# A Dynamic Machine Learning Model for Accelerated Oil Spill Remediation

Sathvik Chemudupati
*Department of Computer Science*
*University of Texas at Austin*
United States of America
sathvikc@utexas.edu

***Abstract*—134 million gallons of oil were spilled into the Gulf of Mexico after the explosion of an offshore oil rig in 2010. Known as the Deepwater Horizon spill, this event crippled marine environments spanning thousands of miles and killed countless sea creatures already deemed at risk of extinction. Over 10 years and billions of dollars later, efforts to clean up this spill continue. Rapid mitigation is necessary to prevent future incidents from spiraling out of control. After an oil spill, various organizations must decide how to remediate it. To do so, there are close to a dozen methods employed today. Each approach has its pros and cons and must be carefully selected based on spill conditions. Some techniques (such as in-situ burning of the oil slick off the water) are highly effective but have environmentally degrading effects. Choosing a suboptimal remediation tactic can lead to billions of wasted dollars, and more importantly, leftover oil that continues to harm the environment. During this study, an artificial intelligence (AI) based system using a convolutional neural network (CNN) has been developed to prescribe the most effective oil spill countermeasure. Findings were used to develop a mobile application to further expedite oil spill cleanup and recovery in real time. After being tested at various configurations, the machine learning model achieved a maximum average accuracy of 93.1% after 16.19 seconds of training time with 10 epochs and a batch-size of 16. This work significantly enhances our ability to quickly remediate oil spills, protecting the environment from this disastrous calamity.**

***Index Terms*—Oil Spill Remediation (OSR), Artificial Intelligence (AI), Machine Learning (ML), Deep Learning, Convolutional Neural Network (CNN), Generative Adversarial Network (GAN), Oil booms, In-situ burning, dispersants, bioremediation, sorbents.**

## I. Introduction

During an oil spill on US waters, the United States Coast Guard, Environmental Protection Agency (EPA), National Oceanic and Atmospheric Administration (NOAA), and Wildlife Services (among various other organizations) are often involved in the cleanup effort. These organizations must decide how to best remediate the spill. Today, various oil spill countermeasures ranging from skimmers to heat pressure washing are utilized for this task. The five common oil spill countermeasures of **oil booms**, **in-situ burning**, **dispersants**, **bioremediation**, and **sorbents** were considered during this study (Table I). These specific remediation techniques were selected for the study as they represent a diverse range of oil spill responses that vary in approach, materials, cost, effects, and pros/cons. For example, the implementation of oil booms, while cost-effective, is only effective over a relatively small spill radius. Meanwhile, in-situ burning can clear larger spills but is only functional over a specific thickness of oil slick, not to mention highly environmentally degrading. Dispersants and bioremediation are not as polluting to the atmosphere but risk damaging ocean ecosystems such as coral reefs near the spill site. These latter approaches fail to recover much of the oil that is spilt—sorbents address this issue but are only operable over small spills like their oil boom counterparts. In addition, the success of these oil spill countermeasures relies heavily on the surrounding conditions at the oil spill site. For example, in-situ burning and sorbents cannot be feasible when large concentrations of debris are present at the spill location. Bioremediation and oil booms would not be preferred during rough sea conditions that contribute to a faster rate of oil spill spread. My objective was to create a convolutional neural network (CNN) to recommend an oil spill countermeasure given the ambient conditions at the spill. Below are the ambient conditions I trained my machine learning model to recognize. Considering these factors, the following ambient conditions at the spill site were taken into account during this study: **wind speed**, **wave height**, **wave swell**, **presence of debris**, **amount of oil spilt**, **slick thickness**, **range of oil spill** (Table II). Most work in this domain targets oil spill prevention and detection. This study tackles the issue from the other side by presenting a novel oil spill response and remediation algorithm via a convolutional neural network (CNN) model developed by the researcher in case prevention fails.

