Science & Technology Development Journal – Engineering and Technology 2024, ():1-8

Open Access Full Text Article

Research Article

Score-based decision tree: A simple approach for smart irrigation using real data

Dinh Phuc Hung[1], Nguyen Tran Tho[2], Trung Dang Anh[2], Nam Thoai[1,2,*]

Use your smartphone to scan this QR code and download this article

[1]*High Performance Computing Lab, Faculty of Computer Science and Engineering (HPC Lab), Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City, Vietnam*

[2]*TIST Lab, Advanced Institute of Interdisciplinary Science and Technology, Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City, Vietnam*

Correspondence

Nam Thoai, High Performance Computing Lab, Faculty of Computer Science and Engineering (HPC Lab), Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City, Vietnam

TIST Lab, Advanced Institute of Interdisciplinary Science and Technology, Ho Chi Minh City University of Technology (HCMUT), Vietnam National University Ho Chi Minh City, Vietnam

Email: namthoai@hcmut.edu.vn

History

- Received: 28-9-2023
- Accepted: 29-3-2024
- Published Online:

DOI :

ABSTRACT

In addition to effectiveness, practicality and efficiency have been considered crucial when considering an automated irrigation system. Awareness of such requirements has only increased since freshwater resources are becoming scarce, particularly in many agricultural regions of Vietnam. A considerable amount of effort has been put into creating approaches to solving these problems, which can be classified into two main approaches: Supervised learning, and reinforcement learning. Ordinary supervised learning approaches tend to rely on input from farmers and experts' knowledge. However, such approaches may lead to inaccuracy due to human over-estimation or underestimation of the amount of water needed, thus leading to resource waste and ramping up production costs. In contrast, reinforcement learning methods have proven to be efficient given their ability to hastily adapt to new changes or trends in the environment. But despite the benefits, its need for a reliable simulation system and commitment of time for running through trial-error steps has rendered it impractical for real-world uses. Moreover, deployment of such methods encounters resource-wise and architecture-wise setbacks. This paper proposed a simple mixture of said approaches that attempt to adapt the environment to a desired state. This paper also presented an overview of the environment settings and the system architecture in which the proposed method will be deployed in a way that the method can interact with the states of the environment. Our approach is also deployable on machines with limited computing power, does not require pre-configurations in a simulated environment, and the need for human intervention is minimal. The performance evaluation of the proposed method is also presented and shows remarkable improvement of the method over a set of data gathered from the environment.

Key words: Smart Irrigation, Machine Learning, Supervised Learning, Reinforcement Learning

INTRODUCTION

As our knowledge of artificial intelligence (AI) and its application in many aspects of our lives develop, much consideration has been made toward the use of it in agriculture. The necessity and immediacy of advancing farming practices have reached an unprecedented level, mainly in developing countries like Vietnam due to its nature of being one of the most vulnerable sectors to climate change impacts such as drought, flood, pests, and diseases[1]. Additionally, smart irrigation is a cross-disciplinary subject that strives to water plants using the least amount of water where possible, while still maintaining plant growth and crop production during harvest seasons, by integrating information technology into farming practices.

However, several prominent agricultural regions in Vietnam are grappling with freshwater scarcity, a situation resulting from various natural issues such as drought, soil salinization, and climate change, Ha and Simon[2] analyzed the urgency of water conservation in Vietnam's agriculture. Consequently, the development of an intelligent irrigation system, capable of autonomously scheduling irrigation plans and reducing water usage while still ensuring crop yield, has become indispensable.

