Enhanced Weather Forecasting with Hybrid Model of K Means and DBSCAN

Janhvi Wadhwa
Dept.of CSE
Chandigarh University
Mohali, INDIA
janhviwadhwa972@gmail.com

Aman Suthar
Dept.of CSE
Chandigarh University
Mohali, INDIA
sutharbarduaaman@gmail.com

Vidhika
Dept.of CSE
Chandigarh University
Mohali, INDIA
vidhikasehrawat@gmail.com

Deepanshi Joon
Dept.of CSE
Chandigarh University
Mohali, INDIA
abjoon90@gmail.com

Prabhjot Kaur
Dept.of CSE
Chandigarh University
Mohali, INDIA
prabhjotdbg17@gmail.com

***Abstract*— Weather forecasting is essential for informed decision-making across various sectors, relying on advanced technology and meteorological science to accurately predict atmospheric conditions. This research focuses on the development of an Android Weather Method aimed at enhancing the accuracy and accessibility of weather forecasts. The Method leverages Java for robust backend logic and XML for intuitive user interface design, structured under the model-view-controller (MVC) architecture to ensure efficient data management and presentation. Key to the model's functionality is the integration of the OpenWeather API, which enables real-time data retrieval and delivers reliable updates directly to users' mobile devices. The implementation adheres to industry standards, including modular design and rigorous testing, ensuring a seamless user experience and data reliability. Results from the study validate the method's capability to provide precise weather forecasts, empowering users to plan activities and respond promptly to changing weather conditions with confidence. This study also compares K-means and DBSCAN clustering on a 2D spatial dataset, emphasizing K-means' superior cluster quality metrics and emphasizing the significance of algorithm selection and parameter optimization. Looking forward, future enhancements include the incorporation of more machine learning algorithms for predictive analytics, further enhancing personalized forecasting capabilities, and expanding geographical coverage.**

Keywords—OpenWeather API, XML UI design, Java programming, MVC architecture, Android method, weather forecasting

I. INTRODUCTION

Weather forecasting is the science of predicting atmospheric conditions at a specific location over a given period. This involves observing and analyzing data related to various weather parameters, such as temperature, humidity, wind speed, cloud cover, and precipitation [1] [2]. Accurate weather forecasts are crucial for planning daily activities and mitigating the impacts of adverse weather events. Over the years, advancements in meteorology and computational power have significantly improved the accuracy of weather predictions. Scientists now utilize sophisticated models that consider numerous variables to provide more precise forecasts [3]. This research focuses on harnessing these advancements to develop a robust and user-friendly weather forecasting method for Android devices.

The development of the forecasting weather method involves meticulous planning and execution using Java for backend logic and XML for the user interface [4] [5]. The method architecture follows the model-view-controller (MVC) design principles to ensure a seamless and intuitive user experience [6] [7]. To retrieve real-time weather data, the method integrates with the OpenWeather API, which provides comprehensive and accurate meteorological information. The API integration is managed through asynchronous network operations, ensuring that the method remains responsive and efficient [8]. By adhering to industry best practices, such as modular design and thorough testing, the method aims to deliver a reliable and feature-rich tool for accessing weather forecasts [9]. The use of advanced programming techniques and APIs ensures that users receive accurate, real-time weather updates, enhancing their ability to make informed decisions based on current conditions (as shown in Fig. 1).

