Development of cyberthreat-resistant flood prediction systems using multi-sensor data verification[⋆]

Petro Venhersky[1,†], Orest Onyshchenko[1,†], Yaryna Kokovska[1,†] and Oleksandr Korchenko[2,*,†]

[1] *Ivan Franko National University of Lviv, 1 Universitetska str., 79000 Lviv, Ukraine*
[2] *State University of Information and Communication Technologies, 7 Solom'yanska str., 03110 Kyiv, Ukraine*

Abstract
The article describes the architecture of the intelligent flood prediction system, which provides increased resistance to cyber threats through the implementation of a multi-sensor data verification mechanism. The proposed approach is based on the analysis of the consistency of data from various sources, including satellite observations, IoT sensors and meteorological services. Describes the Proof of Concept (POC) implementation in Python using Flask to create a RESTful API, as well as sensor network emulation and integration with open data Copernicus and OpenWeatherMap. The mathematical model of data processing, algorithms for detecting anomalies, calculation of weighted indices and examples of practical application, in particular scenarios for detecting attempts to falsify data and protect the system from attacks such as data injection, are presented in detail. The results demonstrate the effectiveness of the approach to improve the reliability of flood forecasts in complex cyber threat conditions.

Keywords
flooding, cybersecurity, multi-sensory check, forecasting, Python

1. Introduction

In the 21st century, the problem of flooding became particularly acute due to climate change and urbanization processes [1]. As a result of global warming, the intensity and frequency of extreme hydrometeorological phenomena have increased, which threatens the safety of settlements, infrastructure and the environment. In this regard, flood forecasting systems have become an integral part of preventive risk management.

Along with the growth of the importance of such systems, so does their digital dependence. The data used in modern forecasting systems comes from various sources: satellite images, ground-based weather stations, IoT sensor networks, mobile applications, etc. [2–8]. However, such multi-source nature creates new vectors of cyber threats, in particular the possibility of intentional distortion or forgery of input data.

The purpose of this work is to develop the architecture of the flood forecasting system, which is able to detect and ignore potentially falsified data, thereby ensuring high reliability of forecasts in a cyber-hazardous environment. The basis of the reliability of such a system are the mechanisms of multi-sensor verification of the reliability of the received data.

2. Architecture of the proposed system

The architecture of the proposed system is built on the principles of service-oriented and modular architecture, which provides scalability, expandability and failure resistance of individual components [9]. Each of the modules performs a clearly defined function and interacts with others through standardized interfaces and APIs, which simplifies the integration of new data sources or predictive models.

⋆*CSDP'2025: Cyber Security and Data Protection, July 31, 2025, Lviv, Ukraine*
* Corresponding author.
† These authors contributed equally.
petro.venherskyy@lnu.edu.ua (P. Venherskyi); orest.onyschenko@lnu.edu.ua (O. Onyshchenko); yaryna.kokovska@lnu.edu.ua (Y. Kokovska); agkorchenko@gmail.com (O. Korchenko)
0000-0001-9808-7404 (P. Venherskyi); 0009-0008-3903-1222 (O. Onyshchenko); 0000-0002-5814-9890 (K. Khorolska); 0000-0003-3376-0631 (O. Korchenko)

The data to the system comes from various independent sources, which reduces the risk of a successful attack on one specific source:

- Medium and high resolution satellite images from open programs such as Sentinel-2 (European Copernicus program) or Landsat-8 (USA).
- Data from public and private ground weather stations, including measurements of precipitation, temperature, humidity, atmospheric pressure.
- Indicators from the network of IoT sensors located at critical points: sensors of soil moisture, water level in rivers or drainage systems, flow rate.
- If available—crowdsourcing messages from residents through a mobile application or web portal, which may indicate flooding, shore breaks or other events.

At the initial stage, all input data is sent to the collection and unification module, where formats are converted into a single data structure, values are converted to single units of measurement (for example, millimeters of precipitation or centimeters of water level), as well as pre-filtering gross errors, such as missing values or incorrect timestamps.

Next, the data goes to the validation and filtering module, which compares the obtained values with each other according to several criteria:

- Time consistency (whether there are significant deviations in the short period).
- Spatial consistency (for example, the difference between neighboring sensors and a satellite image).
- Historical patterns (comparison with typical values for a given season or month).
- Trust in the source (each source has a weighting factor that is dynamically updated).