Several studies have proposed methodologies to either partially[3] or entirely[4,5] base the irrigation system on a particular metric of the environment, in such cases, soil humidity. One old-fashioned way of controlling this metric involves implementing certain policies, which involve triggering the water pump when the humidity is off the desired threshold or adhering to a daily fixed timeframe. However, this approach's inability to automatically change its threshold values rendered it vulnerable to concept drifts of the environment, such as varying demands of plants on soil moisture at different crop stages or with different plant species or seasonal changes of the environment's state. On the other hand, AI-based research typically presumes the correctness of farmers' irrigation practices and attempts to replicate these experiences based on historical data[4]. Nevertheless, farm-

Cite this article : Hung D P, Tho N T, Anh T D, Thoai N. **Score-based decision tree: A simple approach for smart irrigation using real data**. *Sci. Tech. Dev. J. – Engineering and Technology* 2024; ():1-8.

ers' decisions may be incorrect, and the more they are directly involved in crop irrigation, the more devastating the impact on freshwater conservation is[3]. In some circumstances, under-irrigation decisions made by farmers will also lead to a decrease in the final crop yield[6]. To further address the difficulties, applications of AI have been met with physical and economic constraints due to the field being a late adopter of AI, combined with the lack of interest from governments. Previous work[7] attempts to mitigate such drawbacks but this approach is computationally intensive as it requires machines to interact with the environment continuously. Thus, practical implementations of research are still limited. Our paper presents a simplified yet applicable method for estimating irrigation time for plants that uses previous data of the environment but can still adapt to the real-time environment and seasonal changes of the environment. Moreover, our method is lightweight and can be easily deployed onto machines with very weak computing power.

The rest of this article is organized as follows. Section 2 summarizes the related work. Section 3 describes our system architecture and deployment. Then, Section 4 describes the conventional decision tree method and presents our proposed method. In Section 5, we describe our experiment settings and results for performance evaluation. Finally, concluding remarks are drawn in Section 6, along with our future work on the topic.

RELATED WORK

In order to assess the condition of soil, several studies which are based on soil moisture. Ho et al.[3] proposed a traditional approach to the problem by forecasting the moisture rate with a simple model and setting up a wireless system for the farmers to monitor and water the plants with little effort. Although this approach is practical and water-efficient, it is still dependent on the farmers to water the garden, which does not promise an optimal crop yield. Chen et al.[4] and La et al.[7] proposed an autonomous irrigation scheduling method based on an ensemble of several models such as support vector machines, decision trees, and neural networks. Both works tried to predict if a specific state of the environment needs watering, based on a set of rules deducted from farmers' experience. Their works were on point and are more suitable for systems that support continuous irrigation. However, for systems that can only afford to irrigate up to twice a day, this approach shows its drawbacks as it requires the pump motors and the server to stay active continually, which is very inefficient.

For this specific type of problem, we mainly concentrate on creating a lightweight algorithm, computing inexpensive, and interpretable. Thus, our main point of interest in designing an algorithm that satisfies our needs is that it must share similar characteristics to a decision tree. Domingos et al.[8] provide an algorithm called Very Fast Decision Tree (VFDT), which exploits the idea that a small sample can often be enough to choose an optimal splitting attribute using Hoeffding bound, but this method is used for a specific purpose and does not learn via a policy. However, Féraud et al.[9] came up with online decision trees that correspond to a policy and make decisions based on that policy, much like a reinforcement learning approach. Inspired by their works, we propose a method to overcome the practical difficulties of computationally heavy methods. Besides that, our algorithm is simple and easily deployable on servers with limited computing power.

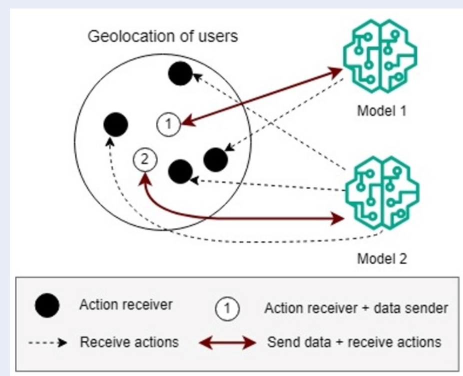

Figure 2: Deployment of models to users.

