Grenze International Journal of Engineering and Technology, January Issue

Artificial Intelligence and Internet of Things Enabled Sea Water Life Saver System

V S Prathiksha[1], C Stanly Felix[2], V Keerthika[3] and B Hasika[4]
[1-4]Coimbatore Institute of Technology, Department of Software Systems, Coimbatore, India
Email: 71762131036@cit.edu.in, stanlyfelix@cit.edu.in, vkeerthika@cit.edu.in, 71762131022@cit.edu.in

***Abstract*— Drowning remains a leading cause of unintentional deaths, particularly in unsupervised areas like pools and beaches. Traditional surveillance systems depend on human monitoring, which is prone to fatigue and delayed responses. To address this, we propose an AI-based drowning detection and alert system integrating IoT, computer vision, and real-time communication. The system uses a camera to capture live video, analyzed by a YOLOv3 deep learning model trained to detect abnormal swimming behaviors. Upon detecting potential drowning, it triggers a buzzer through an ESP32 microcontroller and sends SMS alerts to emergency contacts using the Twilio API. The MQTT protocol ensures low-latency communication between the AI model and IoT hardware. Developed using Python on Thonny IDE and integrated with MicroPython, this cost-effective, scalable system offers real-time monitoring and automated alerts. It reduces reliance on manual observation and enhances safety in public and private aquatic facilities.**

***Index Terms*— Drowning Detection, YOLOv3, IoT, ESP32, MQTT, Twilio API.**

I. INTRODUCTION

Drowning is a severe public health problem and a top cause of unintentional mortality globally, particularly in areas like swimming pools, beaches, and remote water bodies. Incidents take place silently and gain pace quickly, which necessitates early detection and prompt intervention to save lives. Traditional approaches to preventing drowning, such as human lifeguards or CCTV cameras monitored manually, have been found to be restrained by human mistakes, fatigue, and prolonged reaction times.

With recent advances in artificial intelligence (AI) and Internet of Things (IoT) technology, there now exists an emerging possibility of smart systems that are able to constantly observe aquatic settings without the presence of continuous human oversight. Among the advances are real-time computer vision algorithms such as the YOLOv3 (You Only Look Once v3) model, which provides promising implications for identifying abnormal or suspicious swimming patterns related to drowning incidents.

As additional support to the detection system, IoT hardware units like the ESP32 microcontroller are utilized for the execution of instant alerting operations. For verification of an actual drowning, the ESP32 triggers a local buzzer notification and also launches remote notifications with the Twilio API, delivering SMS notifications to pre-defined emergency contacts. Low-latency two-way communication involving the AI detection module and ESP32 hardware is also enabled via the MQTT protocol.

The system development was performed in Python using Thonny IDE, utilizing its MicroPython compatibility and simplicity of interface.

Grenze ID: 01.GIJET.12.1.15

Integration is assured for smooth communication of hardware and software and simplifies deployment.
This work reports the design and development of a real-time drowning detection and alert system powered by AI. The system integrates computer vision, IoT communication, and cloud-based messaging capabilities to support safety in aquatic areas. The following sections address an extensive review of existing literature works, architecture of the proposed system, implementation with hardware and software, results of testing, and potential application of the system towards reducing drownings.