Based on the agreed and confirmed data, a forecast module is launched, which can be implemented in two variants depending on the specific infrastructure:

- A physical hydrological model that takes into account the relief, soil type, water catchment areas and hydraulic parameters.
- A neural network for predicting time series, such as LSTM or GRU, which learns from historical data and additionally takes into account the weather forecast.

The final processing results are transferred to the notification and response module. This module has two key tasks. The first is the formation of text and graphic reports for specialists of the hydrometeorological service, local authorities or operators of engineering structures. The second is automatic notification of the population via SMS, push notification, email or integration with state public notification platforms. This module can also provide data via the REST API for visualization on the flood map in a web application or mobile application.

A feature of the architecture is the possibility of scaling: adding new sensors or satellite channels does not require changing the logic of the entire system, it is enough to connect them through the collection and unification module and configure validation rules.

In the future, it is also proposed to expand the functionality through an analytics module for automatic detection of long-term trends, a module for generating flood development scenarios, and integration with edge-computing solutions for local data preprocessing directly on devices or mini-servers in the field. The proposed approach is also partially based on the methodology presented in the work of Yaryna Kokovska [10].

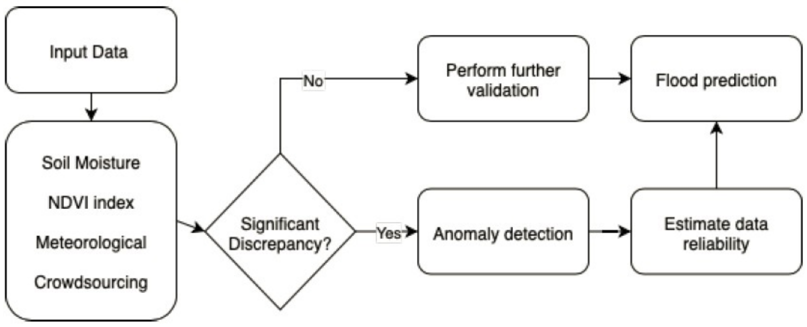

Figure 1: Multisensor data verification diagram

Thus, the architecture of the system is complex, multi-component and resistant to manipulation by intruders, since the decision to confirm or reject data is based on the mutual control of several independent sources, and not on trust in only one channel.

3. Mathematical model

The flood forecasting system is based on the principles of processing and verifying multi-source data [11]. At the input, the system receives values from a set of sensors:

$$S = \{s_1, s_2, \dots , s_n\}.$$

At each point in time to t, the s sensor provides a measured value $x_{s,t}$. Next, several successive steps of data processing are performed to ensure reliability and forecasting.

Step 1: Normalize **the data.**

For each sensor, the normalized value is calculated:

$$z_{s,t} = \frac{x_{s,t} - \mu_s}{\sigma_s}$$

Where μ_s and σ_s—the average and standard deviation according to the historical data of a particular sensor. This allows you to compare values from different sources, even if they measure different physical parameters.

Step 2: Detecting anomalies using the interquartile swing (IQR).

For each sensor, the interval of permissible values is calculated:

$$Q1_s - k \cdot IQR_s \le z_{s,t} \le Q3_s + k \cdot IQR_s,$$

Where $Q1_s$ and $Q3_s$—first and third quartiles, and $IQR_s = Q3_s - Q1_s$. The k parameter (usually k = 1.5 or k = 2) determines the sensitivity to emissions. If the normalized value goes beyond this interval, it is designated as potentially anomalous.

Step 3: Calculation of the integral reliability index.

To obtain a single risk assessment, the weight of each data source ws is taken into account, which reflects the historical reliability or expert assessment of the importance of the sensor:

$$V_t = \frac{\sum_{s \in S_s} w_s \cdot z_{s,t}}{\sum_{s \in S} w_s}.$$

The integral index V_t reflects the average normalized value, taking into account the credibility of each source.

Step 4: Predicting the risk of flooding.

The system compares the integral index with the threshold value P (for example, $P = 1.5$):

$$V_t > P \Longrightarrow \text{high risk of flooding.}$$

If V_t exceeds the threshold, the system activates the notification mechanism and triggers the response measures.

Explanation of the relationship of formulas:

Normalization converts all data into a single scale, the detection of anomalies allows you to discard unreliable values, and the calculation of the integral index gives a summary assessment, which is used to make a risk decision.

4. Example with five sources

Let us consider a practical example where data at the moment of time t are obtained from five heterogeneous sources:

- IoT sensor 1: 32.0
- IoT sensor 2: 33.0
- NDVI data: 28.5
- Weather station: 30.8
- Crowdsourcing: 35.0.

