



**UNIVERSITY OF HOUSTON**

**CNST 6308**

**DATA ANALYTICS IN CONSTRUCTION  
MANAGEMENT**

Fall 2023

Professor: Dr. Lu Gao Ph. D

**SHIP IMAGE CLASSIFICATION**

Authors:

Chandra Prakash Rao Boinapally 2249602 [cboinap2@cougarnet.uh.edu](mailto:cboinap2@cougarnet.uh.edu)

Venkata Satya Reddy Kallam 2249009 [vkallam@cougarnet.uh.edu](mailto:vkallam@cougarnet.uh.edu)

Varshitha Pendyala 2244279 [vpendyal@cougarnet.uh.edu](mailto:vpendyal@cougarnet.uh.edu)

## Table of Contents

S. No.	Title	Page No.
1.	Abstract	3
2.	Introduction	3
3.	Literature Review	4
4.	Problem Definition	5
5.	Dataset Information	6
6.	Exploratory Data Analysis and Visualization	7
6.1.	Images Before Preprocessing	7
6.2.	Preprocessing Of Data	8
6.3.	Images After Preprocessing	8
6.4.	Splitting Of Data	8
6.5.	Data Augmentation Overview	8
7.	Model Building	9
7.1.	Inceptionv3	9
7.2.	Sequential	10
7.3.	Transfer Learning Integration	11
7.4.	Compile Model	12
7.5.	Fitting Model	12
7.6.	Saving Model	12
8.	Model Evaluation and Metrics	13
8.1.	Accuracy	13
8.2.	Classification Report	14
8.3.	Confusion Matrix	15
9.	Conclusion	16
	References	17

## **1. ABSTRACT:**

This project responds to the imperative need for robust ship detection across critical domains, including defense, security, marine pollution control, sea safety, and fisheries management. Commissioned by a government maritime and coastguard agency, the initiative leverages advanced computational Neural Network systems. The primary objective is to engineer a model that excels in both accuracy and efficiency for classifying diverse ship types. As a dedicated consultant for this venture, our focus is on harnessing innovative technology to identify and categorize ships accurately from photographs captured by survey boats. This pioneering solution holds the potential to significantly enhance maritime surveillance capabilities, bolster national security, and proactively address challenges related to marine environmental protection and fisheries management. The developed model stands poised as a cornerstone in advancing the agency's mission to fortify maritime safety and security through state-of-the-art neural network methodologies.

## **KEYWORDS:**

Ship Images and

Categories: 1. Cargo, 2. Military, 3. Carrier, 4. Cruise and 5. Tankers

## **2. INTRODUCTION:**

The detection of ships holds pivotal significance in addressing many challenges within the maritime sector, ranging from environmental preservation to safety considerations. A sophisticated Neural Network system is currently being implemented as part of an initiative led by the Governmental Maritime and Coastguard Agency. This initiative underscores the critical nature of accurately identifying ship types from survey boat photographs. In the role of a consultant, the primary task is to develop a model that not only ensures precision and reliability in ship categorization but also meets efficiency standards.

The foundational dataset for this endeavor comprises 6252 images. Various ship categories are denoted by numerical designations, including Cargo (1), Military (2), Carrier (3), Cruise (4), and Tankers (5). The dataset is characterized by two key variables: "image," representing the image's and "category," denoting the ship category.

Evaluation of the model's performance will be based on the weighted F1 score, emphasizing the importance of a balanced approach to achieve high recall and precision. This project not only poses a technological challenge but also signifies a practical application of state-of-the-art Neural Network to enhance operational efficiency, security, and maritime safety. Competitors in this domain have a significant opportunity to contribute to tangible solutions in both coast guard and maritime industries.

### **3. LITERATURE REVIEW:**

InceptionV3, a deep convolutional neural network architecture, has gained prominence in computer vision tasks, particularly image classification. Developed by Google, it represents a significant advancement in the field, known for its efficiency and accuracy.

