Deep Learning Homework (Megajánlott jegyért)

Group: DeepSea

2018 Airbus Ship Detection Challenge

Team Members:

- Csáki Márton (R0OQD4)
- Marák Levente (K2DE0K)
- Ogleznyev Pável (GRKO04)

1. Introduction and Related Work

Domain Overview: Satellite imagery analysis is critical for maritime security and environmental monitoring. This project focuses on semantic segmentation using the Airbus Ship Detection Challenge dataset. Unlike simple classification, our goal is to locate ships at the pixel level (binary masking) amidst challenges like cloud cover, varying ship sizes, and extreme class imbalance.

Related Work & Model Selection: In the field of object detection, architectures like YOLO or Mask R-CNN are often state-of-the-art.

- *Why not YOLO?* YOLO is optimized for bounding boxes, which lacks the pixel-level precision required for detailed ship segmentation.
- *Why U-Net?* We selected the U-Net architecture, because its originally designed for biomedical imaging, U-Net is the industry standard for segmentation where localization is just as important as classification. It is lighter and often converges faster on segmentation tasks compared to heavier R-CNN based models.

2. System Architecture

We implemented a custom U-Net consisting of two main paths:

- Encoder (Contracting Path): Extracts features using Convolutional blocks (Conv2D + ReLU + BatchNormalization) and MaxPooling. We incorporated Dropout to prevent overfitting.
- Decoder (Expansive Path): Reconstructs the mask using Transposed Convolutions.

- Skip Connections: Critical for satellite imagery, these connections concatenate high-level features from the encoder with the decoder, preserving spatial details of small ships that would otherwise be lost.

3. Implementation and Data Strategy

The dataset contains over 192k images (768x768). Handling this efficiently required specific solutions.

3.1 Data Pipeline & Environment Optimization

- Colab SSD Pipeline: A major bottleneck was reading data from Google Drive. We designed a pipeline to copy the dataset to the local Google Colab SSD instance before training. This significantly reduced I/O latency.
- RLE Decoding: We implemented helper functions to convert Run-Length Encoded strings into dense NumPy masks.

3.2 Addressing Class Imbalance

The dataset is heavily imbalanced (~78% empty sea). Training on raw data would result in a model that constantly predicts "background."

- Undersampling: We applied a strategic 1:3 ratio (1 ship image for every 3 empty images), ensuring the model learns to distinguish ships from waves without being overwhelmed by empty space.

3.3 The "Smart Cropping" Strategy

To handle the 768x768 resolution without losing small ship details via resizing:

- Dynamic Crops: We implemented a generator that extracts 384x384 crops.
- Logic: If a ship is present, the crop centers on the ship (with random jitter). If empty, a random crop is taken. This forces the model to focus on relevant features rather than vast empty oceans.

4. Training and Hyperparameter Optimization

Configuration:

- Loss Function: Combined BCE + Dice Loss. BCE handles classification, while Dice Loss optimizes the pixel-overlap, which is superior for imbalanced segmentation tasks.
- Optimizer: Adam Optimizer.

Hyperparameter Tuning: We did not rely solely on default values. We conducted iterative experiments:

- *Learning Rate:* We utilized a ReduceLROnPlateau callback. Initial high learning rates led to unstable gradients, so we tuned the start rate and decay factor based on validation loss stagnation.

- *Batch Size:* Optimized to maximize GPU memory usage without causing OOM (Out of Memory) errors.