Since these sources have different measurement scales, variances, and reliability, we first normalize the data [12]. After normalization (taking into account the historical average and standard deviation of each sensor), we obtain standardized values:

$$z_{1,t} = 1.2,\ z_{2,t} = 1.4,\ z_{3,t} = 1.1,\ z_{4,t} = 1.3,\ z_{5,t} = 2.0.$$

Next, we assign specific weights to each data source to reflect their relative importance and reliability. For instance:

$$w_1 = 1.0,\ w_2 = 1.2,\ w_3 = 1.5,\ w_4 = 1.0,\ w_5 = 0.8.$$

Using these weights, we calculate the integral index V_t as the weighted average of the normalized values:

$$V_t = \frac{1.0\cdot 1.2 + 1.2\cdot 1.4 + 1.5\cdot 1.1 + 1.0\cdot 1.3 + 0.8\cdot 2.0}{1.0+1.2+1.5+1.0+0.8}$$

$$V_t = \frac{1.2+1.68+1.65+1.3+1.6}{5.5} = \frac{7.43}{5.5} \approx 1.351$$

We then compare the obtained value with a predefined threshold $P = 1.5$, which serves as a decision criterion:

- If $V_t \geq P$, the system recognizes a high risk of flooding and may activate early warning procedures.
- If $V_t < P$, as in this example ($1.351 < 1.5$), the system does not detect a significant threat at this moment.

It is important to emphasize that the contribution of each source depends both on its current anomaly (how large $z_{i,t}$ is) and on its weight w_i. For instance, although the crowdsourcing source shows the highest anomaly ($z_{5,t} = 2.0$), its lower weight (0.8) limits its influence on the integral index. Conversely, NDVI data ($w_3 = 1.5$) have a higher impact even if the anomaly is moderate.

The flexibility of this model allows the system to adjust dynamically: if in future observations one of the sensors reports extremely abnormal data, the integral index V_t could increase significantly and exceed the threshold. This triggers alerts and preventive measures, making the system an effective early warning tool.

Thus, this approach makes it possible to:

- Integrate heterogeneous data sources (physical sensors, remote sensing data, and human observations).
- Balance the influence of more and less reliable sources through weighting.
- Reduce the risk of false positives caused by errors or manipulations in individual data streams.

In summary, the method provides a robust and interpretable mechanism for real-time flood risk assessment based on multi-source data fusion.

5. Proof of concept

To explore how the proposed flood risk assessment model could work in practice, we built a Proof of Concept (POC) as a simple web service. This POC shows how different data sources can be combined, anomalies filtered out, and an integrated risk index calculated in real time, all while adding a basic layer of cybersecurity.

Main components of the POC:

- Data acquisition: The service receives data via HTTP POST requests in JSON format. Typical data sources include IoT sensors, NDVI satellite measurements, and weather station data.
- Historical context: Historical measurements help set dynamic thresholds used for detecting anomalies.
- Trust scores: Each data source is assigned a trust score based on its historical reliability and significance.
- Anomaly filtering: The system removes values that fall outside acceptable ranges, using Tukey's method (based on the interquartile range).
- Weighted integration: After filtering, the validated data are combined into a single risk index using a weighted average, where trust scores act as weights.

Technical implementation: The prototype is implemented as a Python web service built with Flask. It relies on numpy to perform statistical calculations such as percentiles and interquartile ranges (IQR). The service runs locally on port 5000 and is designed to receive JSON input from either simulated clients or real-world sensors.

Implementation (Python):