#### *Architectural Innovation:*

Szegedy et al. introduced InceptionV3 with the aim of addressing challenges in computational efficiency and performance. The architecture is characterized using inception modules, incorporating parallel convolutional filters of varying sizes to capture features at different scales [1].

#### *Transfer Learning Success:*

InceptionV3 has proven effective in transfer learning scenarios. Pre-trained on large-scale datasets such as ImageNet, the model demonstrates remarkable adaptability to diverse image classification tasks with minimal fine-tuning [2].

#### *Computational Efficiency:*

The inception modules contribute to computational efficiency by reducing the number of parameters while maintaining expressive power. This efficiency is crucial for real-time applications and resource-constrained environments [3].

#### *Applications Across Domains:*

InceptionV3 finds application in various domains beyond image classification, including object detection and segmentation. Its versatility makes it a preferred choice in research and industry applications [4].

#### *Continued Evolution:*

The Inception architecture has evolved over time, with subsequent versions addressing specific challenges. InceptionV3, building on its predecessors, stands out for its balance between model complexity and performance [5].

#### **Limitations:**

InceptionV3 exhibits certain limitations. Firstly, the extensive computational demands associated with its inception modules may pose challenges for deployment on resource-constrained devices. Additionally, while successful in transfer learning, its adaptability might be constrained in domains with markedly different data distributions. The reduction in parameters for computational efficiency could potentially compromise nuanced feature representation. Furthermore, InceptionV3, like many deep architectures, may struggle with interpretability, hindering the understanding of its decision-making process. Lastly, evolving to address specific challenges, InceptionV3 may require continuous adjustments for optimal performance across diverse applications.

#### **4. PROBLEM DEFINITION:**

The problem at hand is to design and implement a sophisticated Neural Network model capable of accurately classifying ships using images. This system is critical for the maritime sector, where it will play a significant role in environmental conservation and the enhancement of maritime safety. The model will be a part of an initiative by the Governmental Maritime and Coastguard Agency to improve the precision of ship detection and categorization from survey boat photographs.

The primary objectives for the model include:

**High Accuracy:** Ensuring that the model correctly identifies the ship type from the images, which is critical for applications in maritime surveillance and environmental protection.

**Efficiency:** The model should be able to process and categorize images quickly and effectively to support real-time decision-making.

**Reliability:** The system must be robust, providing consistent results across a variety of conditions and ship types.

**Balanced Performance:** The model's success will be judged on a weighted F1 score, which combines recall (the model's ability to detect all relevant instances) and precision (the model's ability to only detect relevant instances), ensuring that the system is balanced and minimizes false positives and negatives.

The dataset consists of 6252 images that are pre-labeled with categories, each representing a type of ship. The aim is to use these images to train the Neural Network to recognize and categorize new, unseen images of ships into one of the five categories: Cargo, Military, Carrier, Cruise, or Tankers.

## 5. DATASET INFORMATION:

The ship image classification dataset comprises images representing various categories. Here are sample images from each category within the dataset.