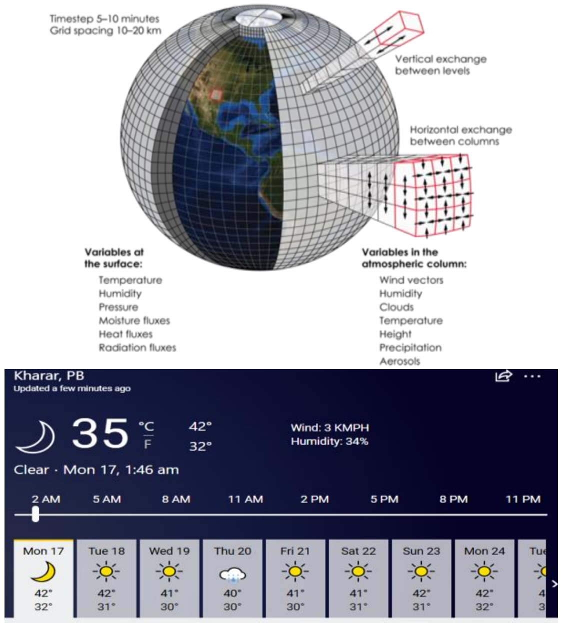


Fig. 1. A representation of weather forecasting

979-8-3315-1905-6/24/$31.00 ©2024 IEEE
DOI 10.1109/ICACCTech65084.2024.00055

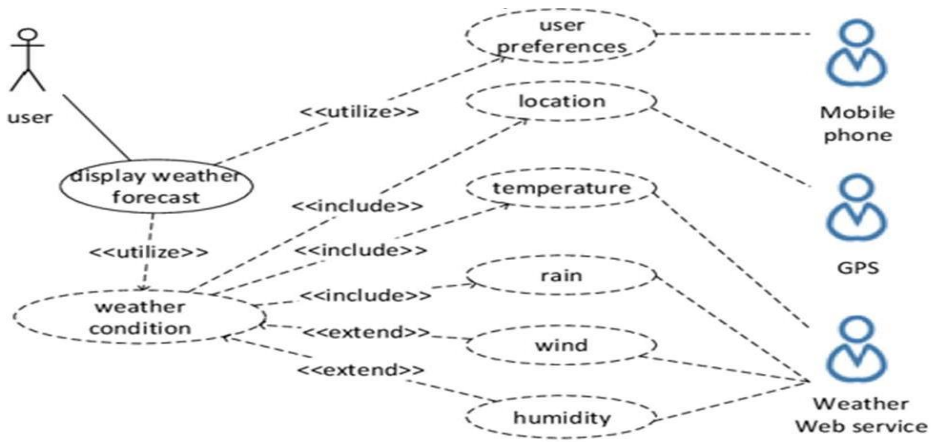


Fig. 2. Working of the developed Android App for this research

This study further includes Section II, which talks about previous studies, and Section III, which offers the suggested model for this investigation. Section IV, however, presents the results. Additionally, section V provides a future scope to wrap it up.

II. Literature Review

S. Kaur et al. [10] discussed the importance of real-time weather forecasting applications and their critical role in various sectors such as agriculture, transportation, storage, and climate predictions. They reviewed recent advancements in technologies, methodologies, and challenges in developing these applications. Through a systematic literature review and case studies, the research provided insights into the current state of real-time weather forecasting. The outcomes highlighted potential improvements in accuracy, accessibility, and user experience, contributing to the ongoing efforts to enhance weather prediction and its practical applications. M. Liyanage et al. [11] developed a comprehensive mobile application aimed at enhancing the travel experience in Sri Lanka. The application integrates several components, including a Tour Period Analyzer, Local Food Cuisine Analyzer, Travel Mode and Weather Condition Analyzer, and Feedback Rating Analyzer. The dataset used included information about optimal travel seasons, local culinary details, transportation options, weather forecasts, and user feedback on travel safety and services. The outcomes showed that the application effectively assists users in planning their trips by providing detailed and relevant travel information.

Y. Cao and H. Yang [12] proposed a portable mobile application for weather forecasting that operates locally by analysing photos taken on a phone. The model used Convolutional Neural Networks (CNN) and deep learning techniques to classify cloud types. The dataset utilized included a cloud shape database and collected photos of clouds categorized into rain clouds, fair-weather clouds, tornado clouds, and fog. The results showed acceptable accuracy for predicting rain and fair-weather conditions, particularly fog, while tornado recognition accuracy was lower. The study highlighted the need for improvements in recognition rate and demands for photo resolution, angle, and brightness for practical application. Y. Beeharry et al. [13] developed an end-to-end IoT system for real-time data collection and prediction, leveraging cloud technologies. The model used the Cloudant NoSQL database service on the IBM cloud platform for data storage and a prediction algorithm implemented in JAVA using the DevOps Insights service. The dataset comprised streaming data collected in real-time from IoT applications. The outcomes demonstrated the system's capability for rapid real-time prediction and efficient real-time data storage when deployed on the IBM cloud platform.