Listing 1: Prototype Flask service for flood risk calculation

```
from flask import Flask , request , jsonify
import numpy as np
from datetime import datetime

app = Flask ( __name__ )

# Trust ratios ( may be updated over time )
trust_scores = { 'iot_sensor ' : 0.9, ' satellite_ndvi ': 0.95, '
   weather_station ' : 0.85 }
```

```
# Simplified historical data for IQR
historical = {
        ' iot_sensor ' [ 29.0, 30.1, 30.3, 30.8, 31.0] ,
        ' satellite _ ndvi ' [ 28.5 , 29.2, 29.8, 30.0, 30.1] ,
        ' weather_station ' [ 29.7, 30.5, 31.0, 31.2, 31.1]
}

# Threshold value for forecast ( simplified )
FLOOD_THRESHOLD = 31.5

@app . route ( '/validate', methods =[' POST '])
def validate_ and_predict ( ) :
        data = request.json
        timestamp = data.get('timestamp')
        location = data.get('location')

# Checking for mandatory fields
if not timestamp or not location:
    return jsonify({'error': 'Missing timestamp or location'}), 400

validated = {}
anomalies = []

# Step 1: IQR for each source
for source in ['iot_sensor', 'satellite_ndvi', 'weather_station']:
    value = data.get(source)
    if value is None:
        continue
    hist = historical[source]
    q1, q3 = np.percentile(hist, [25, 75])
    iqr = q3 - q1
    lower, upper = q1 - 1.5 * iqr, q3 + 1.5 * iqr

    if lower <= value <= upper:
        validated[source] = value
    else:
        anomalies.append({
            'source': source,
            'value': value,
            'reason': 'Out of IQR bounds'
        })

# Step 2: Calculation of the integral reliability assessment
if validated:
    weighted = sum(validated[k] * trust_scores[k] for k in validated)
    total_weight = sum(trust_scores[k] for k in validated)
    final_score = weighted / total_weight
else:
    return jsonify({'error': 'All data rejected as anomalies'}), 400

# Step 3: Forecast (simplified: compare with the threshold)
flood_risk = "High" if final_score >= FLOOD_THRESHOLD else "Normal"
```

```
result = {
    'timestamp': timestamp,
    'location': location,
    'validated_data': validated,
    'anomalies': anomalies,
    'final_score': final_score,
    'predicted_flood_risk': flood_risk
}

return jsonify(result)

if __name__ == '__main__':
    app.run(port=5000, debug=True)
```

How it works: The system first validates incoming data by checking whether each value falls within dynamically calculated thresholds based on the interquartile range (IQR). Only these validated data points contribute to calculating the final risk index. Trust scores are applied so that less reliable data sources have a smaller influence on the final result. Ultimately, the computed risk score can then be compared to a predefined threshold *P*, which helps trigger early warnings if the risk becomes significant.

Results and cybersecurity impact: This proof of concept showed that even a simple statistical anomaly detection approach can noticeably reduce false alarms caused by manipulated or spoofed data. The use of trust scores adds an extra layer of protection by limiting the impact of compromised or less reliable sources. Finally, the web service architecture enables easy integration with real-time dashboards and automated alert systems, supporting practical deployment in real monitoring scenarios.

6. Cyberattack interception

In the monitoring system, IoT sensors play a crucial role in providing real-time data for environmental parameters, such as water levels in rivers or precipitation rates. For example, IoT sensor 2 transmits a water level reading of 40.0. After the system normalizes this measurement to a standardized scale, the resulting value is 8.33.

This value significantly exceeds the expected range defined by the Interquartile Range (IQR) thresholds, which are dynamically calculated based on historical and recent data distributions. Because 8.33 falls outside these robust statistical boundaries, the system flags it as an outlier and discards it to avoid skewing the overall assessment. This filtering mechanism is vital to maintaining data integrity and preventing erroneous inputs from affecting decision-making.

6.1. Potential impact of cyberattacks on sensor data

However, such anomalies can sometimes result not from sensor malfunction or environmental extremes but from deliberate cyberattacks targeting IoT infrastructure. Attackers may inject false data or manipulate sensor outputs to create misleading signals. The consequences of these cyber threats on flood prediction and risk assessment systems can be profound:

- **False Alarms**: Manipulated sensor readings that exceed thresholds may trigger unnecessary emergency responses, causing economic loss and public panic.
- **Missed Alerts**: Conversely, if attackers feed artificially low or normalized data during real flooding events, the system may underestimate the risk, delaying critical warnings and increasing the threat to human life and property.

- **Data Integrity Degradation**: Repeated cyber interference undermines trust in the sensor network, forcing reliance on fewer data sources or less frequent manual verification, reducing system responsiveness.
- **System Exploitation**: By understanding the filtering mechanisms (such as IQR-based outlier rejection), attackers may tailor inputs to bypass detection, injecting subtle but harmful distortions that degrade model accuracy over time.

6.2. Code example of cyber attack

Listing 2: Malicious client script manipulating sensor data

```
import requests

# URL of the flood prediction API
url = "http://127.0.0.1:5000/validate"

# Attacker's crafted data: values near the upper bounds of IQR,
# designed to avoid anomaly rejection but still bias final risk upward
malicious_data = {
    "timestamp": "2025-07-20T12:00:00Z",
    "location": "River Point-42",
    "iot_sensor": 31.0,       # upper bound from historical IQR
    "satellite_ndvi": 30.1,   # also near upper bound
    "weather_station": 31.1   # near upper bound
}

response = requests.post(url, json=malicious_data)
print ("Response status:", response.status_code)
print ("Prediction result:", response.json())
```

This attack leverages knowledge of the historical data distribution and trust coefficients used in the risk calculation. Since the injected values remain within acceptable IQR thresholds, the system's anomaly filter accepts them as valid. The final weighted score, after combining these biased values, may exceed the flood risk threshold (31.5), causing the system to wrongly predict a high flood risk.