*Fig. 1. Cargo*



*Fig. 2 Military*



*Fig. 3 Carrier*

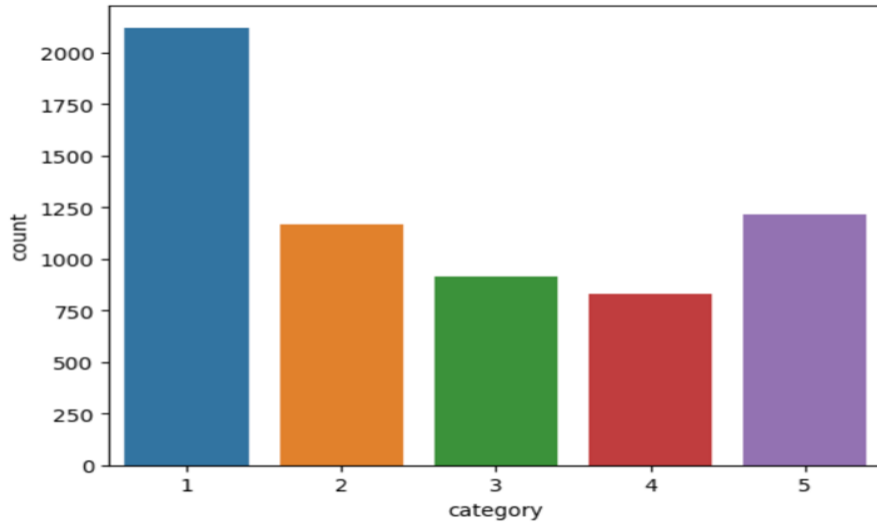


*Fig. 4 Cruise*



*Fig. 5 Tankers*

## 6. EXPLORATORY DATA ANALYSIS AND VISUALIZATION:



*Fig. 6 Count plot of all classes.*

The graph presented delineates the frequency distribution of various classes of ships along the y-axis, with categorical values labeled on the x-axis. Specifically, the number 1 corresponds to Cargo ships, 2 to Military vessels, 3 to Carriers, 4 to Cruise ships, and 5 to Tankers. The y-axis quantifies the number of occurrences for each ship category within the dataset.

### 6.1. IMAGES BEFORE PREPROCESSING:



*Fig. 7 Images Before Preprocessing*

The provided snapshot illustrates an array of images arranged in a 2x4 grid, each representing a distinct category and varying in size. These images serve as initial data prior to preprocessing and demonstrate the diversity in dimensions and types of ships within the dataset. It is evident from this collection that preprocessing steps are required to standardize the image sizes for consistency before they can be effectively utilized in the training and development of a classification model. This standardization is crucial to ensure that the model can learn from the features of each category without bias towards a particular image size.

## 6.2. PREPROCESSING OF DATA:

In the project's initial phase, a pivotal preprocessing step revolves around handling image data. The primary and foremost task is to resize all images to fixed length, considering the dataset comprises images of varying dimensions.

Every image in the dataset is loaded, transformed into the RGB format, and then resized to a consistent dimension of 128x128 pixels. Following this, the pixel values undergo normalization, being scaled to fit within the [0, 1] range. The resulting resized images are organized into a NumPy array, ready for deployment in machine learning. This step is crucial in preparing the images for subsequent utilization in machine learning models.

## 6.3. IMAGES AFTER PREPROCESSING:



*Fig. 8 Images After Preprocessing*

The image array depicted above, organized into two rows and four columns, showcases the output of the preprocessing phase, where each image has been resized to uniform dimensions. This standardization across the dataset facilitates the categorization of each image into its respective class, such as Cargo, Military, Carrier, Cruise, and Tankers. With the preprocessing step completion, ensuring consistent image sizes, the dataset is now prepared for the subsequent stages of training and model development.

## 6.4. SPLITTING OF DATA:

The purpose of splitting data is to divide our dataset into two subsets: one for training our ship detection model and the other for testing its performance. This division helps us evaluate how well our model generalizes to new, unseen data.

In simpler terms, it is like splitting a deck of cards into two parts, one for learning (training) and the other for testing how well we have learned (evaluation). This step is crucial in ensuring that our ship detection model is both effective and reliable across various situations.

## 6.5. DATA AUGMENTATION OVERVIEW:

**Purpose and Importance:** In our project, we employed data augmentation as a crucial technique to enhance the training process and improve the performance of our neural network model. The primary objectives and importance of data augmentation are as follows:

Our data augmentation strategy involved the application of several common image transformation techniques:



**Rotation:** Images were rotated by a specified angle, introducing variations in orientation. This helps the model recognize objects from different perspectives.

**Horizontal and Vertical Flipping:** Images were horizontally and vertically flipped. Flipping augments the dataset with mirror images, enhancing the model's ability to handle variations in object orientation.

