



Experimental evaluation of ship detection using U-Net with various backbone networks

Vishalakshi Kamath

Master's Thesis
Master of Engineering - Big Data Analytics
2022

MASTER'S THESIS	
Arcada	
Degree Programme:	Big Data Analytics
Identification number:	8732
Author:	Vishalakshi Kamath
Title:	Experimental evaluation of ship detection using U-Net with various backbone networks.
Supervisor (Arcada):	Amin Majd, D.Sc. (Tech.) and PhD
Commissioned by:	
<p>Abstract: The growth in global trade has led to the growth in global ship traffic. Maritime security, safety and tracking has become more critical. Organizations and governments globally are developing applications to improve maritime surveillance. Among the widely used solutions, satellites are used to remotely scan and record images of waterways and shores. These image data feeds are processed using deep learning algorithms like Convolutional Neural Network (CNN) to detect ships with great precision. The challenges in this process are the image quality, size of ships, other varied noise within data and the computational requirements of handling and managing big data. The thesis work covers evaluating the U-Net model with popular transfer learning networks for semantic image segmentation on Airbus ship detection data. The backbones are pretrained on ImageNet dataset. Deep learning models were developed using Keras API with TensorFlow backend. The U-Net model with the selected backbones are compared with each other using standard metrics.</p>	
Keywords:	Ship detection, Transfer Learning, Deep Learning, CNN, Satellite images, Semantic Image segmentation, U-Net architecture, Maritime surveillance, ResNet-50, InceptionV3, MobileNetV2, EfficientNetB2 and DenseNet-121.
Number of pages:	50
Language:	English
Date of acceptance:	

CONTENTS

1	Introduction.....	9
1.1	Background	9
1.2	Motivation and Outcome	9
1.3	Thesis Structure	10
2	Background Theory.....	10
2.1	Ship Detection	10
2.2	Challenges in Ship Detection	11
3	Literature Review.....	13
3.1	Deep Learning	13
3.2	Artificial Neural Networks (ANN)	14
3.3	Convolutional Neural Network (CNN).....	15
3.3.1	<i>Forward Propagation</i>	16
3.3.2	<i>Backward Propagation</i>	16
3.4	Semantic Segmentation	17
3.5	Transfer Learning	19
3.5.1	<i>Commonly used Transfer Learning Models</i>	19
3.6	Satellite imagery	24
3.7	Related Work.....	25
4	Methods and material	26
4.1	Dataset	26
4.2	U-Net: Convolutional Networks for Image Segmentation.....	30
4.2.1	<i>U-Net Network Architecture</i>	30
4.2.2	<i>Encoder Network</i>	30
4.2.3	<i>Skip Connections</i>	31
4.2.4	<i>Bridge</i>	31
4.2.5	<i>Decoder Network</i>	31
4.3	Backbone of Network	31
4.4	Evaluation Metrics	32
4.4.1	<i>Intersection over Union</i>	32
4.4.2	<i>F1 Score</i>	32
4.4.3	<i>Precision</i>	33
4.4.4	<i>Recall</i>	33
5	Results And Conclusions.....	34
5.1	ResNet-50 as Backbone	34

5.2	DenseNet-121 as Backbone	35
5.3	InceptionV3 as Backbone.....	35
5.4	VGG-16 as Backbone	36
5.5	MobileNetV2 as Backbone	36
5.6	EfficientNetB2 as Backbone.....	37
5.7	Model Test Result.....	37
5.7.1	<i>U-Net+MobileNetV2</i>	39
5.7.2	<i>U-Net+EfficientNetB2</i>	39
5.7.3	<i>Comparison of U-Net+EfficientNetB2 and U-Net+MobileNetV2</i>	40
5.7.4	<i>Predictions</i>	41
6	Further improvements.....	42
7	References	43
8	Appendices	47
8.1	CSC-Puhti sample batch script	47
8.2	Image prediction results	48

Figures

Figure 1. Optical remote sensing images under different conditions.....	12
Figure 2. Sea surface, ship target and cloud samples	13
Figure 3. Single layer Perceptron / Artificial neuron	14
Figure 4. Multi-Layer Perceptron.....	15
Figure 5. Architecture of Convolutional Neural Network (CNN).....	16
Figure 6. Backward propagation	17
Figure 7. Computer vision tasks.....	17
Figure 8. Semantic segmentation example	18
Figure 9. Semantic segmentation: fully convolutional.....	18
Figure 10. Inception model architecture.....	20
Figure 11. Model Scaling	20
Figure 12. VGG Neural Network Architecture	21
Figure 13. Skip (shortcut) connections in ResNet.....	22
Figure 14. ResNet-34 architecture.....	22
Figure 15. MobileNet architecture.....	23
Figure 16. DenseNet Architecture VS ResNet Architecture	24
Figure 17. The EM Spectrum	25
Figure 18. Ship counts in data	27
Figure 19. Image count with valid ships.....	27
Figure 20. Image data in RLE	28
Figure 21. Images with masks	29
Figure 22. U-Net architecture - Example for 32x32 pixels in the lowest resolution.....	30
Figure 23. Precision prediction possibilities	33
Figure 24. ResNet-50 metrics, 1-15 epoch.....	34
Figure 25. DenseNet-121 metrics, 1-15 epoch.....	35
Figure 26. InceptionV3 metrics, 1-15 epoch	35
Figure 27. VGG-16 metrics, 1-15 epoch	36
Figure 28. MobileNetV2 metrics, 1-15 epoch.....	36
Figure 29. EfficientNetB2 metrics, 1-15 epoch.....	37
Figure 30. Comparison Metrics, at epoch 15.....	38
Figure 31. MobileNetV2 metrics. 1-49 epochs	39