6.3. System response and recalculation

By discarding obvious outlier values such as the initially received 8.33 from IoT sensor 2, the system recalculates the overall risk index:

$$V_t \approx 1.475 < 1.5 \Rightarrow \text{The risk of flooding is low.}$$

By discarding the outlier value from IoT sensor 2, the system recalculates the overall risk index

$$V_t \approx 1.475 < 1.5 \Rightarrow \text{The risk of flooding is low.}$$

This recalculation reflects a more accurate and trustworthy risk level, preserving the reliability of flood warnings. Nonetheless, the presence of cyberattacks necessitates continuous improvements in anomaly detection algorithms, cybersecurity measures, and multi-source data validation to ensure resilience against evolving threats.

7. Future implementation: Interactive user interface

Looking ahead, an important extension of the current system will be the development of an interactive user interface designed to enhance usability and situational awareness. This interface will integrate map-based visualization [13], allowing users to monitor real-time flood risk data directly on geographical maps. Through this interface, operators and analysts will be able to track conditions at specific river points, view live updates from connected sensors, and quickly identify areas where risk levels exceed predefined thresholds.

The planned system will highlight critical zones, display recent measurements and computed risk indices, and provide historical trends to support decision-making. By visualizing data spatially, the interface will make it easier to detect emerging threats, compare conditions across different locations, and issue timely alerts when necessary. This approach will not only improve clarity and accessibility but will also help bridge the gap between raw sensor data and actionable insights for flood risk management teams.

In future iterations, the interface could also incorporate additional layers, such as weather forecasts or satellite-derived indices, to provide a richer and more comprehensive view of flood risk dynamics.

Together with the backend anomaly detection and trust scoring mechanisms, this user-facing component will complete the system as a practical tool for real-time monitoring and early warning.

8. System efficiency

The efficiency of a flood risk monitoring system is vital for its practical effectiveness, particularly when managing large-scale sensor networks and processing real-time data streams. Efficiency in this context encompasses not only computational performance but also the accuracy of results, timeliness of responses, and effective management of available resources.

Regarding computational efficiency, flood prediction systems continuously process data from numerous sources, including IoT sensors, satellites, and weather stations. To achieve rapid analysis, the system relies on optimized data processing algorithms such as normalization, outlier detection through methods like IQR-based filtering, and calculation of risk indexes. Performance improvements are often gained by employing lightweight statistical methods that quickly discard anomalous data without imposing significant computational burdens. Additionally, techniques like multi-threading or distributed computing enable simultaneous data ingestion and processing, which reduces latency and improves overall throughput. Another important strategy is to perform incremental updates—recalculating risk indexes only when significant new data arrives—thus conserving computational resources compared to full recalculations for every minor data change.

Efficiency in data usage further enhances system reliability and resource allocation. By weighting sensor inputs based on their trust scores, the system effectively reduces noise and prioritizes higherquality data. The adaptive calculation of thresholds based on recent data distributions helps prevent the unwarranted rejection or acceptance of data points, maintaining data integrity. Moreover, the fusion of multiple heterogeneous data sources, such as IoT devices, satellite imagery, and weather station measurements, increases the robustness of the system. This combination compensates for possible sensor failures or cyberattacks, ensuring more consistent and reliable flood risk assessments.

Given that many sensors operate in remote or power-constrained environments, efficiency must also be considered in terms of energy and network usage. Implementing low-power communication protocols reduces the energy consumption of sensors and extends their operational lifespan. Eventdriven data transmission further conserves network bandwidth by enabling sensors to send data only when meaningful changes occur. Incorporating edge computing capabilities, where data processing happens locally on sensors or gateways, reduces the volume of data transmitted to central servers, thus lowering bandwidth requirements and decreasing latency.

Ultimately, an efficient flood risk monitoring system delivers timely warnings, providing sufficient lead time for response actions. It maintains accuracy by ensuring that resource optimization does not compromise the quality of data or the precision of risk indexes. Furthermore, the system's scalability allows it to accommodate growing sensor networks without performance degradation. Balancing these aspects of efficiency is essential for maintaining a resilient and practical flood monitoring solution capable of operating reliably under both normal conditions and potential cyber threats.