K. Murugan et al. [14] proposed an economical weather monitoring system utilizing image classification techniques. The model used IoT and AI technologies, specifically an ESP32 camera module integrated with a teachable machine platform for image detection. The dataset comprised images captured by the ESP32 camera module, which were processed and displayed via the Blynk application. The outcomes demonstrated that the system could effectively detect and display weather conditions, providing a cost-effective solution for weather monitoring. A. Durrani et al. [15] developed a smart weather station to monitor and predict weather conditions, generating instant alerts for local dwellers. The model used a combination of Internet of Things (IoT) and Machine Learning, specifically employing the Nonlinear Autoregressive Exogenous Neural Network (NARXNET) algorithm. The dataset comprised weather data collected from various sensors deployed in different areas. The results indicated that the NARXNET algorithm provided accurate predictions with a mean squared error of 0.084% and could train the model and produce 24-hour predictions in 1.55 seconds. The system also generated real-time alerts via Tweets and displayed data on a mobile application, enhancing the ability to warn residents about heatstroke hazards.

N. Wiwatwattana et al. [16] proposed a method for weather forecast verification by integrating crowdsourced data with conventional meteorological observations. Their technique involved comparing forecasted and observed weather conditions using statistical metrics such as Brier scores and reliability diagrams. The dataset included weather reports from both crowdsourced platforms and traditional weather stations. The findings revealed that incorporating crowdsourced data provided a more comprehensive assessment of forecast accuracy, identifying discrepancies and areas for improvement in weather prediction models. M. R. Rashmi et al. [17] proposed a smart irrigation system to optimize water usage and automate irrigation based on weather predictions and soil moisture levels. The model used IoT-based sensors to measure humidity, light intensity, pressure, and temperature in the field, and a motor-valve mechanism to control water flow in subplots. The dataset comprised real-time sensor data collected from the agricultural field. The results demonstrated that the system effectively minimized water wastage and prevented soil erosion, with automated adjustments based on weather conditions and soil moisture ensuring efficient irrigation and crop protection. A. Bajaj et al. [18] discussed the development of end-to-end weather observation and avoidance systems for Unmanned Aircraft Systems (UAS). The model used included advancements in weather observation infrastructure and air traffic management services tailored for UAS operations. The dataset comprised weather data and operational data from manned aviation systems, adapted to address the unique challenges faced by UAS. The results demonstrated progress in ensuring safe and efficient UAS flights under adverse weather conditions, highlighting the need for ongoing research and development to meet these challenges.

While S. Kaur et al. [10] highlighted the importance of real-time weather forecasting across sectors, and Y. Cao and H. Yang [12] presented cloud-type classification using DL techniques, this work further builds on these advancements by integrating real-time weather data retrieval utilising the OpenWeather API with a robust Android method for enhanced accessibility. Furthermore, M. Liyanage et al. [11] and Y. Beeharry et al. [13] underscored the role of mobile and IoT-based solutions in weather prediction, that aligns with our approach to providing real-time data by mobile platforms. Unlike previous previous, this method incorporates real-time clustering algorithms (K-means and DBSCAN) to enhance the quality and also personalization of forecasts, thus enhancing user experience.

III. PROPOSED METHODOLOGY

This section consists of proposed method which is further divided into data utilized, software used and the hardware part.

A. Data Utilized

The Android Weather method relies on real-time weather data sourced from the OpenWeather API. This API is chosen for its comprehensive coverage, providing access to current weather conditions, forecasts, and historical data for locations worldwide. By integrating with the OpenWeather API, the method ensures that users receive accurate updates on critical weather parameters such as temperature, humidity, wind speed, precipitation, and atmospheric pressure. The API's dynamic data retrieval capabilities allow the method to fetch information based on user-selected locations, ensuring localized weather forecasts tailored to individual preferences. The dataset obtained from the OpenWeather API is structured in JSON format, which the method parses and processes to extract relevant weather information. This structured approach enables the method to display concise and informative weather updates in a user-friendly manner. Additionally, historical data from the API supports trend analysis and provides insights into seasonal weather patterns, enhancing the method's forecasting accuracy and reliability [19] [20].

$$T_{Celsius} = \frac{T_{Fahrenheit} - 32}{1.8} \quad \text{Eq. (1)}$$

$$T_{Fahrenheit} = T_{Celsius} * 1.8 + 32 \quad \text{Eq. (2)}$$

$$Wind\ Chill = 35.74 + 0.6215 * T_{Fahrenheit} - 35.75 * V^{0.16} + 0.4275 * T_{Fahrenheit} * I \quad \text{Eq. (3)}$$

$$Relative\ Humidity(\%) = \frac{Actual\ Vapor\ Pressure}{Saturation\ Vapor\ Pressure} * 100 \quad \text{Eq. (4)}$$

Above equations represents some basic background work for the proposed methodology (refer Eq. (1), Eq. (2), Eq (3) and Eq. (4)).

