2023 20th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON) | 979-8-3503-0052-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/SECON58729.2023.10287454

LSTM-Driven Scheduling for Energy-Efficient Crop Monitoring in Wireless Networks

Ziyue Dang
Dept. of Computer Science and Technology
Tsinghua University
Beijing, China
dzy19@mails.tsinghua.edu.cn

Fan Dang
Global Innovation Exchange
Tsinghua University
Beijing, China
dangfan@tsinghua.edu.cn

Yankun Yuan
School of Information Science and Engineering
Yanshan University
Qinhuangdao, China
yyk13964371891@gmail.com

***Abstract*—Low-power wireless networks are widely used to monitor crop growth in smart agriculture. However, there is a growing need for more fine-grained monitoring to improve the yield of certain fruits and vegetables. The system must maintain low power consumption of peripheral devices while still providing a satisfactory quality of experience (QoE) for more frequent queries. Conventional fixed-time communication between central and peripheral devices fails to offer a well-rounded solution to this trade-off problem. To achieve a better balance, we propose an LSTM-driven transmission scheduling method. By learning the user's past query patterns, the LSTM predicts the time of future queries initiated by the users, allowing the system to plan data transmission between the central and peripheral nodes ahead of time. Our method also predicts the future pattern of collected data to ensure that significant changes are actively recorded, even if not queried. Compared to other machine learning methods, our LSTM prediction results have a smaller error. The simulation results demonstrate that our approach can greatly improve QoE while achieving lower power consumption.**

***Index Terms*—Low-power WSNs, LSTM, Smart Agriculture, IoT**

I. Introduction

The use of low-power wireless sensor networks (LPWSNs) in smart agriculture has grown in popularity for improving farm production and efficiency. LPWSNs consist of small, low-cost sensors that measure environmental variables like temperature, humidity, soil moisture, and light intensity. They find applications in precision irrigation, crop monitoring, and pest management. LPWSNs enable real-time data on crop growth, allowing farmers to optimize conditions by adjusting soil moisture and fertilizer application rates. LPWSN usage is expected to further increase with evolving technology [1].

Implementing detailed monitoring in agriculture is essential for enhancing the yield and quality of popular fruits and vegetables. For perishable crops like strawberries and tomatoes, picking them at the right moment is crucial for retaining their quality. Current reliance on subjective experience for gauging plant growth introduces errors in processes like pollination, spraying, harvesting, and marketing. However, advanced monitoring of environmental factors such as temperature and humidity can provide valuable insights for nurturing and harvesting these crops, optimizing their growing conditions, and increasing production [2, 3]. Precise and real-time monitoring of crops and their growing conditions is thus essential for ensuring better yields and quality. Fine-grained monitoring allows farmers to achieve improved production outcomes and optimize their farming processes.

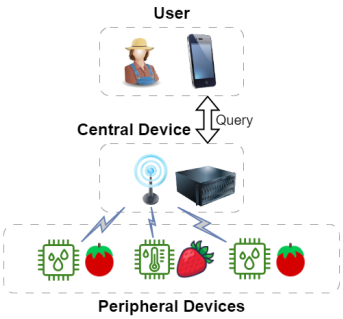


Fig. 1. Example of an LPWSN system that monitors environmental factors for crop growth.

Figure 1 illustrates a typical crop monitoring scenario. Sensors for monitoring temperature and humidity are placed in fields or greenhouses and wirelessly connected to a central device. These sensors collect and transmit data regularly. Farmers can remotely access the data stored on the central device through a mobile app to monitor crop growth. Key characteristics of this scenario include:

- **Low power consumption**: Peripheral devices are battery-powered to eliminate the need for cables, requiring low power consumption to minimize battery replacements.
- **Good Quality of Experience (QoE)**: Users expect up-to-date data when accessing the central device, necessitating a superior QoE to minimize the time difference between data collection and user query.
- **Differentiated user query patterns**: As sensors are installed for various crops and needs, user query patterns towards specific sensor data may vary and lack a distinct pattern.