Figure 32. EfficientNetB2 metrics, 1-49 epochs 39
Figure 33. Comparing Metrics - MobileNetV2 & EfficientNetB2 - for test data 40
Figure 34. Predictions..... 41

Tables

Table 1. Comparing model metrics. 37
Table 2. Comparing EfficientNetB2 and MobileNetV2 after 49 epochs 40

ABBREVIATIONS AND TERMS

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
RoI	Region of Interest
R-CNN	Region Based Convolutional Neural Network
Mask R-CNN	Mask Region Based Convolutional Neural Network
Cascade R-CNN	Cascade Region Based Convolutional Neural Network
Faster R-CNN	Faster Region Based Convolutional Neural Network
ResNet	Residual Network
GPU	Graphical Processing Unit
IoU	Intersection of Union
TP	True Positive
FP	False Positive
FN	False Negative

FOREWARD

I would like to express my gratitude to the Big Data Analytics team at Arcada University of Applied Sciences including my study colleagues and the teachers to name a few Dr. Magnus Westerlund, Dr. Leonardo Espinosa Leal and Andrej Scherbakov-Parland without whom the journey was not possible. Special thanks to my supervisor Dr. Amin Majd for his constructive feedback and insightful supervision throughout the project.

Last but not the least, I would like to thank my family especially my spouse, my son and friends for supporting me both spiritually and mentally throughout this journey. Without them, I would never have accomplished this.

I feel incredibly grateful for all of you.

Helsinki

May 2022

Vishalakshi Kamath

1 INTRODUCTION

Today's global economy and trade possibilities allow manufacturers around the world to manufacture products, ship and market the products worldwide. The global trade growth has increased shipping traffic and the need for ship detection and tracking. This study covers some of the Deep Learning, transfer learning methods for ship detection using satellite images with different encoders and loss functions to train segmentation models.

1.1 Background

As the market and consumption becomes global, the products needs to be shipped to consumers using the most economic logistics. Maritime transport logistics is critical for global trade and countries, regions are getting into large agreements for improving multilateral trade possibilities.

As global trade and shipping traffic grows, the infractions on international waterways like ship accidents, piracy and illegal cargo trade are on rise. Maritime security, safety, and tracking requirements have increased. Several companies and government bodies have developed systems to support safe navigation and security defining the border control zones and regulations. These ship detection applications are important also for defense to monitor country territorial shores for any possible enemy ship intrusions.

With the advancement in satellite imaging, it is now possible for satellites to continuously provide image data feeds. Due to remote satellite imaging and sensing constraints, these images require complex processing to enable detection of ships with good precision, avoiding the noise in the image data. Deep learning and advancements in data handling and computing has been successful in improving the object detection models.

1.2 Motivation and Outcome

This thesis is based on the Airbus Ship Detection Challenge using the satellite image data available in Kaggle data sources (Airbus ship detection challenge, 2018). The semantic

segmentation models have been successfully used for various types of images and have been used for ship detection. The study compares the U-Net model with available transfer learning methods like ResNet50, InceptionV3, MobileNetV2, DenseNet-121, VGG-16 and EfficientNetB2. U-Net with these pretrained models were further trained and validated to get standard metrics.

1.3 Thesis Structure

The thesis has the following main chapters:

- Introduction – Provides the business context, motivation and outcomes for research
- Literature review and Theoretical Background
- Materials and Methods
- Results and conclusions
- Future improvements

2 BACKGROUND THEORY

With the increase in global trade and shipping traffic, the navigation, tracking, border control applications are in demand. Various organizations and government bodies are working on developing solutions to ensure safe navigation and security to maritime activities limiting pollution and avoiding illegal cargo movements.