ABSTRACT DESIGN AND SYSTEM DEPLOYMENT

Abstract Design

In this section, we will provide a concise overview of the structure of our system. An illustration of the conceptual framework of our system is shown in Figure 1. The irrigation setup will establish a direct connection with the sensor array belonging to an environmental monitoring system. Our software stack is mostly comprised of the following software:

- Hadoop[10] is a framework for distributed processing and data storage across a cluster.
- Spark[11] is a framework for processing data at scale.
- Kafka[12] is a platform for handling real-time data events.

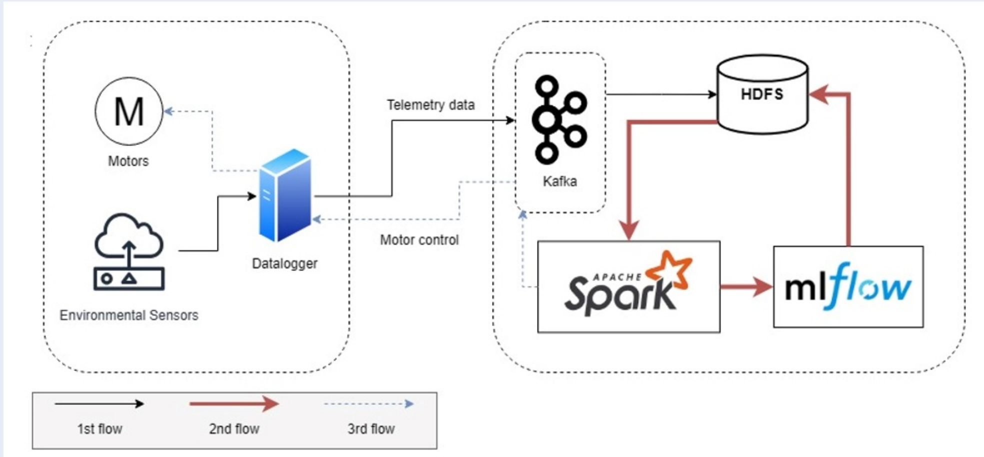

Figure 1: Abstract design of the system.

- MLflow is a library for managing machine learning model lifecycle.
- Delta Lake is a storage framework that integrates well with Spark and provides better performance and reliability than Hadoop.

We chose them for their well-known reputation, scalability, stability, and our prior knowledge of them, which leaves more time for us to investigate. Particularly, the system has three major working flows:

Flow 1: Environmental sensor data is routinely gathered, compiled at a local edge station, and forwarded to our centralized server to serve applications related to the data, such as environmental monitoring dashboards and crop management systems. Consequently, the data is directed to a Kafka topic to accommodate diverse applications. For our irrigation application, we utilize a Spark cluster to subscribe to the topic, retrieve data from the Kafka topic, and conduct initial processing before storing it into a Hadoop Distributed File System (HDFS) as a large Delta table.

Flow 2: During this flow, our application will load data from the HDFS to train a model from the data. The model is then sent to the MLflow server, which manages and monitors the pipeline of the models created by our application.

Flow 3: The application will infer an irrigation schedule from the environmental state of the garden. This schedule will be sent to the remote stations, where it will be used to create irrigation decision.

Algorithm 1: Decision_Tree_Build

Input: Initial dataset D_0, tree root T

if *stopping condition* is False:
  foreach column c in D_0:
    foreach value val in c:
      split D_0 into 2 parts r, l based on val
      if r, l not empty:
        calculate IG
      end if
    end foreach
  end foreach
  split data into 2 parts D_r, D_l and create 2 nodes N_r, N_l based on val that associates with the highest IG
  N_r=Decision_Tree_Build(D_r, N_r)
  N_l=Decision_Tree_Build(D_l, N_l)
end if
return T

Figure 3: Conventional Decision Tree.