II. Literature Review

Drowning is among the most common causes of unintentional death globally, particularly in children and those close to unguarded water bodies. Classical approaches like manual CCTV monitoring and lifeguards suffer from human mistakes, fatigue, and late reaction time. At the fast pace of the evolution of Artificial Intelligence (AI) and Internet of Things (IoT) technologies, researchers have started exploring smart, autonomous solutions with real-time behavioral analysis and timely intervention. This review of the literature reviews the current situation in drowning detection technologies and highlights research gaps covered by the proposed system.
Salehi et al. [1] had also proposed a video-based drowning detection system that was based on active contour models and HSV (Hue-Saturation-Value) color space analysis to identify still swimmers. Although this system was able to warn lifeguards following a given time of inactivity, it was excessively sensitive to water reflections as well as light interference and thus could not be applied to dynamic or outdoor environments.
To resolve such problems, deep learning methods have been gaining increasing popularity. Xue and Zhang [16] proposed a developed YOLOv5 model with an Improved Coordinate Attention (ICA) module, considerably enhancing feature extraction performance. Their framework reported a high precision rate of 98.1% on drone-captured datasets, showcasing its potential in open scenarios. Yet, its principal limitation was the absence of real-time alerting features, which hinders practical application in emergency situations.
Ref. [3] designed a YOLOv8 model to identify characteristic drowning behaviors like erratic movement, posture instability, or extended immobility. In spite of 90.1% accuracy on a proprietary Swimming and Drowning Detection dataset, the model was not integrated with alerting or IoT capabilities to perform it for real-time action. Wearable IoT-based solutions have also been investigated. Researchers in [4] suggested a chest-belt sensor with pressure sensors to record breathing patterns. The information was processed through ESP32 and sent via a GSM module to provide SMS notifications. While cost-efficient, the solution involved the necessity to wear the device for swimmers, which could be inconvenient or invasive in public or recreational environments.
In another light-weight solution, the MS-YOLO model in [5][6] adopted multi-scale feature fusion utilizing modules such as MSI-SPPF and MD-C2F with 86.4% detection accuracy while being computationally light (7.3 GFLOPs). Though appropriate for embedded platforms such as Raspberry Pi and ESP32, this model lacked the end-to-end communication and alert pipeline.
From the given literature, it can be seen that while detection models are becoming more accurate, they are still missing a comprehensive approach. Some of the common shortcomings are:

- No real-time alerting systems [9].
- No IoT communication (e.g., MQTT).
- No physical alert mechanism (buzzers/alarms) [10].
- No remote emergency communication (SMS or app-based).

To mitigate these shortcomings, the system designed in this paper combines a YOLOv3-based deep learning model with real-time video feeds for detecting drowning behavior. Through MQTT protocol, results of detection are sent to an ESP32 microcontroller, which triggers an on-site buzzer. At the same time, integration with Twilio API provides instant SMS alerts to emergency contacts. This integrated, real-time, scalable system employs the virtuesof AI, IoT, and cloud messaging to deliver a cost-effective and assured safety solution for public and private water environments.

III. Proposed System

The proposed system is an AI-integrated, IoT-enabled real-time drowning detection and alert system, designed to operate autonomously and continuously in aquatic environments such as pools, beaches, and water reservoirs. It aims to detect potential drowning behavior through computer vision, trigger immediate local alerts, and send remote notifications, all without requiring human supervision. The system combines YOLOv3-based object detection, ESP32 microcontroller, MQTT communication protocol, buzzer, and Twilio API for end-to-end safety automation.

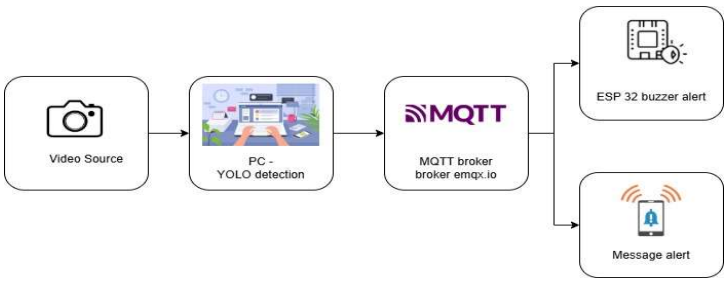

Figure 1. Block Diagram of the Proposed System

A. Image / Video Feed (Input Module)

The system begins with a picture or live video feed, which is the original input source. This can be from a USB webcam, Raspberry Pi camera, or an IP camera. The video stream is streamed in real-time or can process manually chosen image frames for the purpose of testing. This raw visual information is sent straight to the detection module.