2.1 Ship Detection

Ships are being widely used in global trade, defense, fishing, travel, etc., so ship detection has become critical for various purposes like security, border control, traffic, defense and fishing. Using object detection methods ships can be precisely monitored using images from the space or satellite-based imaging. (Kanjir, Greidanus and Oštir, 2018)

Earlier coastal radars and shore based Automatic Identification Systems (AIS) were extensively used, but these limited the monitoring near the shore. Satellite imaging-based ship detection allows more wider monitoring of the ocean. (Oligschläger, 2020)

Ship detection is a special case of object detection, where the background has the characteristics of the sea surface. Ship detection becomes challenging due to various real-time changes of the background - sea surface, lighting, pollution, ship size and orientation of the ships. The ship object variation and the background affect the image clarity and makes ship detection more difficult than other general object detection. (Li et al., 2020)

2.2 Challenges in Ship Detection

Computer vision is an emerging application, as it can be widely used in many military, social and industrial aspects, such as intelligent video surveillance in traffic management, security monitoring, content-based image retrieval and person re-identification. This process of object detection involves locating and classifying objects in images or video frames often received using remote sensing. The challenges in object detection are due to variations in quantities, positions, forms and sizes of objects, and to identify these from the varying background.

Ship detection is just a specific case of the object detection problem. Unlike other general object detection, the remote sensing-based image quality will be degraded by daylight, mists, clouds, ocean waves and other backgrounds like islands, harbor or coastline. The remote sensing satellite images for ships are mainly affected by three factors – weather conditions, imaging conditions and target property. Weather conditions include the sea state & the cloud conditions. Sea states like quite or stormy large waves will affect the imaging. Clouds conditions may vary no clouds, thin or thick cloud. These affect the grey scale distribution in images and the target-ship features. Imaging conditions mainly the lighting like morning, dusk, noon or night. Also, the weather seasons affect the brightness, contrast due to change in target exposure which impacts the image quality. Target property are the ship objects to be monitored. Ships are moving targets on sea and have variable wakes based on the ship size, speed, sea states and weather conditions. These

wakes are connected to ship and can be incorrectly detected as extension of target ship. Ships are in varying sizes and course of movements can be varying like rotational, so the detection algorithms need to handle scale and rotation. In ship detection the low contrast can cause algorithms to miss the objects and the broken clouds, sea waves, ship wakes, islands and other background objects can cause false ship detection. (Yao, Jiang and Zhang, 2016)

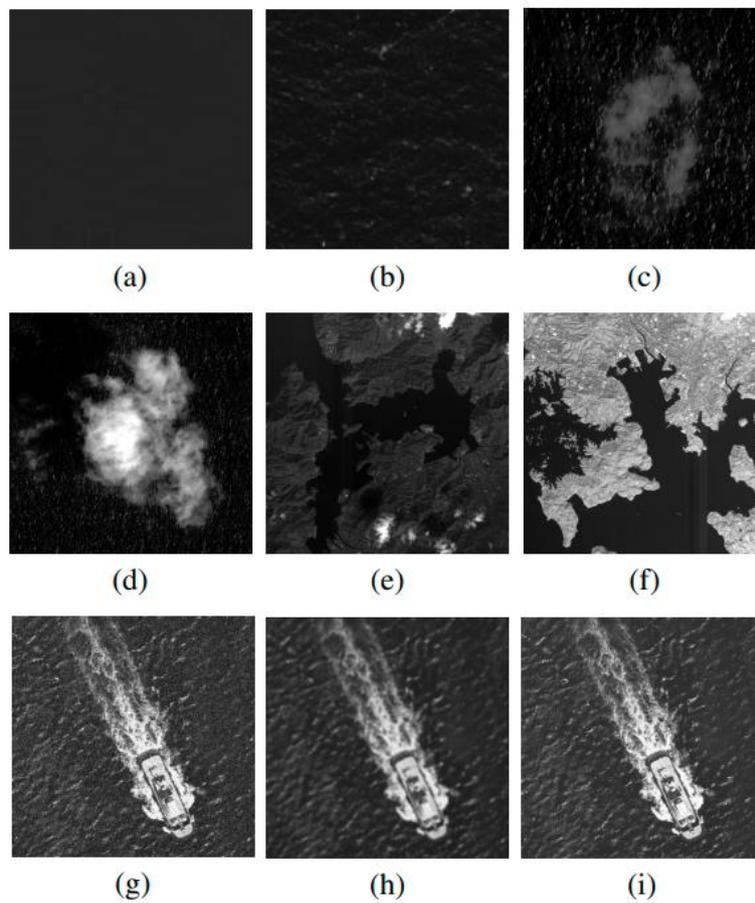


Figure 1. Optical remote sensing images under different conditions.

(a)-(c) – Sea states, (d)-(f) – Cloud state, (g)-(i) – target ship image quality with wakes.