Conclusions

This paper proposes the architecture of the flood prediction system, which is resistant to cyber threats thanks to the use of multi-sensor data verification mechanisms. The use of time series, interquartile swing and weight factor analysis algorithms allows you to identify anomalous or fake data coming from sensors and external sources.

The prototype developed in Python with RESTful API confirmed the viability of the concept: even under the conditions of an attack like data injection, the system successfully identified unreliable data and excluded it from the prediction model. This made it possible to significantly reduce the risk of false triggers and improve the accuracy of flooding risk assessment.

Prospects for further research include: integration with the blockchain to ensure the immutability of the measurement history; use more complex machine learning models to dynamically adjust trust ratios; scale the prototype to work with real data streams in real time.

Thus, the proposed system has the potential to be used in modern disaster monitoring centers, increasing reliability and resistance to cyberattacks.

Declaration on Generative AI

While preparing this work, the authors used the AI programs Grammarly Pro, X-GPT-4 and Gramby to correct text grammar and Strike Plagiarism to search for possible plagiarism. After using this tool, the authors reviewed and edited the content as needed and took full responsibility for the publication's content.

References

[1] M. TajDini, V. Sokolov, P. Skladannyi, Performing Sniffing and Spoofing Attack against ADS-B and Mode S using Software Define Radio, in: IEEE Int. Conf. on Information and Telecommunication Technologies and Radio Electronics (2021) 7–11. doi:10.1109/UkrMiCo52950.2021.9716665

[2] M. TajDini, V. Sokolov, V. Buriachok, Men-in-the-Middle Attack Simulation on Low Energy Wireless Devices using Software Define Radio, in: 8th Int. Conf. on "Mathematics. Information Technologies. Education:" Modern Machine Learning Technologies and Data Science, vol. 2386 (2019) 287–296.

[3] A. Novytskyi, V. Sokolov, L. Kriuchkova, P. Skladannyi, Determining the Error Distribution of BLE Beacons at Antenna Near and Far Fields, in: Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN), vol. 4024 (2025) 133–143.

[4] O. Mykhaylova, et al., Resistance to Replay Attacks of Remote Control Protocols using the 433 MHz Radio Channel, in: Cybersecurity Providing in Information and Telecommunication Systems, vol. 3654 (2024) 98–110.

[5] V. Sokolov, et al., Method for Increasing the Various Sources Data Consistency for IoT Sensors, in: IEEE 9th Int. Conf. on Problems of Infocommunications, Science and Technology (2023) 522–526. doi:10.1109/PICST57299.2022.10238518

[6] O. Bahatskyi, V. Bahatskyi, V. Sokolov, Smart Home Subsystem for Calculating the Quality of Public Utilities, in: Cybersecurity Providing in Information and Telecommunication Systems, vol. 3421 (2023) 168–173.

[7] Y. Sadykov, et al., Technology of Location Hiding by Spoofing the Mobile Operator IP Address, in: IEEE Int. Conf. on Information and Telecommunication Technologies and Radio Electronics (2021) 22–25. doi:10.1109/UkrMiCo52950.2021.9716700

[8] L. Zhou, K. Li, Y. Wang, A Review of Flood Prediction and Monitoring based on IoT and Big Data, J. Hydrology 603 (2022) 126895. doi:10.1016/j.jhydrol.2021.126895

[9] M. Li, W. Xu, J. Tang, Detecting Sensor Data Tampering Attacks in Cyber-Physical Systems, in: Proceedings of the 2019 ACM Conf. on Computer and Communications Security (CCS), 2019, 1503–1516. doi:10.1145/3319535.3354211

[10] Y. Kokovska, Computer Modeling of the Processes of Formation of Water Streams in Channels with an Uneven Bottom, 2017. http://www.iapmm.lviv.ua/dissertation/dis_Kokovska.pdf

[11] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[12] Y. Kokovska, P. Venherskyi, Using of FEM for Modeling of Compatible Movement of Surface Kinematic Waves and River Flows, in: 16th Int. Conf. on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, 2022, 793–797. doi:10.1109/TCSET55632.2022.9767004

[13] P. Venherskyi, Y. Kokovska, Application of GIS-Technology for Modelling Motion Water in the Open Channels, in: 11th Int. Conf. Modern Problems of Radio Engineering, Telecommunications and Computer Science, 2012.