One of the main challenges of optimizing a device's performance is balancing power consumption and QoE. For optimal QoE, peripheral devices must transmit data to the central device in real-time, which significantly increases power consumption. However, reducing the frequency of communication between peripheral and central devices to lower

power consumption can negatively impact QoE. In a simplistic setup where peripheral devices send data at fixed intervals, achieving a balance between low power and high QoE can be difficult. Differentiated user query patterns further complicate this issue, as fixed time communication may not meet the QoE requirements for all query patterns.

This paper proposes a method to optimize transmission scheduling by accurately predicting user query patterns for different peripheral devices based on past queries. The central device can then use these predictions to schedule data transmissions effectively.Machine learning algorithms, specifically LSTM, are integrated into the central device for improved prediction accuracy in complex time series.

Our paper contributes by optimizing transmission scheduling of peripheral nodes using LSTM, achieving a balance between low power consumption and good QoE. By modeling past user queries to predict future queries, the central device can schedule data transmission with peripheral devices, making it suitable for fine-grained crop monitoring scenarios.

The rest of this paper is organized as follows. Section II introduces the related work. Section III presents the system design, including the overall design, implementation and model selection. The simulation results of our proposed methods is shown in Section IV. Section V concludes the paper.

II. Related Work

A. Network Scheduling Optimization Methods based on Machine Learning

Machine learning has provided feasible solutions to the problem of network scheduling optimization and has been extensively studied in recent years.

ARIMA model is a widely applied statistical model that is suitable for predicting stationary time series data. Its good interpretability makes it one of the most commonly used machine learning methods in tasks such as modeling and predicting wireless network traffic [4, 5]. However, ARIMA's limitation is that many real-life time series data is generated by more complex processes, which does not meet the assumptions of the ARIMA model, and leads to unsatisfactory performance.

Random forest is an ensemble learning method that combines multiple decision trees for prediction. The main advantages of random forest are its robustness, interpretability, and good performance in handling large datasets [6]. In Wang et al. [7], random forest regression algorithm was used to accurately predict the arrival time of wireless network traffic, which had a much better performance than linear models.

LSTM neural networks, known for capturing long-term dependencies [8], are suitable for processing sequential data and applied in various time series prediction tasks. Studies [9, 10] demonstrate that LSTM and other recurrent neural networks outperform statistical learning methods like ARIMA in network traffic prediction. Rostami et al. [11] utilize an LSTM-based framework to predict arrival time of data packets and dynamically adjust wake-up signal scheduling. However, overfitting may occur with LSTM and other deep neural networks when training data is limited [12], requiring longer training time and more computing resources compared to statistical learning methods.

Hybrid models are commonly used to combine the strengths of different machine learning methods and effectively reduce prediction errors. Madan and Mangipudi [13] proposed a network traffic prediction method using Discrete Wavelet Transform (DWT), ARIMA, and RNN. First, DWT decomposes the data into nonlinear and linear components, which are then separately predicted using ARIMA and RNN. Yang et al. [14] developed a hybrid model based on ARIMA and Back Propagation Neural Network, optimized with Simulated Annealing (SA). However, if a specific predictor performs poorly, it can become the bottleneck of the hybrid model, significantly limiting overall performance.

B. Wireless Sensor Network Applications in Smart Agriculture

As an effective solution for smart agriculture, various wireless network applications have been proposed for remote monitoring of various environmental parameters and providing data to improve crop yields, reduce input costs, and enhance agriculture productivity.

Many works focus on developing and evaluating WSNs for specific agriculture applications. Diedrichs et al. [15] presents a WSN for frost detection in precision agriculture, while Heble et al. [16] presents a low-cost, low-power IoT network for soil moisture monitoring. Ghanshala et al. [17] proposes a wireless monitoring system in precision agriculture, which focuses on the hardware and software aspects of the system, including energy-saving algorithms and networking protocols. These papers all demonstrate the effectiveness of the WSNs in their respective applications and evaluate them against state-of-the-art designs.

There are also works that emphasize the construction of a more generalized platform that supports IoT devices for smart agriculture. Vasisht et al. [18] presents FarmBeats, an IoT platform designed specifically for agriculture. The platform enables data collection from various sensors, cameras and drones, and is designed to account for power and internet outages. Olivares-Rojas et al. [19] proposes an architecture based on distributed edge-fog-cloud computing to achieve more efficient measurement, processing, and forecasting for agricultural applications, which can provide more intelligent measurement and data processing to help decision-making.