TABLE I
SUMMARY OF FIVE COMMON OIL SPILL COUNTERMEASURE TECHNIQUES THAT WERE CONSIDERED DURING THIS STUDY.

| Countermeasure | Description |
|---|---|
| Oil booms | Floating barrier around spill zone |
| In-situ burning | Burning slick directly off surface |
| Dispersants | Chemicals that expedite oil decay |
| Bioremediation | Oil break-down with microorganisms |
| Sorbents | Collecting spill with absorbant material |

## II. Objectives

During this research, a convolutional neural network (CNN) was developed and trained to evaluate parameters describing the ambient conditions of the oil spill (Table II) and determine

TABLE II
AMBIENT OIL SPILL CONDITIONS EVALUATED TO DETERMINE OPTIMAL OIL SPILL COUNTERMEASURE.

| Ambient Condition | Scale |
|---|---|
| Wind Speed | Beaufort Scale (0-12) |
| Wave Height | Douglass Scale (0-9) |
| Wave Swell | Douglass Scale (0-9) |
| Presence of Debris | True/False (0 or 1) |
| Amount of Oil Spilt | Value in Gallons (gals) |
| Slick Thickness | Value in Millimeters (mm) |
| Range of Oil Spill | Value in Kilometers (km) |

an optimal oil spill countermeasure (Table I) based on those factors. Through experimentation, the model's batch size and epoch count parameters are adjusted to determine the optimal configuration that trains the fastest and yields the highest percentage of accurate predictions in a series of test cases not used during training. After all, this project seeks to develop an oil spill response paradigm based on deep learning that is both rapid and effective. These findings are then integrated into a mobile application that can be used at the spill site to provide rapid and accurate countermeasure suggestions, all while allowing the model to continue learning based on user interaction and feedback.

## III. MATERIALS AND METHODS

Extensive research was conducted regarding when each of the 5 oil spill countermeasures examined in the study (Table I) are intended to be used and their relationship to the ambient conditions considered (Table II) including historical outcomes.

The following procedure was followed for the research and product development:

1) Synthesize a series of oil spills (amount spilt, slick thickness, range) and associated ambient conditions (wind speed, wave height, wave swell, debris) along with the most appropriate remediation technique based on my research.
2) Program a Generative Adversarial Network (GAN) to simulate thousands of such oil spills.
3) Construct a convolutional neural network (CNN) with input parameters as the oil spill conditions and output as a probability matrix of various countermeasures and the confidence level associated with each.
4) Use the CNN model to prescribe optimal oil spill remediation technique on test data not used during the training phase and measure the accuracy.
5) Vary the batch size and epoch numbers and repeat steps 3-4 to determine the configuration that maximizes accuracy while minimizing training time.
6) Create a mobile application that prescribes optimal oil spill countermeasure based on conditions at the spill site to further expedite cleanup and recovery.

TABLE III
END BOUNDS OF UNIFORM DISTRIBUTION USED FOR SYNTHESIS OF SPILLS CORRESPONDING TO OIL BOOM COUNTERMEASURE. REFER TO TABLE 2 FOR A MORE SPECIFIC DESCRIPTION OF THE PARAMETER FORMAT.

| Ambient Condition | Minimum | Maximum |
|---|---|---|
| Wind Speed | 2 | 4 |
| Wave Height | 2 | 4 |
| Wave Swell | 2 | 4 |
| Presence of Debris | 0 | 1 |
| Amount of Oil Spilt | 100 | 10000 |
| Slick Thickness | 0.001 | 3 |
| Range of Oil Spill | 0 | 0.2 |

### A. Oil Spill Parameter Representation

During this study, an oil spill is represented by a series of numerical values corresponding to the ambient conditions at the spill site (Table II) and the specific oil spill countermeasure used for clean-up (Table I). In this model, the features of the input to the CNN are the ambient conditions of the spill and the label (output or prediction of model) is one of five countermeasures to best remediate the corresponding spill. Thus, each oil spill is internally represented as seven features (the number of ambient conditions considered in this study) and one of five labels (the countermeasure assigned for remediation).

In regards to oil spill synthesis in the first stage of the procedure, a Python script was programmed to randomly simulate $500$ distinct oil spills in this manner. Oil spill features were synthesized using a random uniform distribution between a pre-determined range and the label (countermeasure for the spill) for that spill was assigned based on research. The initial synthesized oil spills in the first step of the procedure were procured such that they were evenly distribution by countermeasure—that is, each oil spill remediation tactic (label) appeared an equal number of times ($100$) in the result of the initial synthesis step. Refer to example end bounds of the uniform distributions used to synthesize spills for the oil boom countermeasure (Table III).

As a note, the presence of debris feature did not follow the uniform distribution; it was a fixed integer value of $0$ or $1$ for each spill depending on whether its corresponding countermeasure was optimized for this ambient condition. Also, these end bounds were finalized based on research done on the countermeasures themselves and the conditions to which they were designed for. However, future study can formalize and further validate these numerical parameters to establish a more rigorous baseline—we stopped short of this in the paper. While these parameters were partially inspired by historical precedent, we could not solely rely on them because specific data regarding the ambient conditions of past oil spills and the countermeasure implemented is rather sparse. Moreover, some of the remediation techniques considered such as bioremediation are relatively modern and would not have been used historically at all. Therefore, after research into the topic, these parameters were used as the baseline for this study.