System Deployment

Our method is conducted on a 5000m^2 garden based in Dong Thap, which has a predominant crop, namely mangoes, planted in the garden. The garden is monitored by an array of 40 earth sensors, one water sensor, one air sensor, and an irrigation motor that we can remotely control the amount of water. Those sensors collect environmental data, such as pH, soil moisture, air moisture and temperature,... every minute and send it to the local station for accumulation and preprocessing, The data is then sent back to our server for analysis.

For distributing models to users, we specify two types of users, denoted as black and white dots in Figure 2. One of them (white dots) actively uses the models and

sends data back to our server to retrain a model tailored to the user's environment and the user's irrigation behavior, while the other type of user (black dots) only uses the
models created by other users (white dots) for predicting irrigation decisions based on the assumption that models created by other users are also suitable if those users are geographically adjacent to that user, implying similar environmental characteristics.

METHOD

Conventional Decision Tree

Recall that to construct a decision tree from a dataset D_0 with N variables d_1, d_2,…,d_N and a label L, we must first calculate the entropy for each variable, which is, how well for any variable in a node can be used to split the data of that node:

$$E(D) = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

where p represents the ratio between a label count to its class's count.

Information gain (IG) is then calculated for each splitting variable V to determine the highest IG

$$IG(D,V) = E(D) - \sum_{a \in V} \frac{|D_a|}{|D|} E(D_a)$$

Algorithm 1 (Figure 3) demonstrates the pseudocode of constructing a tree this way.

Proposed Decision Tree

For a dataset D_1 that contains N variables d_1, d_2, … d_N, a set A of actions a_1, a_2, …a_M and its corresponding set R of rewards r_1, r_2, … r_L, we propose another approach to split the dataset into two datasets D_{1r} and D_{1l} based on the value of a variable d_n so that the reward rate of choosing a single action in D_{1r} or D_{1l} that is higher than the sum of rewards in D_1 divided by the size of D_1. Algorithm 2 (Figure 5) demonstrates the pseudocode of constructing a tree based on score.

This algorithm first searches through all variables and for each variable var, searches through every unique value val_{var} and attempts to make a split based on that value:

If var only has binary or categorical values, the data is split based on whether each value $v_0 = val_{var}$ or $v_0 \neq val_{var}$.

If var only has continuous values, the data is split based on whether each value $v_0 \geq val_{var}$ or $v_0 < val_{var}$.

For every split, sum the reward based on that split. The split that returns the highest reward will be used and two new leaf nodes are created, each having one part of the split data from the node above it.

Irrigation time prediction model in the system

In the real scenario, telemetry data is continuously streamed into our system every minute. To integrate new data into our model, we propose a lifecycle for our model to comply with the system's constraints. Figure 4 briefly shows the lifecycle of the system with more concentration on the model lifecycle, which is comprised of 3 major working flows:

- Flow 1: The model will make a prediction, which is one of the available actions to which it is limited, based on the environment states of the last few hours, and trigger the pump motor to run for the predicted period. The predicted action will also be stored for later use.
- Flow 2: The environmental sensors will send back telemetry data to the server. The data will then be used to create rewards based on a policy stored on the server. Both types of data will be stored after that.
- Flow 3: The telemetry data and reward data combined with the action data will be extracted in batches to train a new model with better adaptability to the environment and the policy.

EXPERIMENTAL RESULTS

Metrics, policy and data

For the simplicity of the model, we use 4 environmental metrics: soil humidity (SH), soil temperature (ST), air humidity (AH), and air temperature (AT); along with 1 temporal metric to evaluate the model based on how well it can learn our policy, which farmers and experts in agriculture suggest. Our model will be introduced with four actions: Do not irrigate, irrigate for 10 minutes, irrigate for 20 minutes, and irrigate for 30 minutes.

Our policy, as shown in Table 1, consists of the environment metrics, from which it will return four probabilities of getting a reward, corresponding to the four actions listed above. We also introduce noise to our calculation at the rate of 20%.