III. Methodology

A. Overall Design

In order to optimize the transmission scheduling of low-power network nodes and solve the trade-off problem between low-power consumption and good QoE, a feasible approach is to predict when the user end will initiate the query. Based on this idea, this study proposes an overall design as shown in Fig. 2. Since the central device both interacts with the user end and communicates with peripheral devices, it has the most comprehensive information and controls the entire network. Therefore, this paper proposes to incorporate a predictor into the central device, which learns the query patterns of past users

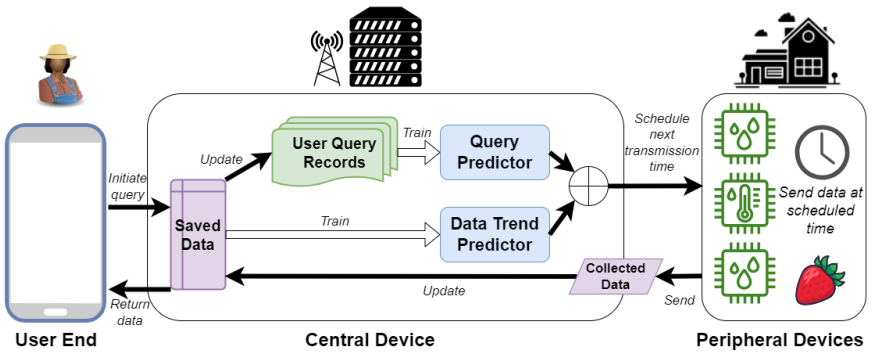


Fig. 2. The overall design.

through machine learning methods to make more accurate predictions.

Specifically, when the user initiates a query to the central device, the central device saves the query history for each peripheral device while returning data. The central device trains a query predictor for each peripheral device's query record. Besides, a data trend predictor is trained for each peripheral device based on the collected data, which predicts the future pattern of the data. This predictor serves as a supplement for the scheduling process, if significant changes is predicted, then the central device will also collect data from the peripheral devices, so that these changes can be actively logged even if not being queried. As data updates, the predictors will also train on the new data regularly to adjust the parameters. The results of the two predictors are combined to provide the scheduled transmission time within a future time window and the next data transmission time is sent to the peripheral device. The peripheral device has a timer and sends data to the central device at the scheduled time. In this way, the peripheral device does not need to consume additional energy to send or receive data when there is no communication demand.

To conclude, the query predictor and data trend predictor work together to determine the optimal transmission time for each peripheral device within a future time window. This means that the central device can predict when the user end will initiate a query or when the monitored data will experience drastic changes, and proactively collect the necessary data from the peripheral devices before the actual query is made. However, incorrect predictions can have negative effects, such as delays in data transmission and reduced QoE. To mitigate this, continuous training and adjustment of predictors based on new data are necessary. Regular updates ensure adaptability to evolving query patterns and data trends, maintaining the effectiveness of the proactive data provision approach. Section IV also validates the proposed design's ability to perform well despite possible errors.

B. Dataset and Model Training

This study uses real smartphone application usage data to simulate user end query records to train and test the query predictor. The app usage dataset used in this study is the open-source dataset collected by Yu et al. [20]. This data set contains usage records of more than 10,000 different applications from over 6 million different devices which were collected in one week. To ensure that the amount of data used for training is sufficient and the usage pattern meets the agricultural application scenario, our study selects usage records with a usage count greater than 1,200 and an average usage interval greater than 450 seconds in the dataset, and splits longer records into lengths of 1,200 to simulate a limited memory. Finally, 34 records with a length of 1,200 are selected for this study. Two open-source garden soil moisture datasets[21, 22] are used to train and test the data trend predictor. Both datasets contain data collected by different sensors and the data was logged every 15 minutes. Similarly, the data are segmented into a fixed length of 1,200. The dataset was split into 37 pieces of data. For each piece of data, our study uses the first 1000 entries for training and validation of the predictor, and the remaining 200 entries for testing the actual performance.