**Zooming:** Random zooming was applied to simulate varying distances between the camera and the objects in the images. This helps the model learn to recognize objects at different scales.

**Width and Height Shifts:** Random shifts in the width and height dimensions were introduced, adding positional variability, aiding the model in learning to detect objects in distinct positions within the image.

## **7. MODEL BUILDING:**

### **7.1. INCEPTIONV3:**

InceptionV3 is a powerful convolutional neural network (CNN) architecture designed for image classification tasks. It was developed by Google and is known for its efficiency and accuracy in recognizing objects within images.

InceptionV3 stands out for its deep architecture, consisting of multiple layers, allowing it to capture intricate patterns and features in images. The network employs a unique "inception" module, which uses different-sized convolutional filters concurrently, enabling it to capture information at various scales.

This architecture helps prevent the loss of valuable information during image processing, making the model more robust. InceptionV3 utilizes a combination of 1x1, 3x3, and 5x5 convolutional filters in its modules, enabling it to capture both local and global features.

The 1x1 convolutions aid in reducing the dimensionality of the input, making computations more efficient. The network employs batch normalization, which helps in stabilizing and accelerating training. It also uses auxiliary classifiers during training, providing additional gradients and aiding in the convergence of the model.

It incorporates global average pooling, which helps in reducing the spatial dimensions of the model while retaining essential information. The model uses many parameters, allowing it to learn intricate patterns in data, but it is still efficient due to its unique architecture. It is pre-trained on a massive dataset, such as ImageNet, making it adept at recognizing a wide variety of objects.

Transfer learning can be effectively applied with InceptionV3, as its pre-trained weights can be fine-tuned for specific tasks. This architecture has been widely used in computer vision applications, including image classification, object detection, and feature extraction.

InceptionV3 is known for its ability to handle multi-class classification tasks, making it suitable for diverse image datasets.

The model has been implemented in popular deep learning frameworks like TensorFlow and Keras, simplifying its integration into different projects. It has proven effective in various

competitions and benchmarks, displaying its state-of-the-art performance. InceptionV3 has a modular design, making it adaptable for different requirements in image analysis tasks.

The model's architecture facilitates parallel processing, leading to faster training times on hardware that supports parallel computation. It has been influential in the development of subsequent Inception models, each aiming to enhance specific aspects of performance. Its versatility and high accuracy make it a popular choice in projects where sophisticated image recognition is essential.