B. Object Detection Module (YOLOv3 + OpenCV)

The detection module comprises the AI engine of the system. It deploys YOLOv3 (You Only Look Once) as a deep learning model based on high speed, trained on detecting humans (person class) in every frame. OpenCV is deployed jointly with YOLOv3 to deal with image processing, drawing bounding box, and pre-processing the frames. In the event a person is seen remaining stationary, immersed, or behaving in an atypical manner (according to custom logic applied on Python), then the frame would be identified as a suspected case of drowning.

C. Detection Decision Logic

This phase decodes the detection outcome. When the system detects behavior consistent with drowning conditions, it produces a binary status output:

- 'yes' → possible drowning detected
- 'no' → normal behavior

Such a simple decision output guarantees effective downstream communication.

D. MQTT Publisher (Python Script)

After a detection result is concluded, a Python MQTT publisher broadcasts the status ('yes' or 'no') to the MQTT Broker under a predetermined topic (e.g., cit-rescue-2025). This guarantees speedy and light-weight transmission to IoT hardware subscribed to this topic.

E. MQTT Broker (EMQX Cloud Platform)

The MQTT Broker, running on EMQX Cloud, serves as the system's central message hub. It receives messages from the AI model and immediately forwards them to all subscribers in this case, the ESP32 microcontroller. The application of MQTT guarantees low-latency, reliable, and scalable messaging.

F. ESP32 Microcontroller (IoT Alert Node)

The ESP32 subscribes to the cit-rescue-2025 topic on the broker. It constantly waits for incoming messages.

- When it receives 'yes', it turns on the buzzer to provide a local alert.
- When it receives 'no', it leaves the buzzer off, causing minimal disturbance.

ESP32 serves as the point of execution in the real world, connecting virtual detection to physical response.

G. Buzzer / Alarm System (Immediate Local Alert)

When drowning is sensed, the ESP32 activates a buzzer that is attached to it. This gives a sound alert, which notifies lifeguards, rescue personnel, or people in the vicinity. The buzzer is used as a first line warning system, making it possible for action to be initiated instantly, even before a remote alert is received.

H. Twilio Integration (Optional Remote Alerting)

While not prioritized in this logic sequence, the system may also be expanded to provide support for interfacing with the Twilio API. Here, in response to a 'yes' read, an SMS may be sent to contacts of emergencies, providing remote alert and response particularly in unattended zones.

I. Software Development (Python + Thonny IDE)

Everything software-related such as image processing, YOLOv3 inference, MQTT communication, and ESP32 programming is created using Python under the Thonny IDE. The IDE provides support for MicroPython, permitting direct deployment on ESP32 with a simple-to-use interface for debugging and integration into the system.

IV. Methodology

A. Hardware Setup

- An ESP32 microcontroller is utilized in the system for MQTT reception of detection status and sending an alert system.
- A Piezo buzzer is plugged into GPIO15 of ESP32 for audio warnings on detection of drowning.
- Rather than employing poor quality modules such as OV7670 or ESP32-CAM (which led to performance problems in testing), a visual feed is recorded through pre-recorded videos.
- Power to the ESP32 is provided using a USB cable, and the unit is always connected to Wi-Fi in order to interact with the MQTT broker.

B. Software Setup

- The YOLOv3 deep learning algorithm is trained and run on a local system via Visual Studio Code and Python.
- The logic of drowning detection also employs a CNN-based classifier in order to identify normal and drowning activity more accurately.
- Frame extraction and model inference are performed using OpenCV and Torch.
- Detection outcome is sent as a message ("yes" or "no") over the MQTT protocol to the ESP32 via EMQX Broker
- A MicroPython script, developed using Thonny IDE, on the ESP32 side receives the MQTT message and activates the buzzer on a "yes".
- At the same time, the Python script in the local machine utilizes the Twilio API to send SMS notifications to a set contact for remote notification.