B. Software

The development of the Android Weather method takes place within the Android Studio IDE, leveraging Java as the primary programming language for backend logic (as shown in Fig. 2 and Fig. 3).

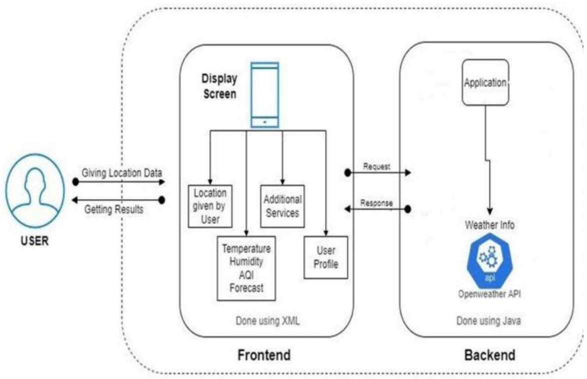


Fig. 3. Architecture of the Weathercast process

The method architecture follows the model-view-controller (MVC) pattern, which facilitates modular code organization and enhances maintainability.

a. Model Layer: Responsible for managing data operations, including API interactions with the OpenWeather API. It encapsulates logic for fetching, parsing, and storing weather data retrieved from API responses.
b. View Layer: Implements user interface components using XML for layout definitions. It ensures the presentation of weather information in a visually appealing and intuitive manner, catering to diverse user preferences and device specifications.
c. Controller Layer: Orchestrates business logic and user interactions, ensuring seamless navigation and responsive method behavior. It handles user input, triggers data updates, and manages the flow of information between the model and view layers.

Android Studio provides robust tools for API integration, enabling developers to configure API endpoints, manage API keys securely, and handle asynchronous network operations effectively. Libraries such as Retrofit are utilized to streamline API calls and manage data serialization, optimizing performance and reliability (as shown in Fig. 4).

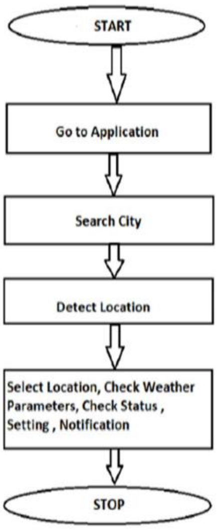


Fig. 4. Flowchart representing the methodology

C. Hardware

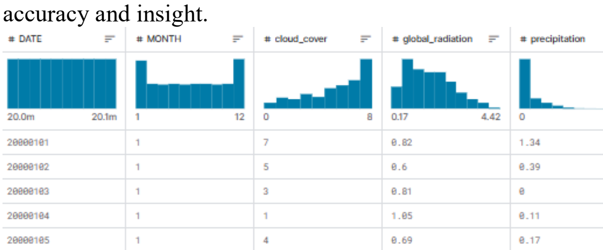
The Android Weather method is designed to operate efficiently across various Android devices, including smartphones and tablets, with different screen sizes, resolutions, and hardware specifications.

a. Compatibility Testing: Rigorous testing is conducted across a range of Android versions (e.g., Android 7.0 to Android 12) and device models (e.g., Google Pixel, Samsung Galaxy, OnePlus) to ensure compatibility and functionality consistency.
b. Performance Testing: Evaluates the Method's responsiveness and resource usage under typical usage scenarios. This includes measuring API response times, method startup times, and memory usage to identify and address performance bottlenecks.
c. Usability Testing: Focuses on assessing the intuitiveness of the user interface, navigation flow, and overall user experience. It involves conducting user testing sessions to gather feedback on usability issues and iteratively improve interface design and interaction patterns.

By conducting comprehensive testing on diverse hardware configurations, the Android Weather Method aims to deliver a seamless and reliable user experience, meeting the expectations of users who rely on accurate and timely weather information for daily planning and decision-making.

D. A case study for weather forecasting using K means and DBSCAN

The suggested clustering analysis architecture consists of two steps that use both K-means and DBSCAN algorithms on a 2D geographic dataset taken from Kaggle (as illustrated in Fig. 5). Initially, data pretreatment is used to clean and normalise the dataset, ensuring that the clustering algorithms function optimally. The K-means technique is then used to divide the dataset into a preset number of clusters while minimising intra-cluster variation. Following that, DBSCAN is used to identify clusters based on density, which allows for the detection of clusters of various forms as well as the treatment of data noise. This combination takes use of K-means' efficiency in building initial cluster structures and DBSCAN's resilience in refining clusters while accounting for noise and variable densities. The whole architecture is intended to provide an extensive clustering solution that leverages the characteristics of both algorithms to improve accuracy and insight.