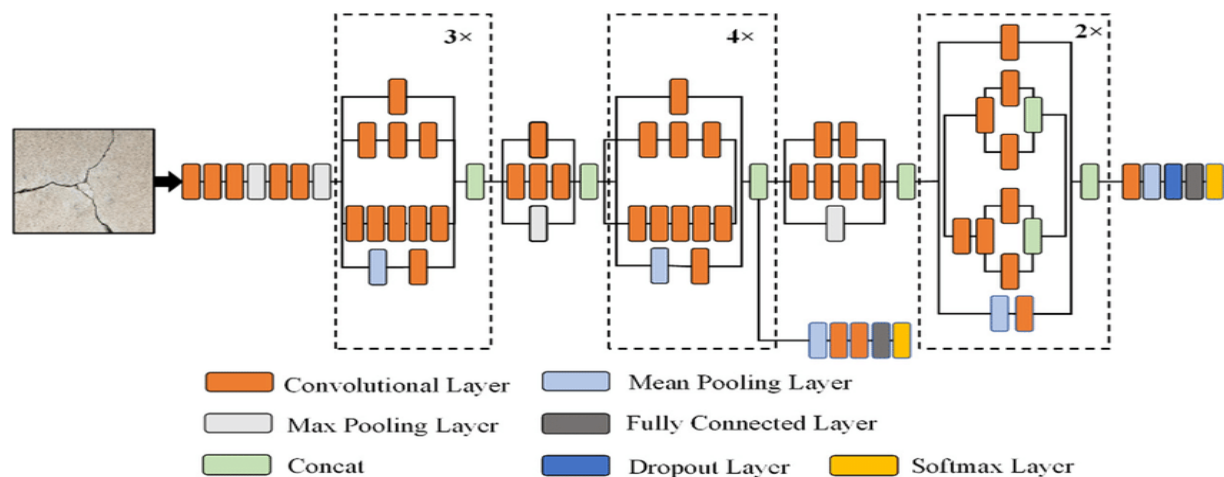


Fig. 9 Architecture of InceptionV3

## 7.2. SEQUENTIAL:

The line model sequential is a crucial component when building neural networks using the Keras library. This line initializes a sequential model, which is a linear stack of layers. The concept of a sequential model is fundamental to Keras, as it allows you to easily create a neural network in a step-by-step fashion. Here are some points you could consider including:

**Model Initialization:** This line is the starting point for creating a neural network model using the Keras Sequential API. It signifies the beginning of the model-building process.

**Sequential Model:** The Sequential class allows for the creation of a linear stack of layers in a neural network. This implies that the model is constructed layer by layer in a sequential manner, with each layer connected to the previous one.

**Layer Stacking:** Neural networks are composed of layers, and the sequential model simplifies the process of adding layers one after another. This stacking of layers defines the flow of information through the network.

**Ease of Use:** The Sequential API is user-friendly and is often recommended for beginners. It provides a clear and intuitive way to build models without requiring an in-depth understanding of the underlying complexities.

**Readability:** The use of Sequential enhances the readability of the code. The model is constructed in a top-down manner, making it easier to understand and modify.

**Flatten:** The Flatten layer is essential for converting the multi-dimensional output from the previous convolutional layers or pre-trained base model into a one-dimensional array. This is necessary before transitioning to fully connected (dense) layers.

**Dense:** The Dense layer is responsible for learning patterns from the flattened input. The choice we made is 256 units implies that this layer has 256 neurons or nodes. The ReLU activation function introduces non-linearity to the model, enabling it to learn complex relationships within the data.

**Dropout:** Dropout is a regularization technique that helps prevent overfitting by randomly "dropping out" a fraction of units during training. A dropout rate of 0.5 means that, during each training iteration, 50% of the neurons in the previous layer will be randomly excluded, promoting better generalization.

The final Dense layer produces the model's output. The choice of no of units corresponds to the number of classes in the classification task. The softmax activation function is commonly used for multi-class classification, converting the raw model output into probability scores for each class.

These above steps represent the construction of the fully connected layers in your neural network. The Flatten layer prepares the data for the dense layers, the first Dense layer captures patterns, Dropout helps prevent overfitting, and the final Dense layer produces the output probabilities for classification.

### **7.3. TRANSFER LEARNING INTEGRATION:**

In current approach, we embraced the principles of transfer learning to enhance the performance of our neural network model. Transfer learning involves leveraging knowledge gained from a pre-existing model on a related task and applying it to a new, similar task. This approach is particularly advantageous when confronted with limited data for the specific task at hand.

**Methodology:** To implement transfer learning, we adopted a modular approach by introducing a pre-trained neural network architecture, denoted as base model, into our model. The integration was achieved using the Keras library's Sequential API using base model as additional layer.

**Choice of Base Model:** The selection of an appropriate base model is a critical decision in the transfer learning process. We opted for a well-established and pre-trained architecture that demonstrated success on a broader task like ours. This choice allowed us to benefit from the learned features and hierarchical representations captured by the base model.

#### **Benefits of Transfer Learning:**

**Improved Convergence:** Transfer learning accelerates the convergence of our model during training, as the initial weights provided by the pre-trained model offer a more informed starting point.

**Effective Feature Extraction:** The use of a pre-trained model as a feature extractor enhances the network's ability to discern relevant patterns and representations, especially in scenarios where labeled data for the specific task is limited.