Our data is collected from sensors installed at a test garden in Dong Thap from 1/2023 to 8/2023, which is comprised of many metrics of the earth, air, and water environment. Tables 2, 3, 4 and 5 demonstrate the covariance matrices of our dataset but sampled into subsets with sizes of 10000, 50000, 100000, and the initial size, respectively. Due to similarities between these matrices, we will only use the 10000 datapoints subset for evaluation because increasing the dataset size does not allow our model to learn further significantly.

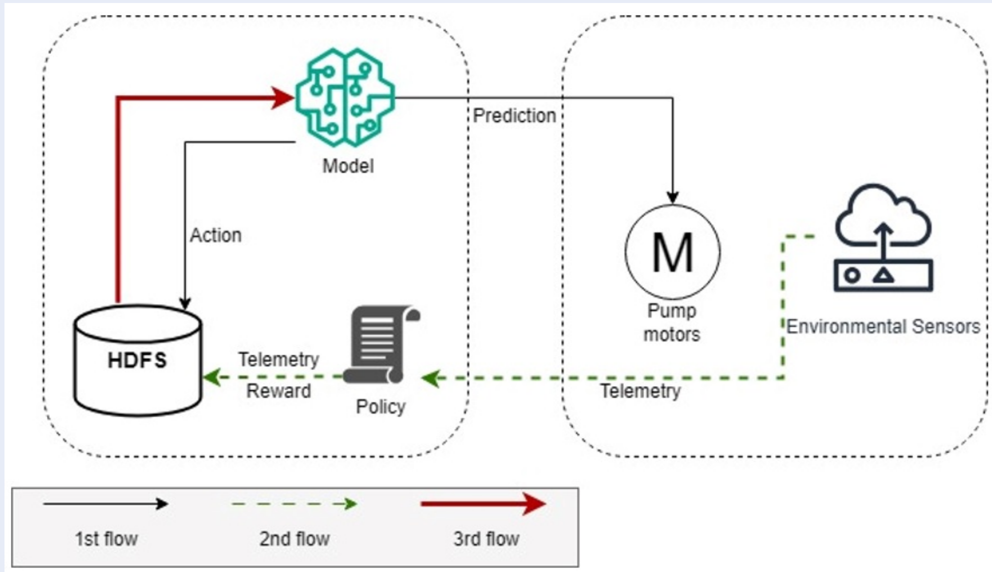

Figure 4: Lifecycle of the model in the system.

Algorithm 2: Score_Based_DT

Input: Initial dataset D_1, tree root T

if *stopping condition* is False:
  foreach variable column c in D_1:
    foreach value val in c:
      foreach action a in D_1:
        rpa = reward per action a rate for all value to the left and right of val
      end foreach
      rpa_{max}: the highest value of rpa's that associates with a_{max}
      r_{incr} = difference between reward rate of D_1 and rpa_{max}, multiplied by size of D_1
    end foreach
  end foreach
  split data into 2 parts D_r, D_l and create 2 nodes N_r, N_l based on val of c that associates with the highest r_{incr}
  N_r= Score_Based_DT(D_r, N_r)
  N_l = Score_Based_DT(D_l, N_l)
end if
return T

Figure 5: Score-Based Decision Tree.

Table 1: Policy for calculating rewards.

Actions Metrics	0 mins	10 mins	20 mins	30 mins
$24 \leq SH \leq 26$	0.5	0.45	0.25	0.3
$SH \leq 24$	0.1	0.25	0.3	0.35
$SH \geq 26$	0.4	0.2	0.1	0.05
$ST \leq 26$	+0.1/pt	+0.1/pt	+0.1/pt	+0.1/pt
$ST \geq 28$	-0.1/pt	-0.1/pt	-0.1/pt	-0.1/pt
$AH \leq 80$	+0.05/pt	+0.05/pt	+0.05/pt	+0.05/pt
$AH \geq 90$	-0.05/pt	-0.05/pt	-0.05/pt	-0.05/pt
$AT \leq 26$	+0.05/pt	+0.05/pt	+0.05/pt	+0.05/pt
$AT \geq 31$	-0.05/pt	-0.05/pt	-0.05/pt	-0.05/pt

Table 2: Covariance matrix for data size of 10000.

