Deep learning a gyakorlatban Python és LUA alapon

Airbus Ship Detection Challenge

Team BHAF

Only Draft version. (Megajánlott jegyért)

Bilszky Márk | Sági Máté Csaba | Pesti Patrik
C0QVQN | PGBWYA | E329CD

University Project

December 2024

Abstract

English Abstract:
This project focuses on detecting ships in satellite images and generating segmentation masks to highlight their exact locations. Using a simplified Mask R-CNN architecture with a ResNet50 backbone for feature extraction, the model generates binary masks that outline the ships. The implementation covers data preparation, visualization, model training, and evaluation, leveraging transfer learning to ensure robust performance. While the current results are promising, further refinements, including higher-resolution masks and data augmentation, are planned to improve accuracy and segmentation quality.

Magyar Kivonat:
Ez a projekt műholdképek elemzésére és a hajók pontos helyét kiemelő szegmentációs maszkok létrehozására összpontosít. Egy egyszerűsített Mask R-CNN architektúrát használunk, amelyben a ResNet50 backbone biztosítja a jellemzők kinyerését, és bináris maszkokat generál a hajók körvonalazására. Az implementáció magában foglalja az adatok előkészítését, vizualizációját, a modell betanítását és kiértékelését, miközben a transfer learning módszert alkalmazza a megbízható teljesítmény érdekében. Az eddigi eredmények ígéretesek, de a pontosság és a szegmentáció minőségének javítása érdekében további fejlesztések – például nagyobb felbontású maszkok és adataugmentáció – vannak tervben.

Contents

1 Introduction

This project is from a kaggle competition [5], titled the *Airbus Ship Detection Challenge*, aims to detect ships in satellite imagery and generate segmentation masks to identify their exact locations. The dataset includes diverse scenarios, such as images with no ships or multiple ships of varying sizes. Our project implements a simplified Mask R-CNN architecture, focusing on binary segmentation tasks.

2 Related Work and Existing Solutions

Detecting objects in satellite imagery is a well-studied problem in computer vision. The Mask R-CNN framework is widely recognized for its versatility in object detection and segmentation [4]. By tailoring the architecture to binary segmentation tasks, this project simplifies the model to focus on ship detection, avoiding the complexity of bounding box regression.
Existing solutions [8], [7] for the Airbus Ship Detection challenge emphasize the use of data augmentation techniques such as rotation, flipping, and scaling to improve model robustness. Many participants employ pre-trained networks, like ResNet and EfficientNet, to leverage learned features [3]. Hyperparameter tuning, experimenting with various model architectures, and applying ensemble methods, where multiple models are combined, are also effective strategies for enhancing detection accuracy. These solutions aim to improve the model's generalization and performance on unseen data. [6]

3 System Design: Network Architecture

3.a Overview

The neural network is structured around a simplified Mask R-CNN architecture [4], designed for binary segmentation tasks. The system leverages a ResNet50 backbone as its feature extractor [3]. During training, the ResNet50 layers are initialized with pre-trained weights, and their parameters are frozen to retain the learned features. This helps reduce computational load and speeds up convergence.

Following the backbone, the network includes the mask head, which processes the feature maps from ResNet50 to generate binary masks. This mask head consists of several convolutional layers and transpose convolutional layers (as seen in Figure 3), enabling the network to upsample and refine the feature maps into a segmentation output. The final layer generates a binary mask of shape 384×384 384×384, marking ship locations in the input satellite image.

Layer (type)	Output Shape	Param #
input_image (InputLayer)	(None, 384, 384, 3)	0
functional_3 (Functional)	[(None, 96, 96, 256), (None, 48, 48, 512), (None, 24, 24, 1024)]	8,589,184
conv2d_3 (Conv2D)	(None, 24, 24, 256)	2,359,552
conv2d_transpose_8 (Conv2DTranspose)	(None, 48, 48, 256)	590,080
conv2d_transpose_9 (Conv2DTranspose)	(None, 96, 96, 256)	590,080
conv2d_transpose_10 (Conv2DTranspose)	(None, 192, 192, 256)	590,080
conv2d_transpose_11 (Conv2DTranspose)	(None, 384, 384, 256)	590,080
mask_output (Conv2D)	(None, 384, 384, 1)	257

Total params: 13,309,313 (50.77 MB)
Trainable params: 13,278,721 (50.65 MB)
Non-trainable params: 30,592 (119.50 KB)

Figure 1: Model layers

3.b Key Features of the Approach

- **Simplification:** Focuses solely on mask generation, for pictures with lowered resolution.
- **Efficiency:** Leverages transfer learning with ResNet50 for robust feature extraction.
- **Scalability:** Predictions could be enhanced by further data processing and by elevating model resolution.

4 Implementation

The implementation is centered around the *deepl_bp.ipynb* Jupyter Notebook. It integrates the Kaggle API for data downloading and provides a streamlined workflow for model training and evaluation. The notebook also demonstrates mask predictions overlaid on satellite images [1].

5 Data Acquisition and Preparation

5.a Dataset Source

The dataset is sourced from Kaggle and includes around 200 thousand satellite images with corresponding segmentation masks encoded in Run-Length Encoding (RLE) [2].