Fig. 1. CNN architecture for oil spill remediation model.

### B. Generative Adversarial Network (GAN) Description

After an initial synthesis of a small number of evenly distribution oil spills, a generative adversarial network (GAN) was trained to synthesize thousands of oil spills based on the underlying trends observed in the initially simulated set. Synthetic data was necessary as more training samples were needed to train the CNN classifier, and the use of a GAN is widely recognized as a viable means of obtaining this data. In this study, the Conditional Tabular GAN (CTGAN) proposed by Xu et al. (2019) was utilized to generate oil spill samples for CNN training. This CTGAN is designed to generate tabular data by training on a baseline set provided to the model. The $500$ synthetic oil spills from the first step of the procedure were tabularized and then passed through the CTGAN. The CTGAN was trained with GPUs in a Google Colab environment and configured for $30,000$ epochs to synthesize thousands of oil spills along with their countermeasure.

### C. Pre-Processing Algorithm

The GAN-generated data was ultimately used for machine learning model training and evaluation. Before that, it must be pre-processed so that meaningful trends are spotted by the algorithm rather than insignificant variances.

The following pre-processing steps were taken:

- The data for each oil spill was verified to be the same length (countermeasure label and 7 surrounding condition features).
- Data was normalized to the range from $0$ to $1$:
- All negative values were multiplied by $-1$ to ensure positive inputs. By virtue of how CTGAN generates values, some data points were inevitably negative.
- All values were scaled down to fit the range by being divided by the maximum possible value for that category of features.

### D. Convolutional Neural Network (CNN) Development

A Convolutional Neural Network (CNN) was developed using the TensorFlow module with the Python programming language. The Neural Network consists of the following eight layers: Layer 1 is a 1D Convolutional in which adjoining weights are utilized to compute the output. Layer 2 reduces the size of the Layer 1 output by considering the most significant weight in a "pool" of any two given weights. Layer 3 reduces data representation into 1 dimension. Layers 4-7 use the Rectified Linear Unit activation function to linearize the data structure. Layer 8 uses the SoftMax activation function to generate a probability matrix of size $5$. The position in the matrix with the largest value (or confidence) corresponds with the final oil spill countermeasure recommendation. Refer to the architecture diagram (Fig. 1) for a brief summary of model layers.

TABLE IV
NUMBER OF TEST CASES FOR EACH OIL SPILL COUNTERMEASURE.

| Oil Spill Classification | Number of Test Cases |
|---|---|
| Oil Booms | 315 |
| In-situ Burning | 289 |
| Dispersants | 328 |
| Bioremediation | 325 |
| Sorbents | 343 |

## IV. RESULTS

The model accuracy was evaluated on a test set that was not used during the training phase. The dataset distribution of the test set used for reporting accuracy, confusion matrices, recall, precision, and F1 scores is displayed in Table IV.

### A. Class-wise Prediction Analysis

Confusion matrices were generated for various epoch and batch size configurations. Furthermore, global precision, recall, and F1 scores for each configuration were analyzed. This provided insight about how these model parameters affected the predictions of specific classes. These metrics for a configuration at 4 epochs illustrates an *overfitting* trend for the model.

At 4 epochs and 16 batches, global scores for precision, recall, and F1 are all roughly $0.93$. The confusion matrix for this configuration illustrates a high degree of accuracy as well as consistency for each type of oil spill being diagnosed. A majority of test cases are predicted correctly.

However, at 64 batches, the model seems to overfit. The number of correctly predicted cases declines significantly for the dispersants and sorbents countermeasures. In the case of the latter, many more sorbent cases are misdiagnosed as bioremediation scenarios. At this configuration, global precision, recall, and F1 are $0.849, 0.831$, and $0.826$ respectively, incurring high losses from 16 batches (Fig. 2). In fact, the