**Generalization Capability:** By leveraging knowledge from a broader task, our model gains the ability to generalize well to the nuances of our target task, even with a small dataset.

#### **7.4. COMPILE MODEL:**

The compile method configures the model for training by specifying the loss function, optimizer, and evaluation metrics, the model should learn from the data during training. It sets up the rules for adjusting the model's internal parameters (weights) based on the difference between its predictions and the actual values.

##### **Loss function (sparse\_categorical\_crossentropy):**

The 'sparse\_categorical\_crossentropy' loss function is commonly used for classification tasks where the target labels. It is suitable for scenarios where each input sample belongs to exactly one class.

##### **Optimizer (Adam):**

We selected the Adam optimizer, which is a popular choice for deep learning tasks. Adam adapts the learning rate during training and efficiently handles optimization challenges, often leading to faster convergence.

##### **Metrics (Accuracy):**

We are monitoring the accuracy metric during training. Accuracy represents the proportion of correctly classified samples and is a commonly used metric for classification tasks. It gives us an indication of how well the model is performing on the training data.

#### **7.5. FITTING MODEL:**

The fundamental part of the model training process is using data augmentation. In this process, the model is trained using the Keras fit method with a generator created by datagen.flow. The training is performed over some no of epochs, and each epoch involves processing batches of augmented data. The steps\_per\_epoch parameter ensures the entire training dataset is processed in each epoch, with the specified batch size facilitating efficient training. Data augmentation is applied in real-time to train introducing variations to enhance the model's ability to generalize to different scenarios. This dynamic augmentation is crucial for mitigating overfitting and improving the model's robustness. The chosen batch size and number of epochs are essential hyperparameters, influencing the efficiency of the training process. Monitoring the training progress, including accuracy and loss metrics, is vital for assessing the model's convergence and identifying potential issues such as overfitting or underfitting. This exemplifies a strategic approach for training neural networks with data augmentation, a key practice for enhancing the model's performance on diverse datasets.

#### **7.6. SAVING MODEL:**

The functionality of saving and loading machine learning models using the joblib library is a crucial aspect of model deployment and reusability. By employing the dump function in joblib,

the trained model is serialized and stored in a file. This serialized form retains the learned parameters, architecture, and configuration, allowing for seamless preservation of the model's state. The subsequent use of `load` enables the retrieval of the model from the saved file, creating a new instance named loaded model. This process eliminates the need for retraining, providing a time-efficient and resource-saving approach.

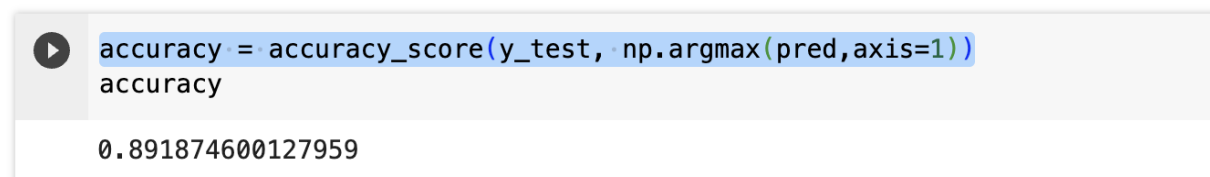
The ability to persist and retrieve trained models is particularly advantageous in scenarios where model deployment is separated from the training environment. It facilitates the integration of pre-trained models into various applications, allowing for quick and efficient model inference without the computational overhead of training. This capability is essential in real-world applications, ranging from deploying models in production environments to sharing models with other researchers or stakeholders.

Furthermore, `joblib`, as a serialization library, is well-suited for handling complex objects, including large machine learning models, efficiently. This is especially beneficial for models with intricate architectures, such as those based on deep learning frameworks. The simplicity of the code exemplifies the user-friendly nature of `joblib`, making it accessible for practitioners across different domains.