(Yao, Jiang and Zhang, 2016)

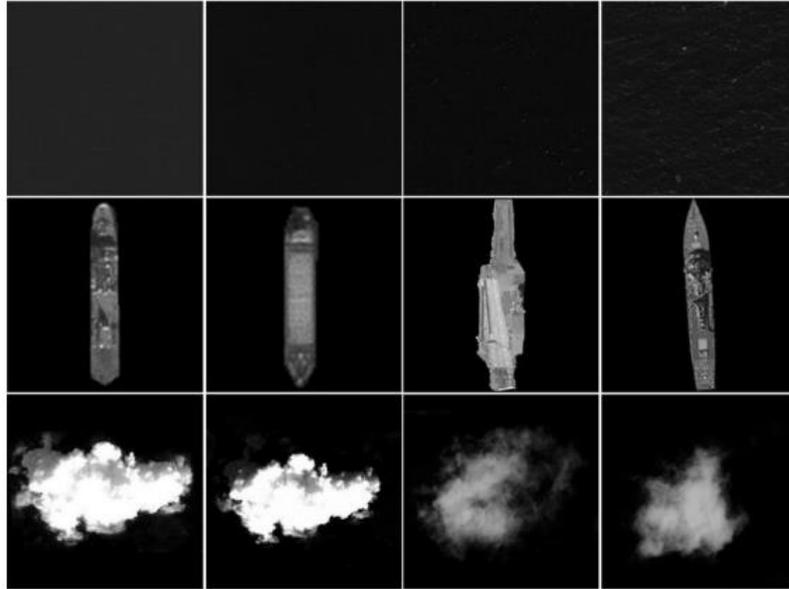


Figure 2. Sea surface, ship target and cloud samples

(Yao, Jiang and Zhang, 2016)

3 LITERATURE REVIEW

3.1 Deep Learning

Deep learning is machine learning using neural network with three or more layers. The neural networks try to learn from large amount of data. A single layer neural network can make approximate predictions and additional layers then improve the accuracy of predictions.

Deep learning is used in many artificial learning (AI) based automation applications where human involvement can be avoided by self-learning automated systems. Deep learning has many applications across industries like digital assistants, automated self-driving vehicles, etc. (IBM Cloud Education, 2020)

3.2 Artificial Neural Networks (ANN)

The artificial neural network (ANN) approach of machine learning is based on simulating the human brain. The Big data analysis needs machine learning to learn large properties within data & their inter-relations to derive an accurate machine model. ANN can model complex non-linear relations within data and are fast, fault tolerant, scalable networks. (Zou, Han and So, 2008)

Simulating the human neuron, a perceptron is an artificial neuron taking various inputs and generating a weighted summation output. The weights are learned during the model training process. (Shanmugamani, 2018)

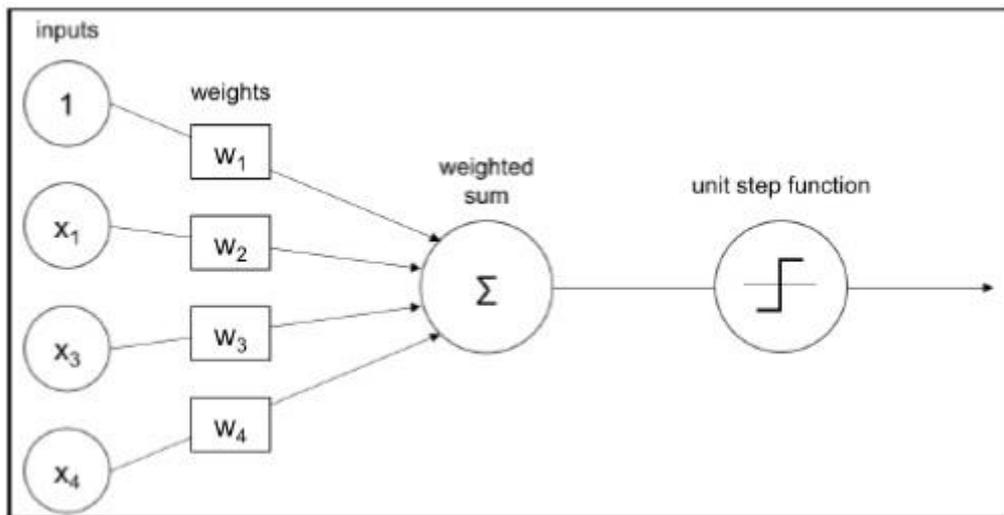


Figure 3. Single layer Perceptron / Artificial neuron

(Shanmugamani, 2018)

A collection of perceptrons and activation functions together form the artificial neural network (ANN). The connected perceptrons form hidden layers or units that map the input to output on non-linear basis.