5.b Preprocessing Steps

- Decoding RLE masks into binary masks.
- Balancing the dataset by controlling the proportion of images with and without ships. (7:3 ship no ship ratio) In the end with this around a 100 thousand images were used for the training.

6 Training Process

6.a Configuration

The model employs a binary cross-entropy loss function with the Adam optimizer. Training and validation datasets are split to ensure balanced evaluation. (3. figure shows the model parameters)

6.b Challenges

The long training time per epoch (2 hours) presents challenges in observing the learning curve. This makes the hyper-parameter optimization process difficult, so we will implement this on a smaller dataset for the final draft and then insert those parameters in the final model.

7 Evaluation: Metrics, Errors

The ResNet50 backbone ensures robust feature extraction, reducing computational burden, and leveraging pre-trained weights for better performance on limited training data.
The model generates binary masks highlighting ship locations. Current results indicate that the masks require refinement for even more precise boundary alignment. In any case with the binary cross-entropy loss function on our resolution, we have achieved an accuracy of 0.9985 and loss of 0.0043 on training dataset. Of course this is not the same as the kagle evaluation metric, there our model would achieve a worse result, thanks to the lower resolution bounding boxes. Still, all in all, the model can detect ships really well, with a crude outline as can be seen on the picture below from the validation dataset.

8 Hyperparameter Optimization

Hyperparameter optimization plays a critical role in improving the model's performance and generalization while maintaining computational efficiency. This section details the tuning process and the resulting optimal configuration.

8.a Approach and Methodology

To optimize the model, the following hyperparameters were selected for tuning:

- **Learning Rate:** Controls the step size for weight updates during optimization, with possible values $[10^{-2}, 10^{-3}, 10^{-4}]$.
- **Mask Head Filters:** Determines the number of filters in the convolutional layers of the mask head, with choices $[128, 256, 512]$.
- **Activation Function:** Defines the non-linearity applied in the network, with options [ReLU, ELU, Tanh].
- **Optimizer:** Adam optimizer was selected as the sole optimization algorithm for its balance between efficiency and stability, in order to reduce the optimization's computational time.

The tuning was conducted using grid search, systematically testing all combinations of the above hyperparameters on a smaller subset of the dataset. The evaluation was based on binary cross-entropy loss and validation accuracy. Early stopping was applied to prevent overfitting during training.

8.b Results

The optimal configuration identified through the hyperparameter tuning process is as follows:

- **Learning Rate:** 10^{-2}
- **Mask Head Filters:** 128
- **Activation Function:** $ReLU$
- **Optimizer:** Adam with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$)

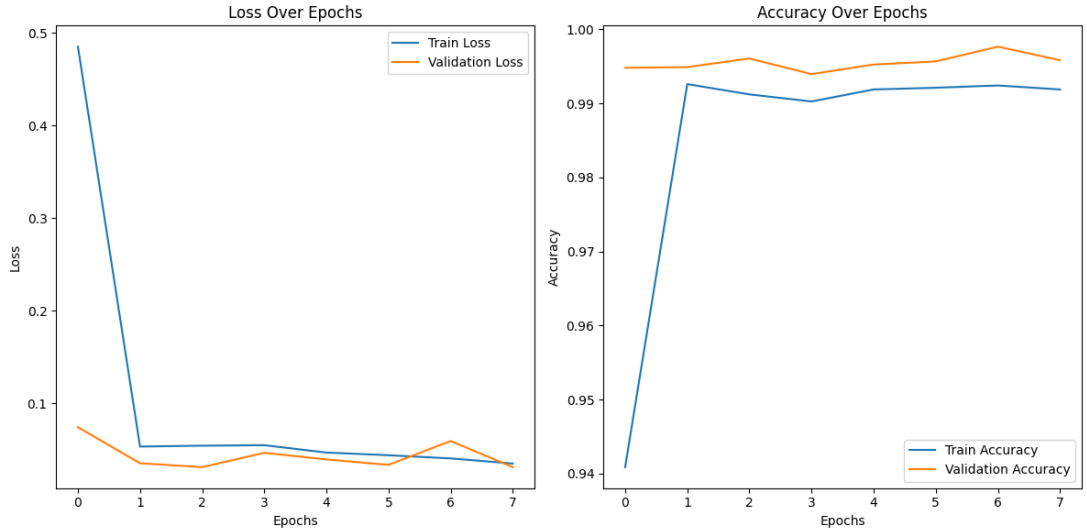

Figure 2: Results with parameters from the hyperparameter optimization (val loss: 0.045)

8.c Final parameter choice

Throughout of the hyperparameter optimization we had to run the test on a small dataset, due to computational limits. After the optimization we tested 2 high accuracy variations on large datasets. One of it were the results from the optimization, however performed worse, then our other selected parameter composition. Thus we settled with the following configuration:

- **Learning Rate:** 10^{-4}
- **Mask Head Filters:** 256
- **Activation Function:** $ReLU$
- **Optimizer:** Adam with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$)