It shows a practical solution for model persistence, offering a convenient means to store, share, and deploy trained machine learning models. This capability enhances the reproducibility and scalability of machine learning workflows, contributing to the broader adoption and application of advanced models in diverse contexts.

## 8. MODEL EVALUATION AND METRICS:

### 8.1. ACCURACY:



```
accuracy = accuracy_score(y_test, np.argmax(pred,axis=1))
accuracy
```


0.891874600127959

*Fig. 10 Architecture of InceptionV3*

The above snippet is from the Colab file that calculates the accuracy on the test data, after fitting using above model, we got an accuracy of 89.2%

So, using the model for the test data we got an accuracy of 89.18%

## 8.2. CLASSIFICATION REPORT:

	precision	recall	f1-score	support
0	0.83	0.92	0.88	556
1	0.92	0.94	0.93	281
2	0.96	0.98	0.97	217
3	0.94	0.93	0.94	213
4	0.90	0.70	0.78	296
accuracy			0.89	1563
macro avg	0.91	0.89	0.90	1563
weighted avg	0.89	0.89	0.89	1563

*Fig. 11 Classification Report of Test Data*

The table provided above show the classification report summarizing the performance of a model on a ship images identification task. Here's how you can interpret the data for your project report:

**Precision:** Indicates how accurate the predictions are; out of all the predictions made for a class, how many were correct. The model performs best in identifying class 2 (Carrier) with a precision of 0.96 and has the most difficulty with class 0 (Cargo) at 0.83 precision.

**Recall:** Shows the ability of the model to find all the relevant cases within a class. Class 2 (Carrier) again scores highest with a recall of 0.98, suggesting that it rarely misses any Carrier ships. Class 4 (Tankers) has the lowest recall at 0.70, indicating that some Tanker ships are being missed by the model.

**F1-Score:** Combines precision and recall into a single metric by taking their harmonic mean. It is particularly useful when the class distribution is uneven. The highest F1-score is for class 2 (Carrier) at 0.97, while the lowest is for class 4 (Tankers) at 0.78. **Support:** Refers to the number of actual occurrences of the class in the specified dataset. For instance, class 0 (Cargo) has the most samples at 556, while class 3 (Cruise) has the fewest at 213.

**Accuracy:** Overall, the model accurately classifies 89% of the total cases, which is a solid performance but may still leave room for improvement.

**Macro Average:** Averages the metric scores for each class without considering the class imbalance. The macro average is 0.91 for precision and 0.89 for both recall and the F1-score.

**Weighted Average:** The macro average, but it also considers the support of each class, giving more weight to larger classes. The weighted average is 0.89 for precision, recall, and the F1-score, aligning with the overall accuracy.