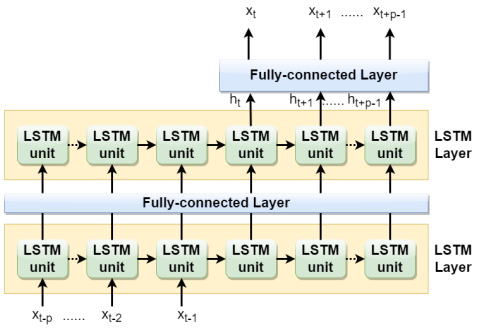


Fig. 3. LSTM predictor architecture. ($x_{t-p} \ldots x_{t-1}$ represents the p observations fed into the predictor, $x_t \ldots x_{t+p-1}$ represents the p predictions following the observations, and h_i represents the output of the LSTM layers. Dashed lines represent the LSTM units not shown in the diagram.)

As shown in Fig. 3, a stacked LSTM neural network architecture is proposed as a predictor. Multiple LSTM units are connected to form an LSTM layer of the network. A fully connected layer is added between each LSTM layer to improve

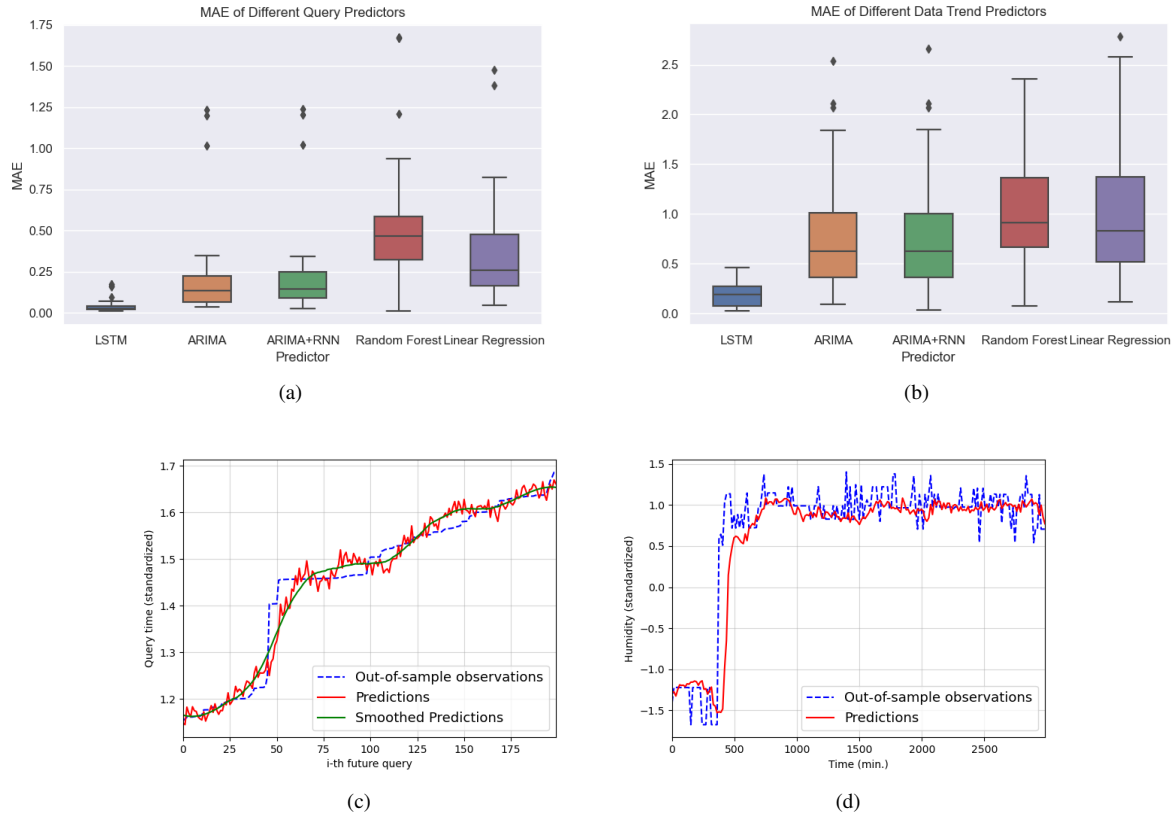


Fig. 4. Model selection. (a) MAE of different query predictors. (b) LSTM prediction sample for future query. (c) MAE of different predictors of the data trend. (d) LSTM prediction sample for future data trend.