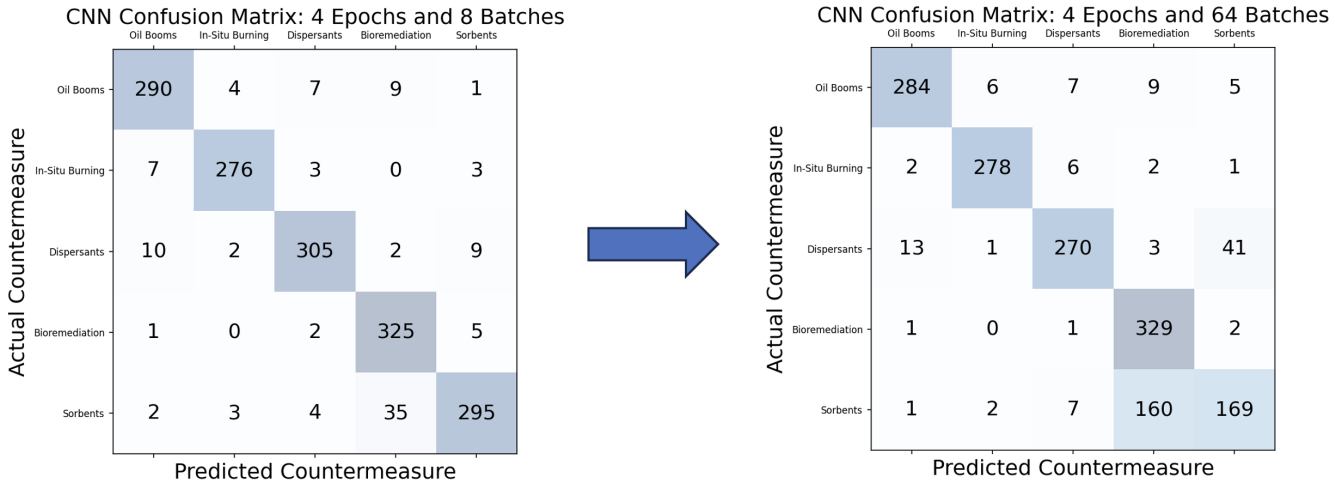


Fig. 2. Comparison of class-wise performance using a confusion matrix between 8 and 16 batches for 4 epoch configuration.

values for these metrics steadily decline for the batch counts tested in between.

The 8 epoch configurations demonstrate a more parabolic trend in regards to overfitting on batch size. At 8 batches, global precision is $0.917$, global recall is $0.908$, and global F1 is $0.909$. The confusion matrix indicates that predictions are fairly accurate and consistent between classes, albeit an abnormally high number $(65)$ of misdiagnosed bioremediation samples into the sorbent category. The model's performance in fact improves at a higher batch size of 16, with approximately $2-3\%$ gains in all metrics respectively and much fewer misdiagnosed bioremediation samples. However, interestingly, in spite of the overall gain, more sorbent samples were misdiagnosed as bioremediation samples (Fig. 3). Interestingly, this configuration achieved the highest global metric values, and precision, recall, and F1 all declined for subsequent batch sizes tested.

It was determined that my CNN model's performance is most optimized at 8 epochs and 16 batches. This configuration has high consistency and accuracy in class-wise predictions as revealed by the confusion matrix. This is corroborated by high values for the other metrics analyzed. Moreover, model training occurs in less than 15 seconds (Fig. 4), which is suitable for the dynamic model prepared in next sections.

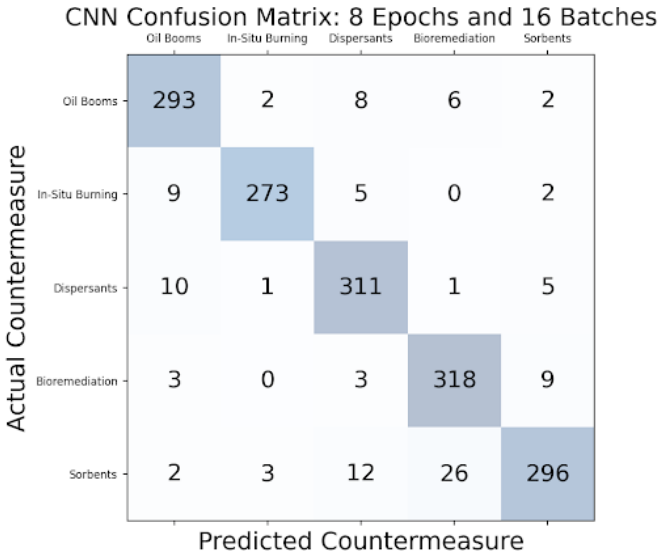


Fig. 3. Confusion matrix for configuration that yielded optimal recall, precision, and F1 performances

### B. Overview

My objective was not only to develop and present a machine learning model but also to determine the optimal configuration to maximize its accuracy and reduce its training time. After all, I seek accurate and rapid oil spill response. In my case, I searched for the perfect batch size and epoch count configuration of my machine learning model to find the combination that maximized the number of correct oil spill countermeasure recommendations and did so most quickly on a test dataset not used during the training phase of the model. The data generated by the GAN was divided into a training set and a testing set. The model trained solely on the training set and was only exposed to the testing set during performance evaluation.