	SH	AH	ST	AT
SH	0.032855	0.008108	-0.010476	-0.005236
AH	0.008108	0.076122	-0.024644	-0.051532
ST	-0.010476	-0.024644	0.043014	0.033396
AT	-0.005236	-0.051532	0.033396	0.047451

Table 3: Covariance matrix for data size of 50000.

	SH	AH	ST	AT
SH	0.032026	0.007855	-0.009361	-0.004651
AH	0.007855	0.076886	-0.025839	-0.051203
ST	-0.009361	-0.025839	0.043341	0.033761
AT	-0.004651	-0.051203	0.033761	0.046515

Table 4: Covariance matrix for data size of 100000.

	SH	AH	ST	AT
SH	0.032374	0.008361	-0.01006	-0.005144
AH	0.008361	0.077220	-0.025342	-0.051285
ST	-0.010006	-0.025342	0.043128	0.033329
AT	-0.005144	-0.051285	0.033329	0.046407

Table 5: Covariance matrix for the full dataset.

	SH	AH	ST	AT
SH	0.032248	0.008191	-0.009732	-0.004938
AH	0.008191	0.077060	-0.025484	-0.051307
ST	-0.009732	-0.025484	0.043059	0.033430
AT	-0.004938	-0.051307	0.033430	0.046512

Evaluation

For evaluation, our test dataset will use data from July and August, the rest of the initial dataset is split 70-30 for the training dataset and testing dataset, respectively. The reward for random actions which are filled in the dataset is approximately 29.19 per 100 data points and the maximum reward, which is, the highest possible reward that our agent can achieve based on our policy for the dataset, is 61.6 per 100 data points.

Table 6 describes our reward based on the model's prediction, model accuracy, which is calculated from our model's gathered reward and our data's highest possible reward, which is ruled via our policy, and uplift, which shows how much higher reward our model gained compared to the reward from the dataset. Also in this table, we compare our baseline decision tree (DT) model with our random forest (RF) model at different tree counts (tc), tree depths (d), and data sizes.

From the table, we can see that the models bring about very high accuracy, while the decision tree models have almost comparable performance to random forest ones. It can also be concluded from it that the tree depth of 3 is the sweet spot for optimal performance in both types of models.

DISCUSSION AND CONCLUSION

This paper proposes a simple reinforcement learning method that uses a decision tree as the policy to be learned by the agent for the irrigation scheduling problem. Using the dataset collected from the sensors placed in an actual environment, combined with a static policy to calculate the reward, we expect that the model should make actions that bring back more reward, without the knowledge of the given policy. From our evaluation, the model has managed to learn the policy from the reward inferred from that policy. However, our work still has the following drawbacks, which are also our future work:

- Our method requires abandoning the old model and training a new one to adapt to new data, which is still compute-intensive to some extent. In the future, we will try to refactor the model to learn from new data incrementally.
- Our approach is based on a static policy. Hence, for each stage of growing a tree, farmers' and experts' suggestions are required to construct a new policy for it.

ACKNOWLEDGEMENTS

This research was conducted within 58/20-DTDL.CN-DP Smart Village project sponsored by the Ministry of Science and Technology of Vietnam and "Developing a data streaming platform used in urban and rural areas" project sponsored by TIST Lab.

We gratefully acknowledge the valuable support and resources provided by HPC Lab at HCMUT, VNU-HCM.

We would like to express our gratitude to Hoang-Loc La for his valuable resources, insights, and methodological support.

CONFLICT OF INTEREST

The authors declare that they have no competing interests.

AUTHORS CONTRIBUTION

Hung Phuc Dinh: Conceptualization, Methodology, Formal Analysis, Investigation, Writing – Original Draft.

Nguyen Tran Tho: Supervision, Funding Acquisition.