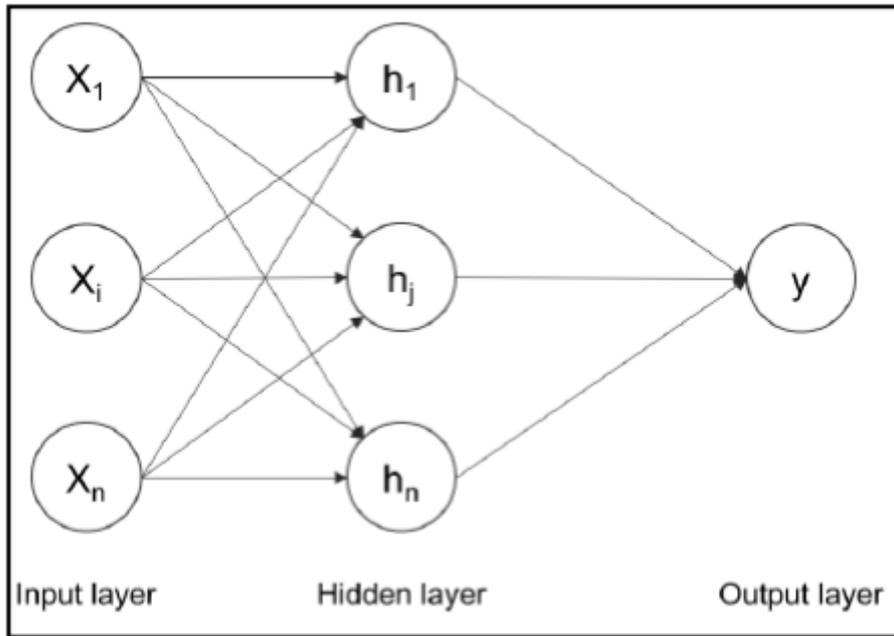


Figure 4. Multi-Layer Perceptron

(Shanmugamani, 2018)

3.3 Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) are multilayer perceptron in regularized form. Here each neuron in one layer is linked to all neurons of next layer. With the regularized form CNNs are less complex and have many applications like face or action detection, text classification, natural language processing, image classification and computer vision functions. CNN uses feed forward architecture. (Alghazo et al., 2021)

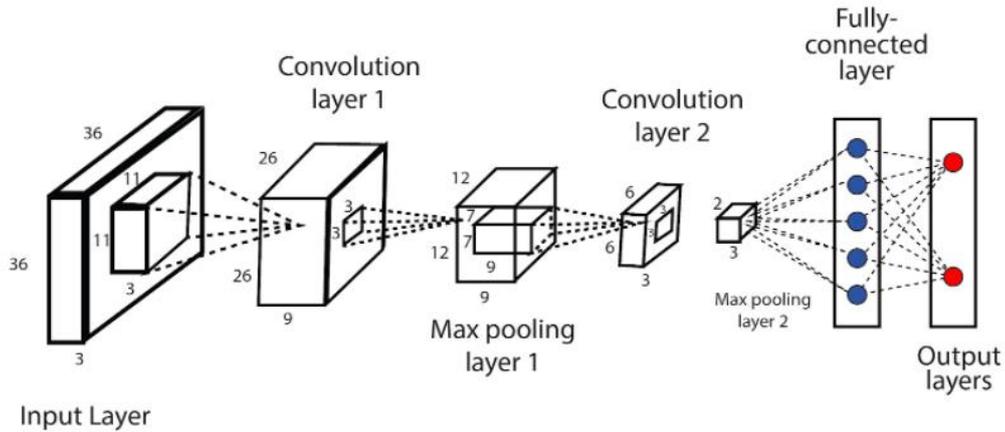


Figure 5. Architecture of Convolutional Neural Network (CNN)

(Balodi, 2021)

3.3.1 Forward Propagation

In forward propagation neural networks is based on mapping function $y = f(x; \theta)$, the model learns the value of θ as a best function approximation. (Schulz and Behnke, 2012)

In forward propagation model, there are no feedback connections. In cases where the forward propagation neural networks are extended to include feedback connections then these are called recurrent neural networks. (Goodfellow, Bengio and Courville, 2016)

3.3.2 Backward Propagation

In backward propagation the information flows from Output layer to Input layer preferably used for corrections or feedback to adjust weights to minimize the loss function. (Seth, 2021)

