Project Initialization and Planning Phase

Date	15 April 2024
Team ID	Team-738165
Project Title	Neural Networks Ahoy: Cutting-Edge Ship Classification for Maritime Mastery
Maximum Marks	3 Marks

Project Proposal (Proposed Solution) template

The purpose of ship classification is to identify various types of ships as accurately as possible, which is of great significance for monitoring the rights and interests of maritime traffic and improving coastal defense early warnings. The images in the data belong to 5 categories of ships - Cargo, Carrier, Military, Cruise and Tankers.

Project Overview	
Objective	The objective of this project is to develop a ship classification system utilizing deep learning techniques, specifically Convolutional Neural Networks (CNNs), to accurately classify images of ships into one of ten categories: Aircraft Carrier, Bulkers, Car Carrier, Container Ship, Cruise, DDG, Recreational, Sailboat, Submarine, and Tug. By leveraging the power of neural networks, the system aims to enhance maritime traffic monitoring, safeguard coastal defense, and streamline early warning systems.
Scope	The project adaptively accesses and enhances the classification of ships, employing deep learning for a more sturdy and efficient system.
Problem Statement	
Description	Addressing the limitation of current technology, lack of resources, and human errors in the current ship classification systems, that affects monitoring, security and alertness of the authorities
Impact	Ship classification holds significant implications for maritime traffic monitoring, coastal defense early warnings, and various other maritime-related applications.

Proposed Solution	
Approach	1. **Data Preparation:** Organize the dataset into categories using the provided train.csv file. 2. **Model Selection:** Utilize the VGG16 model, a pre-trained CNN architecture, for ship classification tasks. 3. **Model Customization:** Modify the top layer of the VGG16 model to adapt it to the specific ship classification requirements. 4. **Training:** Train the customized model using the prepared dataset to learn the distinctive features of each ship category. 5. **Deployment:** Deploy the trained model using the Flask framework to create a user-friendly interface for ship classification.
Key Features	**Classification Accuracy:** The model aims to accurately classify ships into ten categories: Aircraft Carrier, Bulkers, Car Carrier, Container Ship, Cruise, DDG, Recreational, Sailboat, Submarine, and Tug. Achieving high classification accuracy is crucial for effective maritime traffic monitoring and coastal defense. **Transfer Learning with VGG16:** Leveraging the pre-trained weights of VGG16 enables efficient training on a relatively small ship dataset. Transfer learning helps in overcoming the limitations of insufficient training data and reduces training time. **Model Adaptation:** Customizing the top layer of VGG16 for ship classification ensures that the model learns relevant features specific to ship types. Fine-tuning the pre-trained model on the ship dataset enhances its ability to distinguish between different ship categories. **Scalable Deployment:** The model is deployed as a web service using Flask, allowing easy access for real-time ship classification. Scalability features ensure that the application can handle multiple requests concurrently without compromising performance. **Real-time Prediction:** Users can upload images of ships through the web interface, and the deployed model provides real-time predictions on the ship category. Quick response times enable timely decision-making in maritime monitoring and defense scenarios.

Resource Requirements

Resource Type	Description	Specification/Allocation
Hardware		
Computing Resources	CPU/GPU specifications, number of cores	**CPU/GPU:** A machine with a mid-range to high-end GPU is recommended for faster training. An NVIDIA GeForce RTX 3050 or higher would be suitable. **RAM:** 16GB of RAM to comfortably handle dataset loading, preprocessing, and model training.
Memory	RAM specifications	**RAM:** 16GB DDR4 RAM. GPU Memory: At least4GB of VRAM on the GPU for training with medium-sized batches.
Storage	Disk space for data, models, and logs	**Disk Space:** A minimum of 512GB SSD storage for storing datasets, model checkpoints, and related files. **SSD:** Samsung 970 EVO Plus NV Me M.2 SSD for fast data access during preprocessing and training.
Software		

Frameworks	Python frameworks	**Flask:** Version 2.0+ for deploying the trained model as a web service. **TensorFlow:** TensorFlow 2.x with Keras API for building and training the model. **NumPy:** Version 1.20+ for numerical computations and handling arrays.
Libraries	Additional libraries	**Torch:** Version 2.3.0+ for GPU Utilization. **Cuda:** Version 12.3+ for achieving parallel processing capabilities of Nvidia GPUs
Development Environment	IDE, version control	**Visual Studio Code**
Data		
Data	Source, size, format	Kaggle dataset, 453MB, 10259 images