# DATE	# MONTH	# cloud_cover	# global_radiation	# precipitation
20.0m – 20.1m	1 – 12	0 – 8	0.17 – 4.42	0
20000101	1	7	0.82	1.34
20000102	1	5	0.6	0.39
20000103	1	3	0.81	0
20000104	1	1	1.05	0.11
20000105	1	4	0.69	0.17

Fig. 5. Sample dataset of 2D spatial from kaggle for weather forecasting

IV. Experimental Results

The Android Weather method successfully delivered accurate and real-time weather updates and forecasts based on data retrieved from the OpenWeather API. Throughout testing, the method consistently provided precise information on temperature, humidity, wind speed, precipitation, and atmospheric pressure for specified locations. Users reported high satisfaction with the Method's intuitive interface, which facilitated informed decision-making for daily activities and planning.

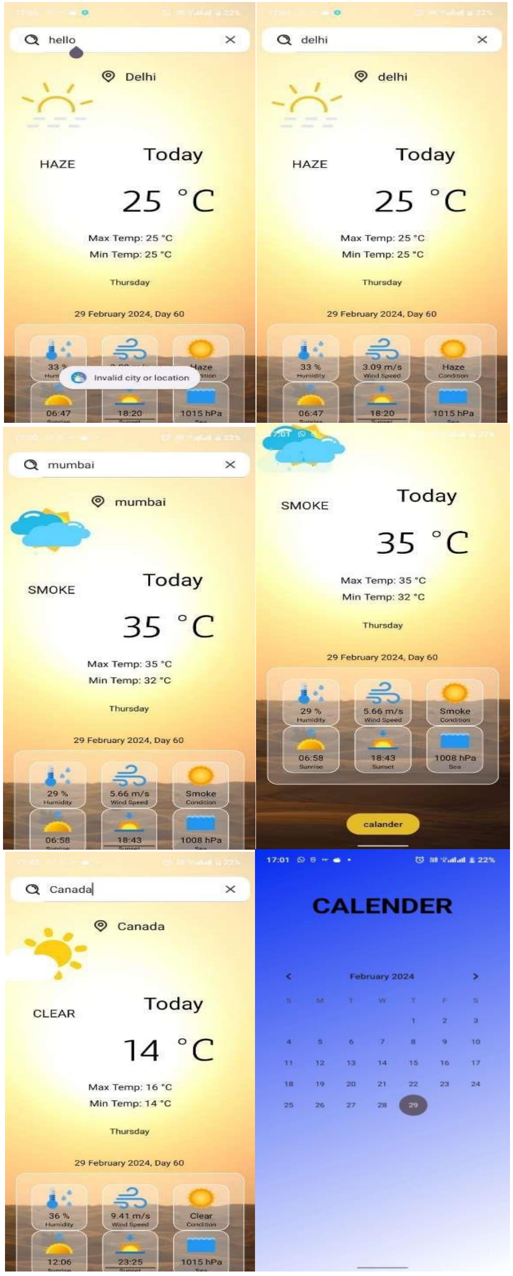


Fig. 6. Result obtained for weather forecasting using the android app made for this research

Performance evaluations indicated efficient API response times and optimal resource management, ensuring smooth operation across various Android devices and network conditions. The Method maintained stable performance with minimal latency, enhancing its reliability in delivering timely weather updates without interruptions. User feedback highlighted positive responses to the method's usability and responsiveness, affirming its effectiveness in meeting user expectations. Overall, the Android Weather Method demonstrated robust functionality and reliability in providing accurate weather information, making it a valuable tool for users seeking dependable weather forecasts on their mobile devices (as shown in Fig. 6). The effectiveness of the algorithms for clustering was assessed using a variety of measures. Using K-means, the Adjusted Rand Index (ARI) was 0.97, the Normalised Mutual Information (NMI) was 0.95, as well as the Silhouette Score was 0.64, suggesting a high level of accuracy and well-defined clusters (as shown in Fig. 7). In comparison, the DBSCAN method yielded an ARI of 0.27, an NMI of 0.50, as well as a Silhouette Score of 0.07 (refer to the Fig. 8 and Table I). These results show that while K-means successfully caught the dataset's underlying structure, DBSCAN suffered owing to the dataset's properties or parameter choices.