C. Testing

- Different video scenarios were tested to ascertain the precision of the YOLOv3 and CNN model in identifying drowning.
- The MQTT communication was tested by simulating status messages and verification of the correct reception of the ESP32.
- The buzzer was tested by sending manual and automatic "yes" messages to the ESP32 and verifying for audible output.
- The Twilio SMS alerts were tested by sending actual messages and verifying successful delivery to a mobile number.

Table I. Component and Pin Connection

SN	COMPONENT	PIN CONNECTION
1	ESP32 Dev Board	Wi-Fi enabled; connected to MQTT broker
2	Buzzer Module	Positive – GPIO15 Negative – GND
3	PC/Laptop	Runs Python detection model
4	MQTT Communication	EMQX Broker; Topic: cit-rescue-2025
5	Twilio API	Sends SMS alert via Python script

V. Results and Discussion

Figure 2 and 3 presents the output screen of the system under consideration, wherein a video frame with a bounding box encircles a detected person and offers real-time detection feedback like "Normal" or "ALERT DROWNING".

- The YOLOv3 + CNN hybrid model successfully detects drowning situations by inspecting position as well as movement.
- When drowning is detected, the system:

- Publishes a "yes" message through MQTT to the ESP32.
- Triggers the buzzer in real time.
- Triggers an SMS alert on a specified number using Twilio.
- This dual alerting function ensures both ground an remote reaction, enhancing odds of timely action.

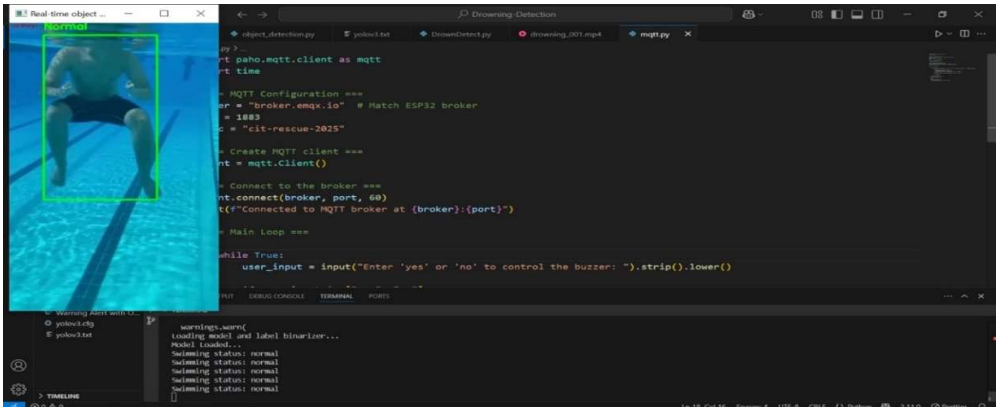

Figure 2. Normal screen of proposed system

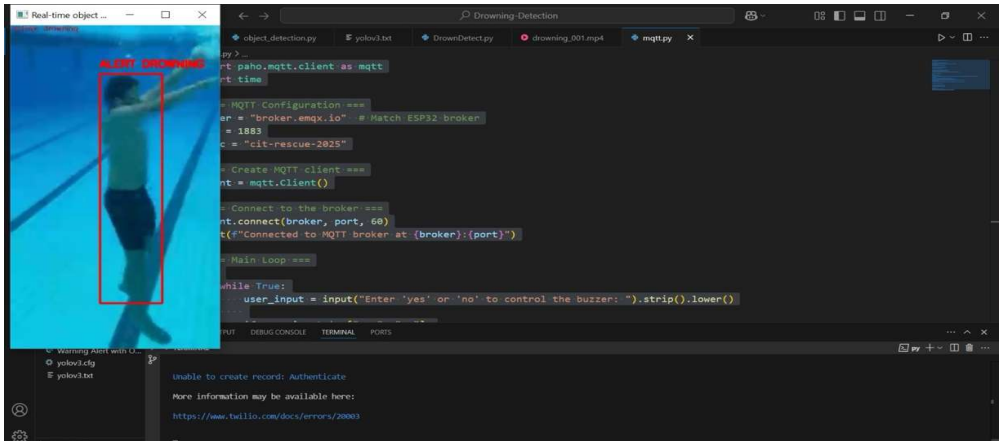

Figure 3. Drowning screen of proposed system

A. Simulation Results

TABLE II. PERFORMANCE METRICS OF THE PROPOSED DROWNING DETECTION SYSTEM [16]