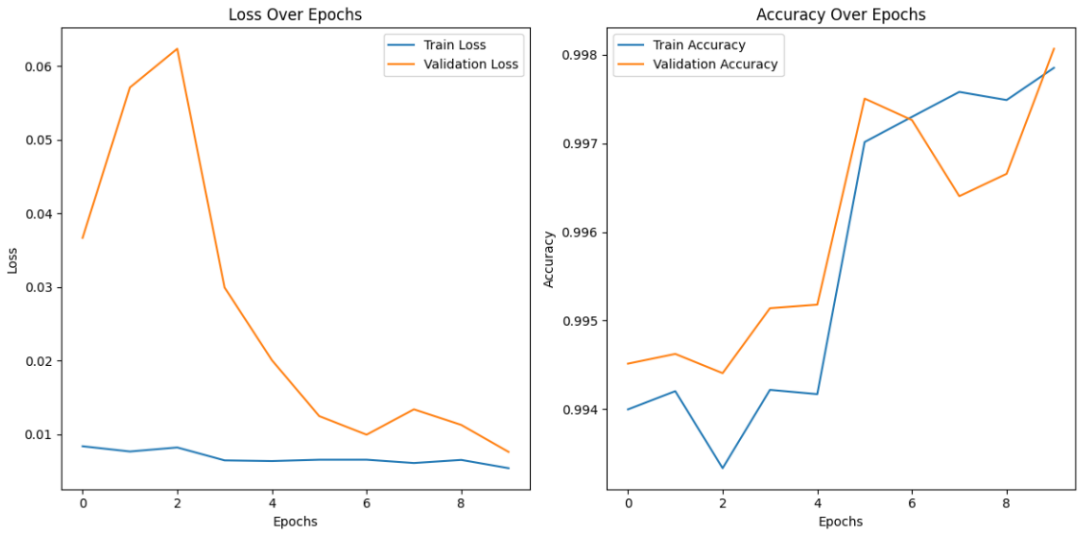

Figure 3: Results with the used parameters (val loss: 0.0076)

8.d Challenges and Insights

The main challenges encountered during the optimization process were:

- **Training Time:** Long training durations on the full dataset limited the exploration of additional hyperparameter configurations.
- **Dataset Subsetting:** While subsetting expedited experiments, it may have led to suboptimal performance when scaling to the full dataset.

Despite these challenges, the tuned hyperparameters significantly improved the model's training stability and performance. Future iterations will consider advanced tuning techniques, such as Bayesian optimization, to further refine the model.

8.e Impact of Hyperparameters

The learning rate had the most substantial effect on convergence, with 10^{-2} balancing speed and stability. The choice of 256 filters for the mask head struck a balance between model capacity and overfitting risk. The ReLU activation function provided consistent and reliable non-linearity, aligning well with the Adam optimizer's performance.

9 Future Work and Conclusion

9.a Planned Improvements Till the Final Deadline

- Implementing hyper-parameter optimization on a smaller dataset.
- Implementing higher-resolution ship detection.
- Evaluate the model based on more metrics.
- Enhancing the documentation.

9.b Summary

The project demonstrates the power of transfer learning and a simplified Mask R-CNN model for ship detection. Although the results are promising, further refinements are required to enhance accuracy and mask precision.
Large language models were used for code completion and for English translation during the documentation.

10 Appendix

Throughout our project, we leveraged the support of AI tools, especially ChatGPT, to address various challenges. Whenever we faced difficulties with syntax or comments, the model provided valuable guidance. Moreover, it was an essential aid in refining this documentation, particularly by assisting with grammar corrections. In summary, ChatGPT was an essential asset to the success of our work, and its input warrants recognition.

References

[1] Bilmark. *Deep Learning - ALPJAI Main Project: DeepL BP Notebook*. Accessed: 2024-12-08. 2024. URL: `https://github.com/bilmark0/Deep-Learning-alpjai-Main-Project/blob/main/deepl_bp.ipynb`.

[2] GeeksforGeeks. *Run-Length Encoding and Decoding*. `https://www.geeksforgeeks.org/run-length-encoding/`. 2023.

[3] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[4] Kaiming He et al. “Mask R-CNN”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2961–2969.

[5] Kaggle. *Airbus Ship Detection*. Accessed: 2024-12-08. 2024. URL: https://www.kaggle.com/competitions/airbus-ship-detection.

[6] Kaggle. *Airbus Ship Detection Challenge: Discussion on Solutions*. Accessed: 2024-12-08. 2024. URL: https://www.kaggle.com/competitions/airbus-ship-detection/discussion/71595.

[7] Authors Unknown (update manually). “Enhanced Mask R-CNN for Ship Detection and Segmentation”. In: *Advances in Ship Detection using Deep Learning*. Springer, 2021, pp. 315–328. DOI: DOINeeded(updatemanually). URL: https://link.springer.com/article/10.1007/978-981-15-5243-4_16.

[8] Authors Unknown (update manually). “Mask R-CNN for Ship Detection & Segmentation”. In: *Deep Learning for Ship Detection and Segmentation in Satellite Imagery*. Springer, 2020, pp. 245–260. DOI: 10.1007/978-981-15-5243-4_14. URL: https://link.springer.com/chapter/10.1007/978-981-15-5243-4_14.