Trung Dang Anh: Supervision, Funding Acquisition.

Nam Thoai: Conceptualization Validation, Resources, Writing – Review & Editing, Supervision, Project Administration, Funding Acquisition.

REFERENCES

1. Viet TN, Giac TN. The relationship between green, sustainable agricultural development, climate change adaptation, and digital transformation in Dong Thap, Vietnam. Lex Humana. 2022 Jul;14(1):486-502;Available from: https://seer.ucp.br/seer/index.php/LexHumana/article/view/2360.
2. Dang H, Martin S. Climate change impacts on water resources in Vietnam: adaptation strategies for agriculture and hydropower - a review. TNU Journal of Science and Technology. 2018;181(5):197-204;.
3. Ho VTT, Hoang ANT, Thu TD, Thi ND. Design and fabrication of smart irrigation technology using soil moisture sensor system for vine in Ninh Thuan, Vietnam. IOP Conf Ser Mater Sci Eng. 2020 Dec;991(1):012143;Available from: https://doi.org/10.1088/1757-899X/991/1/012143.
4. Chen YA, Hsieh WH, Ko YS, Huang NF. An ensemble learning model for agricultural irrigation prediction. In: 2021 International Conference on Information Networking (ICOIN), IEEE. 2021 Jan. pp. 311-316;Available from: https://doi.org/10.1109/ICOIN50884.2021.9333852.
5. Sun L, Yang Y, Hu J, Porter D, Marek T, Hillyer C. Reinforcement learning control for water-efficient agricultural irrigation. In: 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), IEEE. 2017 Dec. pp. 1334-1341;Available from: https://doi.org/10.1109/ISPA/IUCC.2017.00203.
6. D'haeze D, Deckers J, Raes D, Phong TA, Minh Chanh ND. Over-irrigation of Coffea canephora in the Central Highlands of Vietnam revisited. Agric Water Manag. 2003 Dec;63(3):185-202;Available from: https://doi.org/10.1016/S0378-3774(03)00181-1.

Table 6: Model accuracy and performance uplift.

Metric Model	Predicted	Accuracy	Lift
DT(d=2)	58.8635	95.56%	201.66%
DT(d=3)	59.5217	96.62%	203.91%
DT(d=4)	57.8916	93.98%	198.32%
RF(tc=5, d=2)	58.3938	94.80%	200.05%
RF(tc=5, d=3)	59.1060	95.95%	202.49%
RF(tc=10, d=2)	58.6840	95.27%	201.04%
RF(tc=10, d=3)	59.1719	96.06%	202.71%

7. La HL, Quang BP, Tho NT, Anh TD, Thoai N. A semi-supervised method for smart irrigation using real data in the Mekong Delta. In: 2022 International Conference on Advanced Computing and Analytics (ACOMPA), IEEE. 2022 Nov. pp. 27-33;Available from: https://doi.org/10.1109/ACOMPA57018.2022.00011.
8. Domingos P, Hulten G. Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. New York, NY, USA: ACM; 2000 Aug. pp. 71-80;Available from: https://doi.org/10.1145/347090.347107.
9. Féraud R, Allesiardo R, Urvoy T, Clérot F. Random forest for the contextual bandit problem. In: Artificial intelligence and statistics. 2016. pp. 93-101;.
10. The Apache Software Foundation. Apache Hadoop. 2023 [Accessed 2023 Nov 08];Available from: https://hadoop.apache.org.
11. The Apache Software Foundation. PySpark: Python Front End for 'Apache Spark'. 2023 [Accessed 2023 Nov 08];Available from: https://spark.apache.org.
12. The Apache Software Foundation. Apache Kafka. 2023 [Accessed 2023 Nov 08];Available from: https://kafka.apache.org.