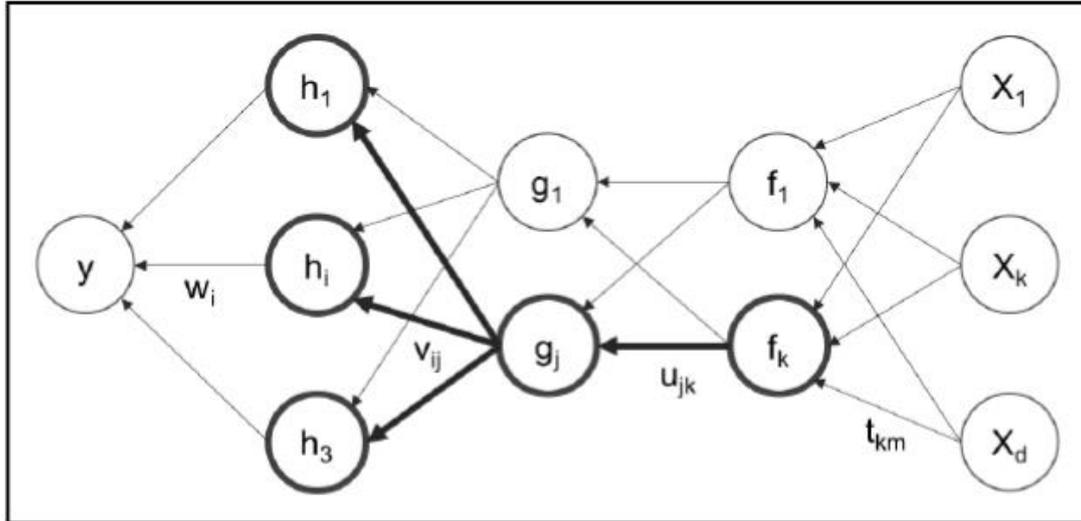


Figure 6. Backward propagation

(Shanmugamani, 2018)

3.4 Semantic Segmentation

Semantic Segmentation approach of computer vision identifies the contents of image for each pixel. It differs from image recognition where entire image is identified.

Some of its primary applications are in autonomous vehicles, human-computer interaction, robotics, and photo editing/creativity tools, where surrounding scene understanding plays a vital role. (Image Segmentation Guide, 2021)

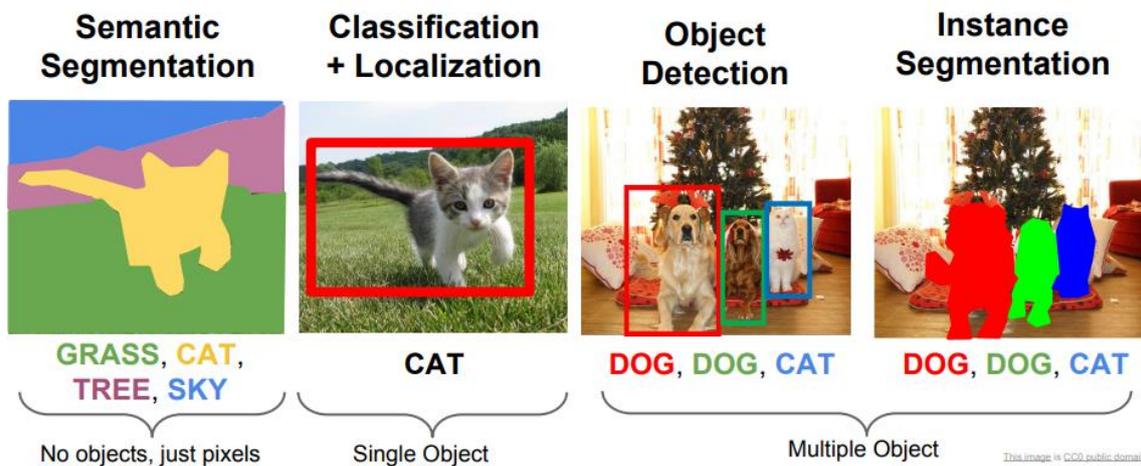


Figure 7. Computer vision tasks

(Li, Johnson & Yeung, 2017)

Semantic Segmentation

Label each pixel in the image with a category label

Don't differentiate instances, only care about pixels

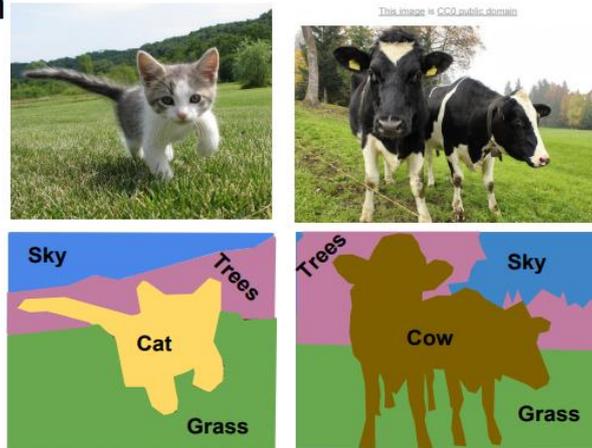


Figure 8. Semantic segmentation example

(Li, Johnson & Yeung, 2017)

Semantic segmentation: Fully convolutional

This has encoder-decoder networks. The encoder module that learns the feature maps and collects semantic information. The decoder module then collects the spatial information. (Chen et al., 2018)