TABLE I
LSTM TRAINING HYPERPARAMETERS

Parameter	Value
LSTM hidden states	128
LSTM layers	2
Dropout probability	0.2
Optimization algorithm	Adam
Initial learning rate	0.001
Number of epochs	150

TABLE II
RNN TRAINING HYPERPARAMETERS

Parameter	Value
RNN hidden states	128
RNN layers	2
Dropout probability	0.2
Optimization algorithm	Adam
Initial learning rate	0.001
Number of epochs	75

prediction accuracy, and dropout is used for regularization. The output of the LSTM layers is fed into a fully-connected layer to obtain the final output of the predictor. The ReLU function is used as the activation function, and Mean Absolute Error (MAE) is used as the loss function. The length of the input and output sequences p corresponds to the prediction window length. The hyperparameters selected for the LSTM predictor training are shown in Table I. Standardization is applied to training data to improve the performance of machine learning methods.

To further evaluate the performance of the LSTM predictor, we conducted a comparative study using other machine learning methods introduced in Section II, including ARIMA, Random Forest, and the ARIMA-RNN hybrid model proposed by Madan and Mangipudi [13]. Furthermore, linear regression was used as a naive baseline. When training the ARIMA model, the stepwise algorithm proposed by Hyndman and Khandakar [23] was used to automatically search for the optimal parameters. The hyperparameters used for training RNN and Random Forest in this study are shown in Table II and Table III, respectively.

C. Model Selection

For each prediction method implemented in section III.B, we use the first 1000 entries of the data for training, and let the trained predictors to predict the next 200 entries. We compare the predicted results and actual observations to calculate their MAE. The MAE of different models are shown in Fig. 4. It can be seen that for both future query prediction and future data trend prediction, the MAE of LSTM is significantly smaller than that of all other prediction methods. The MAE of the

TABLE III
RANDOM FOREST TRAINING HYPERPARAMETERS

Parameter	Value
Number of predictors	1000
Maximum depth	20

ARIMA model and the ARIMA+RNN hybrid model is also smaller than the naive baseline (i.e., linear regression), but the MAE of the hybrid model does not show improvement than that of the ARIMA model due to the limitations of the ARIMA model itself. Random Forest performs poorly, and its MAE is not smaller than that of linear regression. Fig. 4(c) and Fig. 4(d) shows a single case of the prediction results of LSTM, which are very close to the real data, confirming the previous evaluation results.

Based on the above results, we propose to use LSTM architecture for both predictors. From Fig. 4(c), it can be seen that the initial output of the LSTM query predictor contain a large amount of jitter, while the sequence of the next query times must be a monotonically increasing sequence. Therefore, it is necessary to further process the prediction results. In this study, we chose to use a Savitzky-Golay filter to smooth the preliminary output of the LSTM query predictor, and the filtered effect is also shown in Fig. 4(c).

IV. EVALUATION

A. Overview

In this section, software simulations were conducted under the assumption of an ideal state where there are no retransmissions in the data link layer to evaluate LSTM-driven transmission scheduling of energy-efficient network nodes through numerical results.

In Section IV-C and Section IV-B, we present the simulated performance of the query predictor and the data trend predictor, respectively. In this simulation experiment, the ARIMA model predictor and the ARIMA+RNN hybrid model predictor implemented in Section III were used as control groups against the LSTM predictor. These two prediction methods have been proven to have significant accuracy improvements compared to the naive baseline in the evaluation results shown in Fig. 4(a) and Fig. 4(b). The unused portion of the datasets from Section III-B is used as the test data for the simulation experiment. In the experiment, the simulation duration of the three predictors was controlled to be the same.

After evaluating the performance of the two prediction tasks, respectively, we combine the output of two predictors together to simulate the final scheduling process shown in Fig. 2. The results are presented in Section IV-D. We show that compared to fixed-time data transmission that is used in most crop monitoring LPWSNs, LSTM-driven scheduling is an energy-efficient solution with better QoE.