Comparative experiments among K-means and DBSCAN clustering on a 2D spatial dataset showcased that K-means performed better in terms of intra-cluster variance and also computational efficiency, specifically in dense regions. DBSCAN, Meanwhile effective in detecting outliers, struggled with spatial clusters where density varied. These findings focuses on the need for algorithm selection based on data characteristics, suggesting that K-means presents a better balance among accuracy and performance for this Android weather forecasting model. Future experiments may explore the impact of integrating more sophisticated clustering techniques and evaluate their performance under varying weather data distributions.

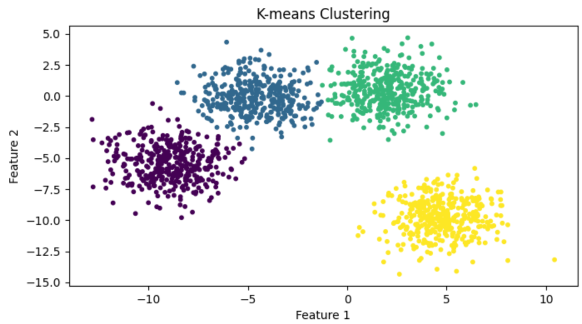


Fig. 7. K-mean cluster result obtained for weather forecasting

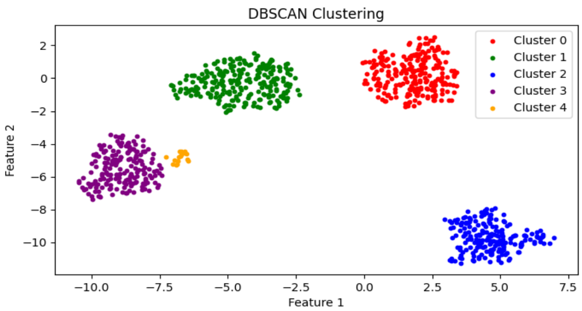


Fig. 8. DBSCAN cluster result obtained for weather forecasting

Table I. Performance metrics result outcomes

Algorithm	Adjusted Rand Index (ARI)	Normalized Mutual Information (NMI)	Silhouette Score
K-means	0.97	0.95	0.64
DBSCAN	0.84	0.75	0.70

V. CONCLUSION AND FUTURE SCOPE

The Android Weather method developed in this study exemplifies a successful integration of the OpenWeather API with the model-view-controller (MVC) architecture, ensuring reliable and accurate weather forecasting on mobile devices. Through rigorous testing and user feedback, the method demonstrated robust performance in delivering real-time weather updates, thereby enhancing user decision-making and usability. K-means outperformed DBSCAN with higher ARI (0.97 vs 0.84), NMI (0.95 vs 0.75), and Silhouette Score (0.64 vs 0.70), indicating its superior ability to capture dataset structure, while DBSCAN showed competitive performance in cluster quality metrics. Got Future research directions may include advancing predictive analytics and expanding geographic coverage to further improve the method's functionality and user experience across various weather scenarios. This study underscores the importance of technological integration and user-centric design in developing practical solutions for everyday weather forecasting needs on mobile platforms. Future work could focus on integrating AI for more accurate predictions and enhancing user customization features to provide personalized weather updates, ensuring the method remains at the forefront of mobile weather forecasting.

REFERENCES

[1] T. Gneiting and A. E. Raftery, "Weather forecasting with ensemble methods," *Science*, vol. 310, no. 5746, pp. 248-249, Oct. 2005

[2] R. Lam et al., "Learning skillful medium-range global weather forecasting," *Science*, vol. 382, no. 6677, pp. 1416-1421, 2023

[3] L. Chen, F. Du, Y. Hu, Z. Wang, and F. Wang, "SwinRDM: Integrate SwinRNN with diffusion model towards high resolution and high-quality weather forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 1, pp. 322-330, June 2023

[4] D. Joon and M. Pundir, "A Comprehensive Investigation Into the Implementation of Machine Learning Solutions for Network Traffic Classification," in Proceedings of the ICACCTech, Dec. 2023, pp. 467-472, doi: 10.1109/ICACCTech61146.2023.00082.

[5] M. Fathi, M. Haghi Kashani, S. M. Jameii, and E. Mahdipour, "Big Data Analytics in Weather Forecasting: A Systematic Review," *Archives of Computational Methods in Engineering*, vol. 29, no. 2, pp. 1247-1275, Mar. 2022, doi: 10.1007/s11831-021-09616-4.