Figures 4 and 5 illustrate accuracy and training time for various configurations of my machine learning model. They show the training time and accuracy at various batch size and epochs that my machine learning model parameters were set to. My machine learning model was deemed more accurate when it correctly provided an oil spill countermeasure for a higher proportion of test cases (which I designed based on careful research) not used during the training phase.

These charts illustrate accuracy trends over the duration of

***Accuracy and Training Times for Various Configurations of My Oil Spill Remediation Machine Learning Model***

| | | Number of Epochs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 2 | | 4 | | 6 | | 8 | | 10 | |
| | | Accuracy | Time (s) | Accuracy | Time (s) | Accuracy | Time (s) | Accuracy | Time (s) | Accuracy | Time (s) |
| Batch Size | 8 | 0.909 | 6.015 | 0.921 | 11.096 | 0.931 | 17.412 | 0.927 | 22.709 | 0.902 | 29.772 |
| | | 0.750 | 6.469 | 0.917 | 13.084 | 0.923 | 18.160 | 0.930 | 23.846 | 0.898 | 31.085 |
| | | 0.886 | 6.105 | 0.915 | 10.609 | 0.925 | 16.664 | 0.919 | 21.178 | 0.930 | 28.212 |
| | Average | 0.848 | 6.196 | 0.918 | 11.596 | 0.926 | 17.412 | 0.925 | 22.578 | 0.910 | 29.690 |
| | 16 | 0.916 | 5.207 | 0.918 | 7.104 | 0.891 | 10.119 | 0.923 | 14.635 | 0.939 | 17.390 |
| | | 0.878 | 4.018 | 0.893 | 6.503 | 0.903 | 9.630 | 0.924 | 13.272 | 0.928 | 15.745 |
| | | 0.906 | 5.298 | 0.908 | 6.771 | 0.918 | 10.414 | 0.919 | 13.092 | 0.927 | 15.437 |
| | Average | 0.900 | 4.841 | 0.906 | 6.793 | 0.904 | 10.054 | 0.922 | 13.666 | 0.931 | 16.191 |
| | 32 | 0.900 | 4.599 | 0.928 | 6.187 | 0.916 | 7.982 | 0.909 | 8.948 | 0.923 | 11.380 |
| | | 0.864 | 2.881 | 0.910 | 6.072 | 0.921 | 6.487 | 0.921 | 7.716 | 0.921 | 9.161 |
| | | 0.901 | 3.208 | 0.862 | 4.253 | 0.919 | 7.780 | 0.924 | 9.251 | 0.918 | 9.693 |
| | Average | 0.888 | 3.563 | 0.900 | 5.504 | 0.919 | 7.416 | 0.918 | 8.638 | 0.921 | 10.078 |
| | 64 | 0.895 | 3.717 | 0.905 | 4.553 | 0.913 | 5.188 | 0.925 | 5.941 | 0.926 | 6.801 |
| | | 0.861 | 2.358 | 0.901 | 3.484 | 0.906 | 4.303 | 0.919 | 5.265 | 0.917 | 6.645 |
| | | 0.843 | 3.828 | 0.910 | 3.108 | 0.899 | 4.042 | 0.931 | 5.527 | 0.926 | 6.382 |
| | Average | 0.866 | 3.301 | 0.905 | 3.715 | 0.906 | 4.511 | 0.925 | 5.578 | 0.923 | 6.609 |

Fig. 4. CNN model accuracy and training times at all batch size and epoch configurations tested.

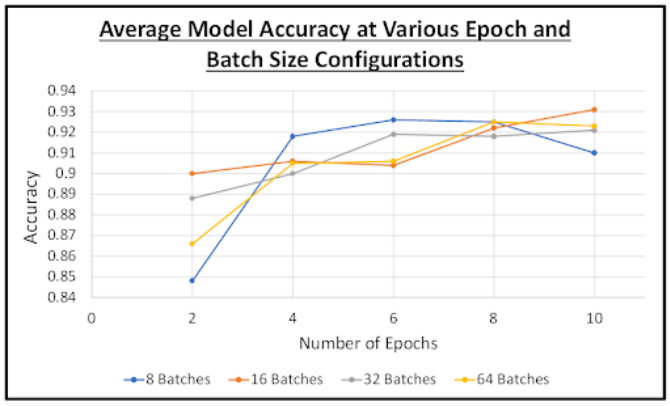


Fig. 5. Relationship between number of epochs and model accuracy for various batch size configurations.