B. Simulated Performance for Future Query Prediction

The query predictor is the crucial part of the system, which provides the basis of future transmission scheduling. The

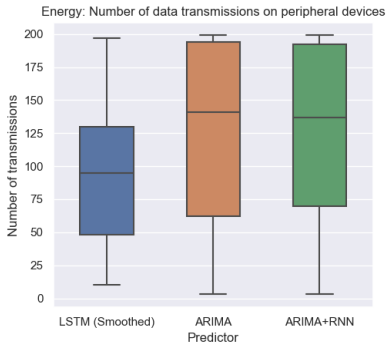


Fig. 5. Number of data transmissions on peripheral devices for different predictors.

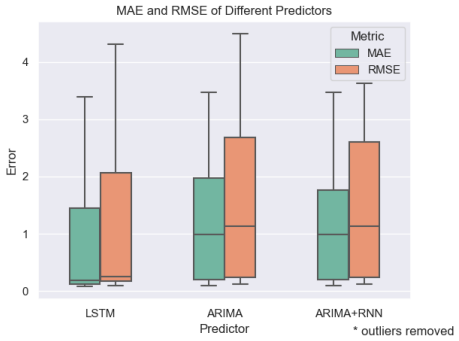
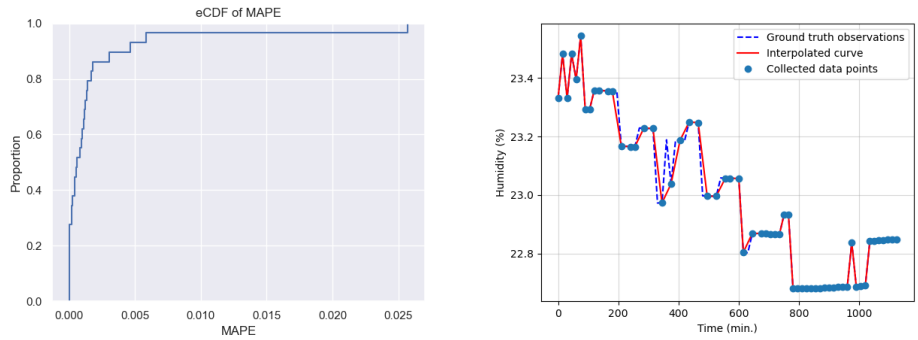


Fig. 6. MAE and RMSE of different data trend predictors.

performance of the query predictor is mainly evaluated from two aspects: QoE and energy consumption.

To evaluate the QoE of the system, this study calculates the difference between the moment when the user initiates a query (i.e., present) and when the central device last collected data from the peripheral device (i.e., past). This time difference indicates how long the data collected by the central device lag behind the queried moment. The smaller the time difference, the closer the system is to the ideal real-time system, and the higher the QoE. This time difference is calculated by finding the nearest scheduled data transmission time before the query time in the central device's outputs and computing the difference between the query time. The comparison between LSTM and other predictors in terms of QoE is shown in Table IV. As the data do not follow a normal distribution, the Mann-Whitney U test is used to compare the medians. Results show that LSTM-driven predictions can achieve significantly smaller time difference, indicating a better QoE compared with other models.

Based on simulation, this study only provides a rough characterization of the energy consumption of different prediction methods: since the main source of energy consumption of peripheral devices comes from communication with the central device, the energy consumption of different prediction methods can be indirectly evaluated by comparing the number of data transmissions on peripheral devices during



(a) (b)

Fig. 7. Accuracy of scheduled data collection compared with ground truth. (a) Empirical CDF of MAPE. (b) A sample of scheduled data collection.

