MECH 658 – Introduction to Machine Learning

Project Report

Mansour Saliba

Tuesday 22 April 2024

**On the Use of Machine Learning for the Detection of Unexploded Ordnances (UXOs)**

**Table of Contents**

I. [Executive Summary: 2](#_Toc196262062)

[II. Introduction: 2](#_Toc196262063)

[III. Dataset: 2](#_Toc196262064)

[IV. Machine Learning Models: 4](#_Toc196262065)

[V. MLOps/Documentation/GitHub: 6](#_Toc196262066)

[VI. Deployment: 7](#_Toc196262067)

[VII. Quality Assurance: 7](#_Toc196262068)

[VIII. References: 8](#_Toc196262069)

# Executive Summary:

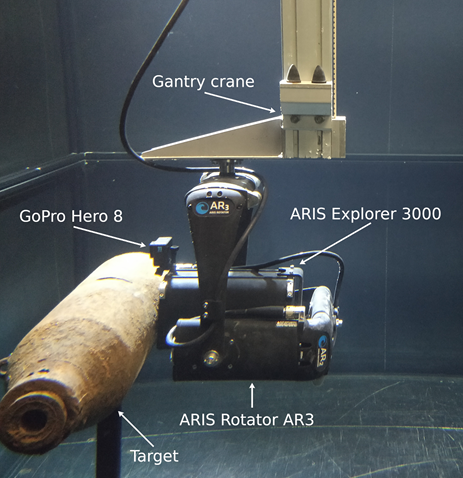
The project seeks to detect unexploded ordnances (UXOs) underwater using machine learning in order to alleviate the safety risks involved in classical detection mechanisms. The dataset consists of sonar data used for training a RESNET18 classification model and image data labeled with bounding boxes for training a YOLOv8n detection model. Detection model metrics were satisfactory while that of the classification model exhibited shortcomings. Tracking of metrics was implemented using mlflow and models were deployed using Docker, served via an API, and connected with a containerized UI. A GitHub repository was created outlining the entire project. Upscaling for real-life implementation entails overcoming numerous issues, primary of which is the need for data reflecting local conditions.

# Introduction:

Unexploded ordnances (UXOs) constitute “explosive weapons such as bombs, bullets, shells, grenades, mines, etc. that did not explode when they were employed and still pose a risk of detonation” (Craioveanu & Stamatescu, 2024). Identification of UXOs demands manual inspection by experts who decide whether an UXO is safe to transport and handle, or has to be destroyed on site (Dahn et al., 2024). This poses an obvious threat, resulting in injuries and deaths of hundreds of people each year worldwide (Cho et al., 2023). Accordingly, this project seeks to explore the use of machine learning algorithms for the detection of underwater UXOs following conflicts, such as the ongoing war on Lebanon.

# Dataset:

The dataset consists of synchronized acoustic and optical sensing of UXOs underwater, developed at the German Research Center for Artificial Intelligence (Dahn et al., 2024). Optical data was collected using a GoPro and acoustic data using an ARIS Explorer 3000 imaging sonar (Fig. 1), implemented on three targets: a rusted 100 lb general purpose bomb, a deformed phosphor bomb, and a mortar bomb (Table 1; Fig. 2). The dataset amounts to a total of 37,278 annotated images and it can be accessed on Zenodo following this link: <https://zenodo.org/records/11068046>



**Fig. 1.** Experiment setup (Dahn et al., 2024).

This dataset amounts to around 60 GB but for the purposes of this demonstrative project, a sample of ~1.7 GB was used, amounting to 2,573 matched instances of sonar, image, and label data after removing mismatched data. Data exploration elucidated the fact that bounding boxes (i.e. labels) match only the optical data, making sonar data unfit for training of a detection model. Hence, to still benefit from both types of data, a second dataset was also incorporated, comprising of sonars collected of common everyday objects underwater (Valdenegro, 2025). Accordingly, positive and negative sonar data will be used for a classification model while the image data with their .JSON labels will be used for a detection model.

**Table 1.** Dimensions and types of recorded UXOs underwater (Dahn et al., 2024).

A close-up of a bomb

AI-generated content may be incorrect.

All images are of scale 1920x1080 while sonars were made uniform at 1636x3025. All labels were made uniform as ‘Unexploded Ordnance’ since type of UXO is not within project scope. Additional data preprocessing steps include the need to rescale bounding box labels by x3 (Fig. 3). Data used for classification was slightly imbalanced at ~16% so positive data was slightly under sampled, and weight on the negative class was increased and its data slightly augmented. All sonar data were then rescaled to (224, 414) and pixel values normalized, reducing computational overhead. As for the data used for detection, it was rescaled to (640, 640), bounding box coordinated were normalized and converted from .JSON to a type compatible with detection model. Both datasets followed an 80% training/10% validation/10% testing splitting approach and training set was augmented for increased robustness of trained models.

A log on a pole

AI-generated content may be incorrect.

**Fig. 2.** Matched sonar and camera frames of three UXOs (Dahn et al., 2024).

A collage of images of a person in a room

AI-generated content may be incorrect.

**Fig. 3.** Scaling of label coordinates.

# Machine Learning Models:

For classification of sonar data, training was initiated with two baseline models: Random Forest Classifier and a simple Convolutional Neural Network (CNN) comprised of 2 convolutional layers and 2 dense layers ending with sigmoid as an activation function for binary classification. The Random Forest Classifier immediately memorized the data with 100% accuracy while the simple CNN was not an adequate model with accuracy reaching ~55%. Accordingly, the final model was trained on RESNET18 which is comprised of 17 convolutional layers and 1 dense layer. The employed loss function is Binary Cross Entropy, the used batch size is 32, and 50 epochs were set as maximum. Optimization techniques included Adam optimizer, early stopping, dropout at 0.3 (removes 0.3 of neurons during training, increasing complexity and reducing overfitting risk), L2 regularization (reduces overfitting risk by penalizing large weights), learning rate scheduler (every 3 epochs divide α by half if validation loss not changed).As for the detection model, YOLOv8n (nano) was chosen for its lightweight and fast yet accurate detection. Batch size was set to 16 and 50 maximum epochs were allowed. Similar optimization techniques were employed such as early stopping and learning rate scheduler.