Fig. 6. Mobile application pipeline.

three trials for different configurations of the model. Figure 5 depicts model accuracy as a function of number epochs for all the batch sizes tested. At 8 batches, model experiences significant performance gain of about 7% between 2 and 4 epochs. Performance remains consistent until 8 epochs, after which the model overfits and drops in accuracy at 10 epochs. At 16 32, and 64 batches, model performance increases with epoch count but shows signs of leveling off beyond 8 and 10 epochs (Fig. 5). This is indicative of over-fitting of the model due to too much training time.

Furthermore, training time of the model tends to increase with epoch count and decrease with batch size (Fig. 4). As epoch count increases for a given batch size, training time declines; at 10 epochs, training time is reduced by upwards of 25 seconds between 8 and 64 batches. On the other hand, training time increases as epoch count rises for a given batch size. At 64 batches, training time doubles between 2 and 10 epochs of training (Fig. 4).

Based on these results, I found that my model was most successful at about 8 epochs and 16 batches in terms of maximizing accuracy and minimizing training time. I used this finding when building my comprehensive mobile application.

## V. Prototype Description

I created a mobile application with the Android Studio Suite that can be used at the site of an oil spill to recommend the most optimal countermeasure. The mobile application has a Python backend for the machine learning computation and a Java-based interface for user interaction. Upon opening the mobile application and bypassing the start screen, the user is

prompted to input information about the ambient conditions of the oil spill.

After classifying the spill based on these factors, the mobile application uses my machine learning model (which I designed based on my experimentation) to prescribe the optimal remediation tactic out of five options. Refer to previous sections for more information about the machine learning architecture. After a countermeasure suggestion is offered, the user can choose to use the result for future remediation efforts. If selected, the conditions entered by the user and the result will be added to the train data and used to train future iterations of the model, contributing to its dynamic nature.

The application "pipeline" summary (Fig. 6) demonstrates how a user will interact with my app. Anybody can open this app at the oil spill site and know immediately how to take accurate action.

## VI. Conclusion

I programmed a convolutional neural network (CNN) to predict the most successful oil spill countermeasure given various conditions at the spill site, with a maximum accuracy of 93.1% under a given configuration. I compiled my model at various batch size and epoch configurations to determine their effect on accuracy and training time. As the number of epochs increased, accuracy increased, but training time was lower. Accuracy increased with lower batch sizes, also at the expense of slower training times. I ultimately created a mobile application that uses AI to determine the most effective oil spill remediation technique in real time. My findings helped me decide which parameters to set in the mobile application model to allow for fast and accurate predictions, allowing cleanup efforts to begin immediately and lead to a successful outcome directly on site of the spill. My mobile application is dynamic because each time a prediction is requested, users can choose to store that result to be used for future iterations of the model. Thus, the machine learning model continues to learn and adapt based on user interaction. My research and product contributes to expedited oil spill response by using machine learning, rescuing endangered ecosystems and fostering a clean future to come.

## VII. CNN Model & GAN Code

The CNN model and GAN data synthesis code developed and utilized by the researcher in this paper are at https://github.com/schem05/Machine-Learning-Model-for-OSR. Please contact the researcher for more information.

## References


[1] Agarwal, Mayur. "10 Methods for Oil Spill Cleanup at Sea." Marine Insight, Marine Insight, 30 Apr. 2021.

[2] "Deepwater Horizon Oil Spill Longterm Effects on Marine Mammals, Sea Turtles." National Ocean Service,National Oceanic and Atmospheric Administration (NOAA), 20 Apr. 2017.

[3] "Oil Spills". National Oceanic and Atmospheric Administration (NOAA).

[4] Schleifstein, Mark. "BP and Its Partners Have Spent $71 Billion over 10 Years on Deepwater Horizon Disaster."NOLA.com, NOLA, 18 Apr. 2020.

[5] Schleifstein, Mark. "BP Oil Spill Cost Fishing Industry at Least $94.7 Million in 2010." NOLA.com, NOLA, 27 June 2016.

[6] "Who Takes Care of the Problem of Oil Spills?" Office of Response and Restoration, National Oceanic and Atmospheric Administration (NOAA).