TABLE IV
QoE: Time difference between query time and latest data update

Predictors	Median (s)	Mann-Whitney U test
LSTM	170.0	$U = 8076540.0$
ARIMA	222.0	$p < 0.001$ (***)
LSTM	170.0	$U = 8245715.0$
ARIMA+RNN	222.0	$p < 0.001$ (***)

TABLE V
Performance comparison between scheduled and fixed-time transmissions

Metric		Median	Mann-Whitney U test
QoE: Time difference	Scheduled	307.0 s	$U = 9672705.5$
	Fixed-time	436.0 s	$p < 0.001$ (***)
Energy: Num. of transmissions	Scheduled	75	$U = 418.0$
	Fixed-time	76	$p = 0.975$

the simulation process. The boxplot in Fig. 5 shows the number of transmissions for each simulation, where LSTM has fewer transmissions on average and median compared to ARIMA and hybrid models. This indicates that the LSTM-driven communication scheduling, which accurately predicts future query time, reduces unnecessary communications while improving QoE and maintaining lower energy consumption.

C. Simulated Performance for Future Data Prediction

As introduced in Section III-A, by predicting the pattern of future data collected by peripheral devices, the data trend predictor can aid the scheduling process by initiating data transmission when significant changes in humidity, temperature, etc. are expected. In this way, users will be notified of these changes in their mobile apps instantly.

In monitoring soil humidity, data transmission is initiated by the data trend predictor when a predicted humidity change greater than 1% is expected in the future. The collected data points are connected using linear interpolation to create a continuous curve. To minimize unnecessary transmissions, the data trend predictor aims to minimize the error between the interpolated curve and the ground truth. We use MAE and RMSE as error metrics, and compare the performance of LSTM, ARIMA, and hybrid models. The results, shown in Fig. 6, highlight the superior performance of LSTM in terms of smaller MAE and RMSE, indicating its better ability to predict data trends.

D. Simulated Performance of LSTM-Driven Scheduling

In this part, we compare our scheduling solution against the current fixed-time data transmission protocol used in the soil moisture datasets[21, 22], which transmits data from peripheral devices to the central device every 15 minutes. We pair the app usage data with the soil humidity data to form 29 sets of data that simulate a complete application scenario.

We evaluate the QoE and energy consumption of two systems using metrics defined in Section IV-B. The results in Table V show that LSTM-driven scheduling improves the time difference between query time and latest data update from 436.0 s to 307.0 s. This improvement is statistically significant under Mann-Whitney U test. Moreover, the energy consumption does not increase, indicating that our solution maintains the low-power characteristics of peripheral devices.

Finally, to validate that the sensor data collected under our scheduling scheme can fully reflect the ground truth, we compute the mean absolute percentage error (MAPE) between the linear interpolated curve and the ground truth. The MAPE's empirical cumulative distribution function (eCDF) is depicted in Fig. 7(a), revealing that over 90% of cases have a MAPE below 0.5% and the maximum MAPE is less than 3%. Fig. 7 shows a sample of data points collected using the proposed scheduling method, illustrating a strong resemblance to the ground truth. This further confirms the feasibility of LSTM-driven scheduling.

V. Conclusion

This paper highlights the trade-off problem regarding the quality of experience (QoE) and energy consumption in agricultural LPWSNs for crop monitoring, which requires a finer granularity. To address this issue, we propose an LSTM-driven scheduling method. Compared to other machine learning prediction methods, LSTM has a smaller margin of

error. Simulation experiments demonstrate that the LSTM-driven scheduling method can significantly enhance QoE while reducing energy consumption compared to typical prediction methods such as ARIMA. It demonstrates a higher level of accuracy in forecasting future data collection patterns. Compared to current fixed-time data transmission solutions, our method can substantially improve QoE, precisely reflect trends in sensor data, and also maintain approximate power consumption.

ACKNOWLEDGMENT

This work is supported in part by the National Key Research Plan under grant No. 2021YFB2900100.

REFERENCES

[1] G. Deepika and P. Rajapirian, "Wireless sensor network in precision agriculture: A survey," in *Proc. of the International Conference on Emerging Trends in Engineering, Technology and Science*, 2016, pp. 1–4.

[2] Q. An, K. Wang, Z. Li, C. Song, X. Tang, and J. Song, "Real-time monitoring method of strawberry fruit growth state based on yolo improved model," *IEEE Access*, vol. 10, pp. 124 363–124 372, 2022.

[3] S. Widiyanto, D. P. Nugroho, A. Daryanto, M. Yunus, and D. T. Wardani, "Monitoring the growth of tomatoes in real time with deep learning-based image segmentation," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 12, 2021.