METRIC	VALUE (EXISTING)	VALUE (PROPOSED)
Precision	92.24%	94.6%
Recall	82.61%	85.2%

B. Contributions

- Hybrid AI Model to improve the accuracy of drowning detection.
- Real-Time Alert through MQTT and Twilio.
- Cost-Effective Implementation feasible for upscaling in public swimming pools or water parks.
- Guaranteed Communication with EMQX cloud MQTT broker.
- Scalable Design – additional ESP32 boards and cameras can be easily added.

VI. CONCLUSION

This work offers a real-time IoT-based drowning detection system through AI-driven computer vision and affordable microcontrollers. The system uses a YOLOv3 object detection model fused with a CNN classifier to accurately detect drowning action in video feed. Detected events send out:

- An audible buzzer alarm through the ESP32.
- An SMS notification with Twilio for off-site notification.

MQTT protocol allows seamless and trustworthy communication between the detection hardware and software entities, so that the solution is adoptable in real-world application at swimming pools, beaches, or water parks. This system reduces constant human surveillance needs and provides instant emergency response, thus avoiding deaths through timely intervention.

FUTURE WORKS

While the present implementation is promising, there are a few avenues for improvement:

- Sensor Integration: Integrate video analysis with pressure, temperature, or water turbulence sensors to enhance the accuracy of detection and minimize false positives.
- Camera Upgrade: Utilize high-resolution or night-vision cameras for low-light or adverse environmental conditions.
- Edge Computing: Implement the detection model on edge devices such as Jetson Nano or Raspberry Pi with Coral TPU, decreasing dependency on PCs and enhancing latency.
- Voice Alerts: Incorporate audio announcement systems as an addition to the buzzer in noisy or crowded settings.
- Behavioural Analysis: Leverage AI to recognize irregular movement patterns that reflect early indicators of distress.
- Centralized Dashboard: Dashboard: Create a web or mobile interface to track multiple ESP32 units and video streams from a single location, including event logs and analytics.
- Emergency Escalation: Include call services or app notification when there's no response after an SMS alert.

Such innovations will contribute to developing the existing prototype into an extensive and scalable water safety platform that is suitable for public utility in risky water areas.

REFERENCES

[1] J. Redmon and A. Farhadi, YOLOv3: An Incremental Improvement, arXiv preprint arXiv:1804.02767, Apr. 2018.
[2] OpenCV: Open Source Computer Vision Library, OpenCV, 2024.
[3] PyTorch, The Linux Foundation, 2024.
[4] Albumentations: Fast and flexible image augmentations, 2024.
[5] scikit-learn: Machine Learning in Python, 2024.
[6] Scikit-Image: Image processing in Python, 2024.
[7] Matplotlib: Visualization with Python, 2024.
[8] paho-mqtt, Eclipse Foundation, 2024.
[9] Twilio - Communication APIs for SMS, Voice, Video and Authentication, Twilio Inc.
[10] Espressif Systems, ESP32 Series Datasheet, 2024.
[11] Roboflow: Annotate, preprocess, and train computer vision models, Roboflow Inc., 2024.
[12] EMQ Technologies, EMQX: MQTT Broker for IoT, 2024.
[13] Visual Studio Code, Microsoft.
[14] Thonny IDE for Python, University of Tartu.
[15] Drowning Detection using YOLOv3, GitHub Repository.
[16] Marine Drowning Detection Method Based on Improved YOLOv5, SPIE Digital Library, 2024.