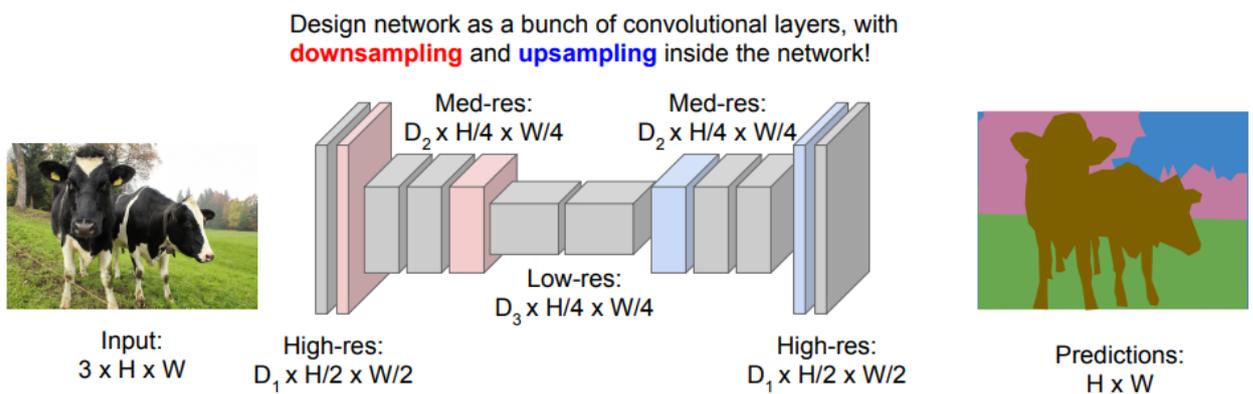


Figure 9. Semantic segmentation: fully convolutional

(Li, Johnson & Yeung, 2017)

3.5 Transfer Learning

Transfer Learning method provides feature representation with pre-trained models such that training effort and time is minimized. Pre-trained models are trained using large datasets like ImageNet, these models can then be tuned for custom applications using case specific image data. This approach reduces the training time and generalization error. Pre-trained models perform poorly when the features learned through the initial training data do not match the problem dataset. Popular pre-trained architectures include VGG family, ResNet, Inception, MobileNet, EfficientNet and DenseNet. These models are trained on ImageNet dataset and can be implemented with frameworks like TensorFlow, Keras, and Pytorch.

ImageNet Dataset – is a large dataset of annotated photographs which are used for computer vision modelling. This dataset has about 14 million images from more than 21000 groups or classes of images and have more than 1 million images that are annotated with bounding box. The ImageNet large scale vision recognition challenge for deep learning required models to be developed to classify these images out into 1000 object categories. (Lendave, 2021)

3.5.1 Commonly used Transfer Learning Models

Inception: Compared to VGGNet, Inception Networks (GoogLeNet/Inception v1) are more computationally efficient, generating fewer number of parameters requiring less memory and other resources. This was proposed in the paper Rethinking the Inception Architecture for Computer Vision, published in 2015, co-authored by Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, and Jonathon Shlens. (Szegedy et al., 2014)

Inception networks need to be optimized to allow easier adaptation, these include factorized convolutions, regularization, dimension reduction, and parallelized computations. (Kurama, 2020)

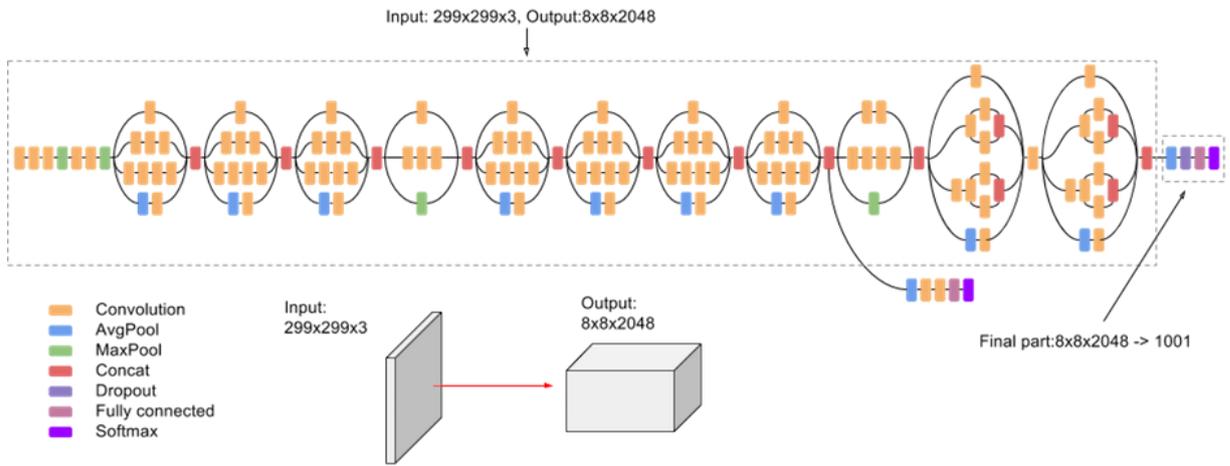


Figure 10. Inception model architecture

(Kurama, 2020)

EfficientNet: This CNN family has been built by Google, it provides better accuracy and efficiency by reducing the number of parameters in comparison to other models. EfficientNet-B0 model is a baseline architecture trained on the ImageNet dataset. EfficientNet is built to provide an effective compound scaling method to achieve maximum accuracy gains. The figure below provides the visualization of scaling possibilities. (Jordan, 2018)