### 8.3. CONFUSION MATRIX:

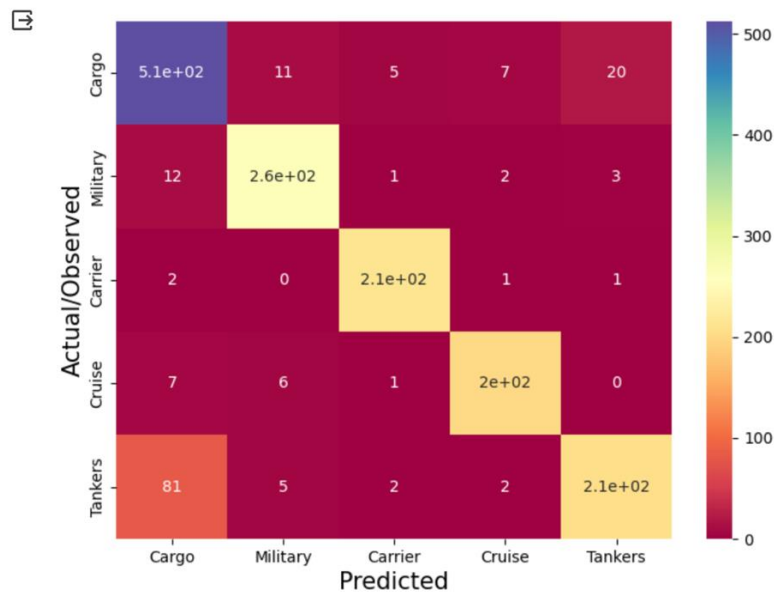


Fig. 12 Confusion Matrix of Test Data

The above image is confusion matrix, which is a tool used in machine learning to evaluate the performance of classification models. It presents the number of observations for each actual class compared to the model's predictions. Here's a general interpretation of the information that such a matrix conveys:

The rows (y-axis) represent the actual classes, which in this case are types of ships such as Cargo, Military, Carrier, Cruise, and Tankers. The columns (x-axis) represent the predicted classes made by the model.

Each cell in the grid shows the count of predictions for each combination of predicted and actual class. The diagonal cells, from the top left to bottom right, typically represent correct predictions where the predicted class matches the actual class. These are often shaded differently to stand out. Off-diagonal cells show misclassifications where the predicted and actual classes do not match. The color scale on the right side provides a visual cue about the number of observations, with one end of the scale (usually darker colors) representing higher counts and the other end (lighter colors) representing lower counts.

For instance, a cell with "5.1e+02" (which means 510) for the actual Cargo class predicted as Cargo indicates a high number of correct predictions for that class. Conversely, a cell with "81" for the actual Tankers class but predicted as Cargo indicates frequent misclassifications between these two classes.

The confusion matrix is crucial for identifying not just the overall accuracy of the model, but also for revealing specific areas where the model might be performing poorly, such as frequently confusing two classes. It helps in understanding the model's predictive behavior and guides further refinement to improve its accuracy.

## 9. CONCLUSION:

It is a significant advance in marine technology, using the InceptionV3 architecture to leverage a powerful Neural Network model to address critical difficulties. The project's importance is highlighted by the dedication to accuracy, effectiveness, and dependability in ship categorization using survey boat photos.

The 6252 labelled photos from several ship categories in the foundational dataset offer a strong foundation for training the neural network. The weighted F1 score highlights the importance of a balanced approach in evaluation metrics, which is essential for practical applications in environmental protection, fisheries management, and marine surveillance. Supported through the strategic selection of InceptionV3, which is well-known for working well in transfer learning situations. However, the recognized constraints of InceptionV3, such as processing requirements and interpretability problems, point to possible areas for model deployment improvement.

A high-accuracy, dependable, and efficient ship classification system is urgently needed, and this project meets that need with well-defined objectives and a clear problem statement. The Governmental Maritime and Coastguard Agency's objectives are well aligned with the suggested approach, which improves maritime security, safety, and environmental preservation.

The development and implementation of a Neural Network model for ship classification is described in depth in this extensive paper, which covers topics including data preparation, transfer learning, model building, and data augmentation. The model's accuracy and resilience are enhanced by the sequential model construction, the incorporation of InceptionV3, and efficient data augmentation methods.

The model's total accuracy of 89.18% is further validated by metrics and evaluation. The model's advantages and disadvantages are revealed by the thorough study using precision, recall, F1-score, and the confusion matrix.

Essentially, this project presents a comprehensive and inventive method of classifying ships, utilizing state-of-the-art technology to make a substantial contribution to marine security, safety, and environmental preservation. The model's sustained usefulness in practical maritime applications will be ensured by continuous improvement and optimization as the project develops.



## REFERENCES:

- [1] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. arXiv preprint arXiv:1512.00567.
- [2] Kornblith, S., Shlens, J., & Le, Q. V. (2019). Do better ImageNet models transfer better arXiv preprint arXiv:1805.08974.
- [3] Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition. A new model and the kinetics dataset. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4724-4733.
- [4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, 770-778.
- [5] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. Proceedings of the IEEE conference on computer vision and pattern recognition, 4700-4708.