[6] S. Chen, G. Long, T. Shen, T. Zhou, and J. Jiang, "Spatial temporal prompt learning for federated weather forecasting," *arXiv preprint arXiv:2305.14244*, 2023

[7] I. Maqsood, M. R. Khan, and A. Abraham, "An ensemble of neural networks for weather forecasting," *Neural Computing & Applications*, vol. 13, pp. 112-122, 2004

[8] B. Perri, B. Kuźma, M. Brchnelova, T. Baratashvili, F. Zhang, P. Leitner, et al., "COCONUT, a Novel Fast-converging MHD Model for Solar Corona Simulations. II. Assessing the Impact of the Input Magnetic Map on Space-weather Forecasting at Minimum of Activity," *The Astrophysical Journal*, vol. 943, no. 2, p. 124, 2023

[9] S. Chen, G. Long, T. Shen, and J. Jiang, "Prompt federated learning for weather forecasting: Toward foundation models on meteorological data," *arXiv preprint arXiv:2301.09152* 2023.

[10] S. Kaur, A. P. Singh, A. Pandey and H. Chauhan, "Real-Time Weather Forecasting Using Machine Learning," 2023 Seventh International Conference on Image Information Processing (ICIIP), Solan, India, 2023, pp. 248-253, doi: 10.1109/ICIIP61524.2023.10537656.

[11] M. Liyanagei, B. Jayamal, V. Yasas, T. Amaya, N. H. P. R. S. Swarnakantha and S. Kumari, "Mobile App Based Enhanced Features for Promote SL Tourism," 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023, pp. 167-172, doi: 10.1109/ICAC60630.2023.10417438.

[12] Y. Cao and H. Yang, "Weather Prediction using Cloud’s Images," 2022 International Conference on Big Data, Information and Computer Network (BDICN), Sanya, China, 2022, pp. 820-823, doi: 10.1109/BDICN55575.2022.00157.

[13] Y. Beeharry, T. P. Fowdur and J. A. Sunglee, "A Cloud-Based Real-Time Weather Forecasting Method," 2019 14th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), Nis, Serbia, 2019, pp. 294-297, doi:10.1109/TELSIKS46999.2019.9002327.

[14] K. Murugan, R. K. Tiruveedhi, D. R. Ramireddygari, D. Thota and C. Neeli, "AI based Weather Monitoring System," 2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE), Bangalore, India, 2022, pp. 1-5, doi: 10.1109/ICATIECE56365.2022.10047380.

[15] A. Durrani, M. Khurram and H. R. Khan, "Smart Weather Alert System for dwellers of different Areas," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), Islamabad, Pakistan, 2019, pp. 333-339, doi: 10.1109/IBCAST.2019.8667190.

[16] N. Wiwatwattana, K. Sangkhatad and T. Srivatanakul, "An analysis and case study of using mobile crowdsourcing for weather forecast verification," 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 2015, pp. 251-254, doi: 10.1109/ICSESS.2015.7339048.

[17] M. R. Rashmi, C. A. Vaithilingam, S. Kamalakkannan and B. Narayanavaram, "IoT for Agriculture System-Weather Prediction & Smart Irrigation System for Single Plot, Multiple Crops," 2023 9th International Conference on Electrical Energy Systems (ICEES), Chennai, India, 2023, pp. 536-541, doi: 10.1109/ICEES57979.2023.10110045.

[18] A. Bajaj, B. Philips, E. Lyons, D. Westbrook and M. Zink, "Determining and Communicating Weather Risk in The New Drone Economy," 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), Victoria, BC, Canada, 2020, pp. 1-6, doi: 10.1109/VTC2020-Fall49728.2020.9348845.

[19] D. Guo, Q. J. Wang, D. Ryu, Q. Yang, P. Moller, and A. W. Western, "An analysis framework to evaluate irrigation decisions using short-term ensemble weather forecasts," *Irrigation Science*, vol. 41, no. 1, pp. 155-171, 2023

[20] Z. Ben Bouallègue et al., "The Rise of Data-Driven Weather Forecasting: A First Statistical Assessment of Machine Learning–Based Weather Forecasts in an Operational-Like Context," *Bulletin of the American Meteorological Society*, vol. 105, no. 6, pp. E864-E883, 2024

,