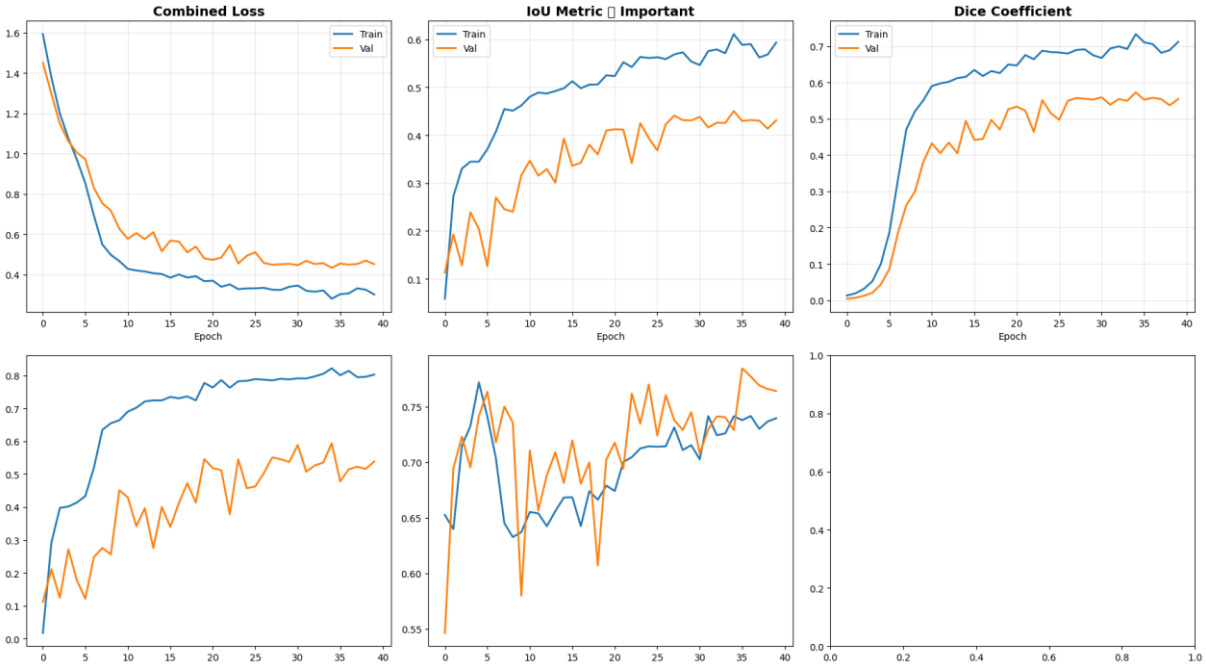
5. Results and Evaluation

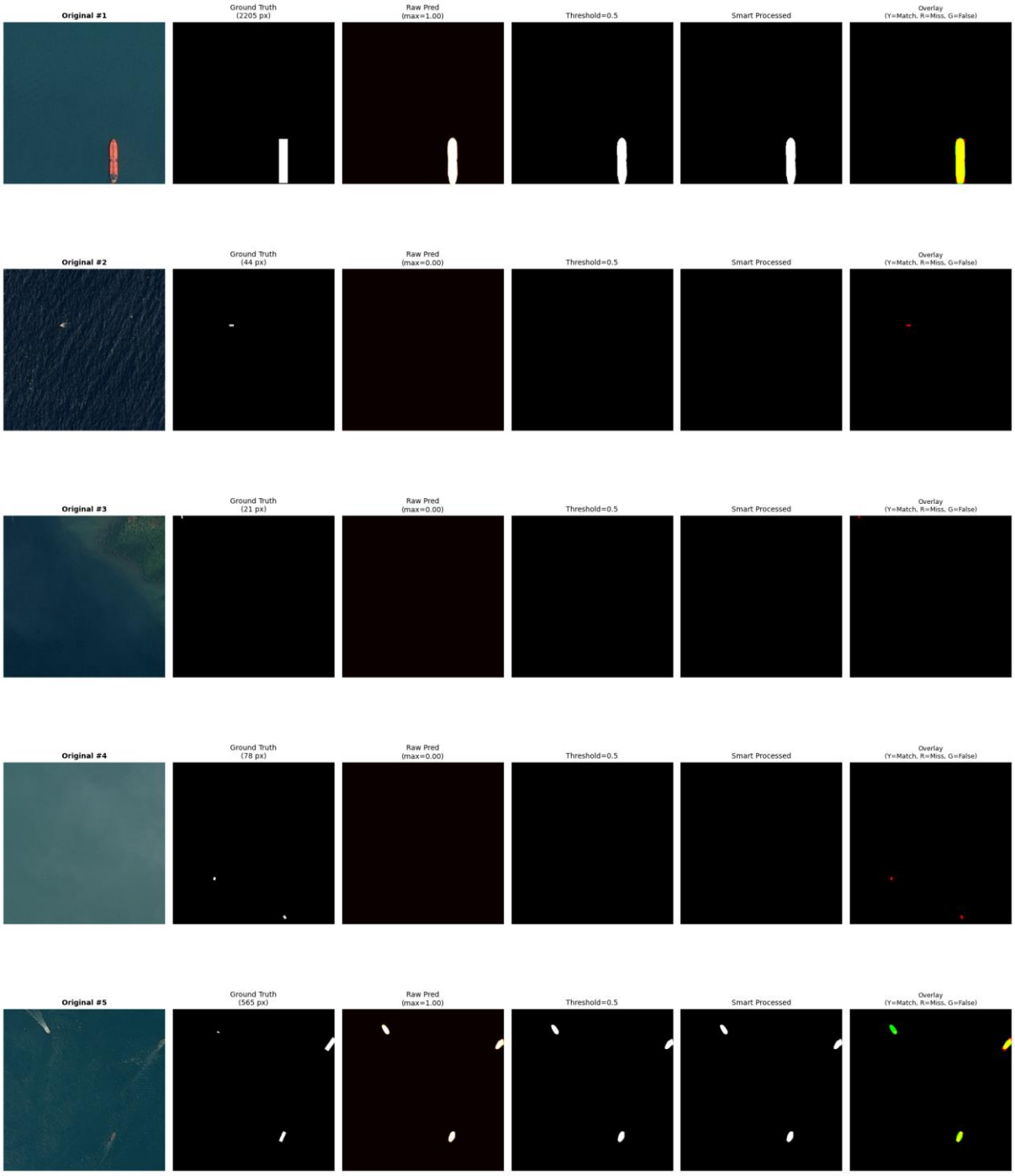
Metrics: The model was trained for 40 epochs.

- Validation IoU (Intersection over Union): ~0.43 – 0.59
- Validation Dice Score: ~0.55 – 0.71

Testing Strategy: Due to the structure of the competition data (where the ground truth for the test set is hidden), we utilized a stratified hold-out Validation set (20% of data) as our proxy for Testing. This set was strictly isolated from the training process to ensure unbiased evaluation.

Visual Performance: Post-processing (thresholding > 0.5 and noise removal) showed that the model successfully segments ships even in cloudy conditions. The *Smart Cropping* strategy was validated as effective, as the model detected small vessels that simple resizing would have obscured.





6. Project Development & Attribution

AI Usage Declaration: In the implementation phase, we utilized AI assistants (LLMs) to accelerate code, but the major conceptual decisions made by our own most of the times.