Tạp chí Phát triển Khoa học và Công nghệ – Engineering and Technology 2024, ():1-1

Open Access Full Text Article

Tạo cây quyết định dựa trên điểm: Hướng tiếp cận đơn giản tới bài toán tưới tiêu thông minh sử dụng dữ liệu thật

Đinh Phúc Hưng[1], Trần Thọ Nguyên[2], Đặng Anh Trung[2], Thoại Nam[1,2,*]

[1]*Phòng thí nghiệm Tính toán Hiệu năng cao, Khoa Khoa học và Kỹ thuật Máy tính (HPC Lab), Trường Đại học Bách Khoa Thành phố Hồ Chí Minh (HCMUT)*

[2]*Đại học Quốc gia Thành phố Hồ Chí Minh, Việt Nam*

Liên hệ

Thoại Nam, Phòng thí nghiệm Tính toán Hiệu năng cao, Khoa Khoa học và Kỹ thuật Máy tính (HPC Lab), Trường Đại học Bách Khoa Thành phố Hồ Chí Minh (HCMUT)

Đại học Quốc gia Thành phố Hồ Chí Minh, Việt Nam

Email: namthoai@hcmut.edu.vn

Lịch sử

- Ngày nhận: 28-9-2023
- Ngày chấp nhận: 29-3-2024
- Ngày đăng:

DOI :

Check for updates

TÓM TẮT

Ngoài tính hiệu quả, tính thực tiễn và tiết kiệm rất được coi trọng khi triển khai những hệ thống tưới tiêu tự động. Nhận thức về những yêu cầu này càng tăng lên khi nguồn tài nguyên nước ngày càng khan hiếm, đặc biệt là ở nhiều vùng nông nghiệp tại Việt Nam. Một lượng lớn nỗ lực được dành cho việc tạo ra các phương pháp giải quyết những vấn đề này và có thể được phân loại thành hai nhóm: học giám sát và học tăng cường. Phương pháp học giám sát thường dựa vào đầu vào từ kiến thức của người nông dân và chuyên gia. Tuy nhiên, những cách tiếp cận đó có thể sai sót do người nông dân tưới quá nhiều hoặc quá ít, dẫn đến lãng phí tài nguyên và chi phí sản xuất. Mặt khác, phương pháp học tăng cường đã được chứng minh là hiệu quả nhờ khả năng thích ứng nhanh chóng với những thay đổi hoặc xu hướng thay đổi của môi trường. Bất chấp điều đó, yêu cầu về một hệ thống mô phỏng đáng tin cậy và sự đầu tư về thời gian thực hiện các bước luyện mô hình đã khiến nó phi thực tế khi sử dụng ngoài thế giới thực. Việc triển khai các phương pháp như vậy còn gặp phải những trở ngại về mặt tài nguyên và kiến trúc hệ thống. Bài báo này đề xuất một phương pháp kết hợp giữa hai nhóm phương pháp trên nhằm điều chỉnh môi trường đến trạng thái mong muốn. Bài báo này cũng trình bày tổng quan đặc trưng của môi trường và kiến trúc hệ thống mà phương pháp đề xuất sẽ được triển khai theo cách mà phương pháp đó có thể tương tác với các trạng thái của môi trường. Hướng tiếp cận của chúng tôi cũng có thể được triển khai trên các hệ thống có nguồn tài nguyên tính toán hạn chế, không yêu cầu việc huấn luyện trong môi trường ảo và giảm tối thiểu sự tác động từ con người. Việc đánh giá hiệu quả của phương pháp đề xuất cũng được trình bày và cho thấy sự cải thiện rõ rệt của phương pháp trên một tập dữ liệu được thu thập từ môi trường.

Từ khoá: Tưới tiêu thông minh, học máy, học giám sát, học tăng cường

Trích dẫn bài báo này: Hưng D P, Nguyên T T, Trung D A, Nam T. **Tạo cây quyết định dựa trên điểm: Hướng tiếp cận đơn giản tới bài toán tưới tiêu thông minh sử dụng dữ liệu thật**. *Sci. Tech. Dev. J. - Eng. Tech.* 2024; ():1-1.