Evaluation of the YOLOv8n yielded satisfactory results. Loss functions for both training and validation sets decreased steadily with increasing epochs while metrics such as precision and recall increased (Fig. 4). Risk of overfitting is certainly not negligible yet it can’t be ascertained as testing set did not lead to maximized metrics; the possibility of collecting field data could have improved the model further.

A graph of a number of data

AI-generated content may be incorrect.

**Fig. 4.** Evaluation metrics of training and validation data for the detection model.

On the other hand, evaluation for the classification model exhibited concerning outcomes. Upon the third epoch, both training and validation accuracy reached a 100%, clearly indicating a high probability of overfitting. Even worse, in the last epoch, validation accuracy remained at 100% while training accuracy decreased slightly. This constitutes a major issue, and in order to try and fix it, steps undertaken included checking data leakage, tuning hyperparameters, increasing and decreasing data augmentation, and training a more complex (RESNET50) and less complex (MobileNetV2) models, yet evaluation results still reflected the same discrepancy (Fig. 5). While uncommon, this behavior can occur and some of its explanations include the fact that training data

A graph with a line going up

AI-generated content may be incorrect.A graph with a line going up

AI-generated content may be incorrect.

**Fig. 5.** Training accuracy of classification model.

**Fig. 6.** Validation accuracy of classification model.

is always augmented while validation/testing data is not (easier to classify), use of dropout (deliberately hurting training performance), the fact that validation metrics are computed on supposedly better weights than training metrics, but primarily it can be assumed to be due to the small dataset. With additional and more complex data instances that reflect real-life conditions, such errors are not be expected to occur.

# MLOps/Documentation/GitHub:

The entire project life cycle is documented through multiple means. Primarily, a [GitHub repository](https://github.com/MansourSaliba/UXO_detection_model) was established from the beginning with a clear directory for each aspect of the project, a straightforward README page, and multiple commits registering the multiple steps from data extraction up to model deployment. Data versioning and control (DVC) was implemented during data processing so that users wishing to follow the same data cleaning and processing steps can do so easily. Also, this report serves as a different aspect of documentation detailing the steps taken and their results in addition to the prepared presentation and UI demo. Proper employment of MLOps entail multiple steps, from feature storing to experiment tracking and monitoring during production (Fig. 7). For the purposes of this project, MLOps processes primarily entailed setting up experiment tracking using mlflow in order to track the performance of the different models trained (Fig. 8). The trained models were also containerized and deployed with a UI.

A diagram of a diagram

AI-generated content may be incorrect.

**Fig. 7.** MLOps primary pillars (Rouis, 2023).

A graph of a function

AI-generated content may be incorrect.

**Fig. 8.** Example of tracking with mlflow.

# Deployment:

Flask API and Docker Desktop were utilized for the containerization and serving of the models and the UI. A docker image was built for each of the classification and detection models after the APIs were set, then docker containers were run and tested. The UI was developed using Streamlit and it was also served via Flask API and containerized using ‘docker --compose’. The UI follows a simple user interface that informs the users about UXOs, prompts them to upload either a sonar or an image, and then it will classify the former or return the latter with a bounding box indicating UXO location. If a UXO is present, the UI alerts the user to maintain a distance and contact relevant authorities such as the Lebanese Mine Action Center (UI Demo can be found on [Google Drive](https://drive.google.com/drive/folders/1-OtviFB_2QhGz4JM7WutXUYMTrTTQ-DH?usp=drive_link)).

# Quality Assurance:

While the model attempts to tackle a crucial societal issue, it only serves as an initial starting point and scaling it requires addressing multiple limitations. First, a real-life model should be trained on all ~60 GB dataset and not a sample, and it should incorporate field-collected data that reflects local conditions of water bodies especially as the used dataset has optimal lighting and water turbidity conditions not necessarily reflective of reality. Second, in such an application, there needs to be particular care for false negatives as they can lead to disastrous consequences. Third, a future model can go beyond detection and incorporate classification of UXO by type if adequate data is available

# References:

Cho, S., Ma, J., & Yakimenko, O. A. (2023). Aerial multi-spectral AI-based detection system for unexploded ordnance. Defence Technology, 27, 24-37.

Craioveanu, M. G., & Stamatescu, G. (2024, June). Detection and Identification of Unexploded Ordnance Using a Two-Step Deep Learning Methodology. In 2024 32nd Mediterranean Conference on Control and Automation (MED) (pp. 257-262). IEEE.

Dahn, N., Firvida, M. B., Sharma, P., Christensen, L., Geisle, O., Mohrmann, J., ... & Kirchner, F. (2024, September). An Acoustic and Optical Dataset for the Perception of Underwater Unexploded Ordnance (UXO). In OCEANS 2024-Halifax (pp. 1-6). IEEE.

Rouis, Y. (2023). A Guide to MLOps with Airflow and MLflow. Medium. Retrieved from <https://medium.com/thefork/a-guide-to-mlops-with-airflow-and-mlflow-e19a82901f88>

Valdenegro, M. (2025). Forward-Looking Sonar Marine Debris Datasets. Github Repository. Retrieved from <https://github.com/mvaldenegro/marine-debris-fls-datasets>