better performance of the system. In addition, the large sample dataset made it easier to train the model to recognize similar neighbors, which is crucial in predicting future water requirements for different class labels.

The android application is user-friendly as it combines all the necessary information for the farmer to utilise and make decisions for different crops on the field. It reduces the time and workforce needed
to water the field. Nonetheless, it will require the understanding of data science analysis to study the data and make subsequent changes if necessary, and this can be a challenge if the farmer does not have the knowledge and understanding to interpret such data.

Finally, future additions to the system could include analysing the evapotranspiration on the field, calculating surface water runoff for the different crop and soil types, and using different water-saving scenarios to determine the amount of water loss for a specific climatic condition and soil type.

References
[1] Pillai R and Sivathanu B 2020 Benchmarking 27(4) 1341–1368
[2] Nigussie E, Olwal T, Musumba G, Tegegne T, Lemma A and Mekuria F 2020 Procedia Comput Sci 177 86–93
[3] Munir M S, Bajwa I S, Ashraf A, Anwar W and Rashid R 2021 Complexity 2021
[4] H. Sattar, I. S. Bajwa, Amin R U et al 2019 IEEE Access 7 144500–144515
[5] Sarwar B, Bajwb B S, Jamil N, Ramzan S and Sarwar N 2019 Sensors 19(14) 3150
[6] Kumar A, Kamal K, Arshad O M, Mathavan S and Vadamala T 2014 in Proceedings of the Global Humanitarian Technology Conference (GHTC) (USA: Institute of Electrical and Electronics Engineers (IEEE))
[7] Parameswaran G and Sivaprasath K 2016 Int J Eng Sci 6 5518
[8] Stubbs M 2016 Irrigation in US Agriculture: On-Farm Technologies and Best Management Practices (Washington: Congressional Research Service)
[9] TongKe F 2013 J Converg Inf Technol 8(2)
[10] Cardenas-Lailhacar B, Dukes M D and Miller G L 2008 J Irrig Drain Eng 134 84–193
[11] Gutierrez J, Villa-Medina J F, Nieto-Garibay A and Porta-Gandara M A 2014 IEEE Trans Instrum Meas 63(1) 166–176
[12] Sales N, Remédios O and Arsenio A 2015 IEEE 2nd World Forum Internet Things (WF-IoT) 2015 p 693–698
[13] Rawal S 2017 Int J Comput Appl 159(8) 7–11
[14] Saab A, Therese M, Jomaa I, Skaf S, Fahed S and Todorovic M 2019 Water 11 252