[4] A. Azari, P. Papapetrou, S. Denic, and G. Peters, "Cellular traffic prediction and classification: A comparative evaluation of lstm and arima," in *Proc. of Discovery Science: the 22nd International Conference*, 2019, pp. 129–144.

[5] Y. Shu, M. Yu, O. Yang, J. Liu, and H. Feng, "Wireless traffic modeling and prediction using seasonal arima models," *IEICE transactions on communications*, vol. 88, no. 10, pp. 3992–3999, 2005.

[6] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[7] N. Wang, B. Li, M. Yang, Z. Yan, and D. Wang, "Traffic arrival prediction for wifi network: A machine learning approach," in *Proc. of IoT as a Service: the 5th EAI International Conference*, 2020, pp. 480–488.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using lstm networks," in *Proc. of the IEEE 29th annual international symposium on personal, indoor and mobile radio communications*, 2018, pp. 1827–1832.

[10] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *Proc. of the 17th IEEE International Conference on Machine Learning and Applications*, 2018, pp. 187–193.

[11] S. Rostami, H. D. Trinh, S. Lagen, M. Costa, M. Valkama, and P. Dini, "Proactive wake-up scheduler based on recurrent neural networks," in *Proc. of the IEEE International Conference on Communications*, 2020, pp. 1–6.

[12] M. M. Bejani and M. Ghatee, "A systematic review on overfitting control in shallow and deep neural networks," *Artificial Intelligence Review*, pp. 1–48, 2021.

[13] R. Madan and P. S. Mangipudi, "Predicting computer network traffic: a time series forecasting approach using dwt, arima and rnn," in *Proc. of the 11th International Conference on Contemporary Computing*, 2018, pp. 1–5.

[14] H. Yang, X. Li, W. Qiang, Y. Zhao, W. Zhang, and C. Tang, "A network traffic forecasting method based on sa optimized arima–bp neural network," *Computer Networks*, vol. 193, p. 108102, 2021.

[15] A. L. Diedrichs, G. Tabacchi, G. Grünwaldt, M. Pecchia, G. Mercado, and F. G. Antivilo, "Low-power wireless sensor network for frost monitoring in agriculture research," in *Proc. of the IEEE Biennial Congress of Argentina*, 2014, pp. 525–530.

[16] S. Heble, A. Kumar, K. V. D. Prasad, S. Samirana, P. Rajalakshmi, and U. B. Desai, "A low power iot network for smart agriculture," in *Proc. of the 4th IEEE World Forum on Internet of Things*, 2018, pp. 609–614.

[17] K. K. Ghanshala, R. Chauhan, and R. Joshi, "A novel framework for smart crop monitoring using internet of things (iot)," in *Proc. of the 1st International Conference on Secure Cyber Computing and Communication*, 2018, pp. 62–67.

[18] D. Vasisht, Z. Kapetanovic, J. Won, X. Jin, R. Chandra, S. N. Sinha, A. Kapoor, M. Sudarshan, and S. Stratman, "Farmbeats: An iot platform for data-driven agriculture." in *Proc. of the USENIX NSDI*, vol. 17, 2017, pp. 515–529.

[19] J. C. Olivares-Rojas, J. A. Gutiérrez-Gnecchi, W. Yang, E. Reyes-Archundia, and A. C. Téllez-Anguiano, "Smart metering architecture for agriculture applications," in *Proc. of the 36th International Conference on Advanced Information Networking and Applications*, 2022, pp. 411–419.

[20] D. Yu, Y. Li, F. Xu, P. Zhang, and V. Kostakos, "Smartphone app usage prediction using points of interest," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 1–21, 2018.

[21] https://www.kaggle.com/datasets/keithbellingham/raingarden-irrigation, accessed: 2023-05-30.

[22] https://www.kaggle.com/datasets/keithbellingham/cannabis-irrigation-using-soil-moisture-sensors, accessed: 2023-05-30.

[23] R. J. Hyndman and Y. Khandakar, "Automatic time series forecasting: the forecast package for r," *Journal of statistical software*, vol. 27, pp. 1–22, 2008.