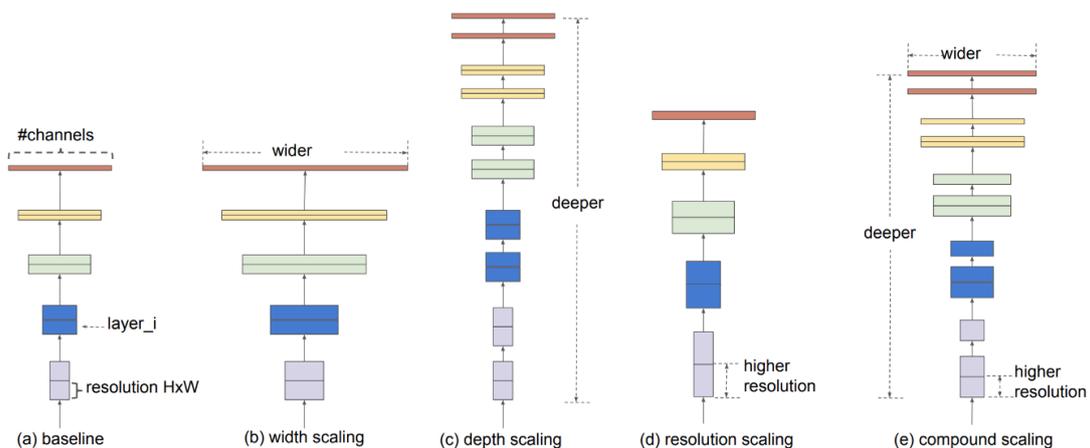


Figure 11. Model Scaling

(Tan and Le, 2019)

(a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) Is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

VGG Family: VGG stands for Visual Geometry Group; it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. The “deep” refers to the number of layers with VGG-16 or VGG-19 consisting of 16 and 19 convolutional layers.

The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNet also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures. (Leonardblier, 2016)

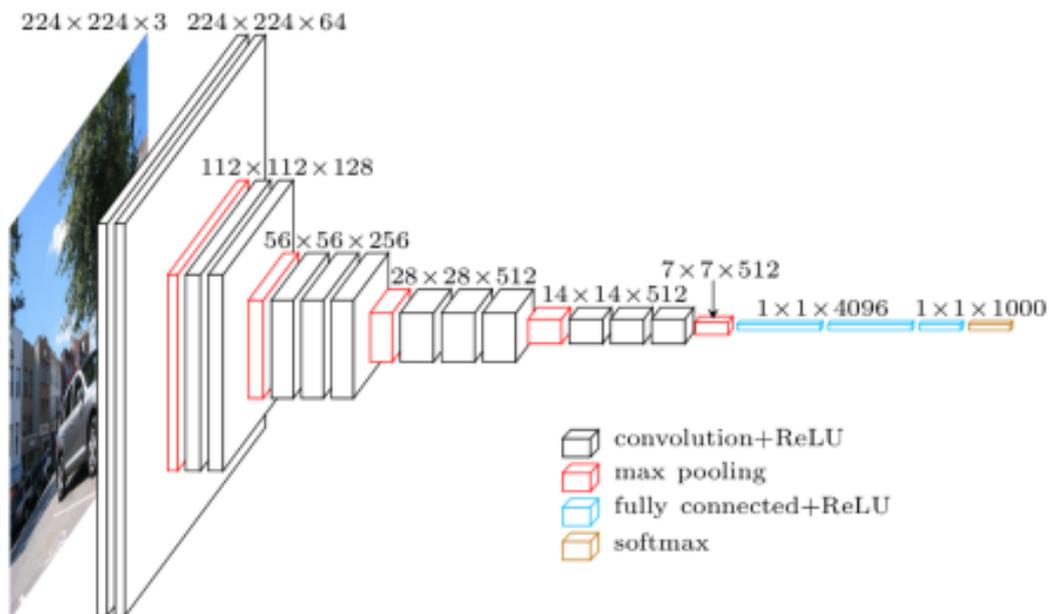


Figure 12. VGG Neural Network Architecture

(Leonardblier, 2016)

ResNet: This model differed from the previous sequential networks, is based on microarchitecture modules or building blocks used to create a new network. This Residual Network model was introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun in paper “Deep Residual Learning for Image Recognition”. In deep learning networks as the number of layers increase there is problem of Vanishing/Exploding gradient, causing the gradient to become 0 or too large, causing increase in error with increase in

MobileNet: This is TensorFlow’s first mobile computer vision model designed for mobile applications.

MobileNet is based on depthwise separable convolutions which reduces the number of parameters in comparison with regular convolutions networks, and so is a lightweight deep neural network. MobileNet is open sourced by Google giving a starting point for training the classifiers. (Pujara, 2020)

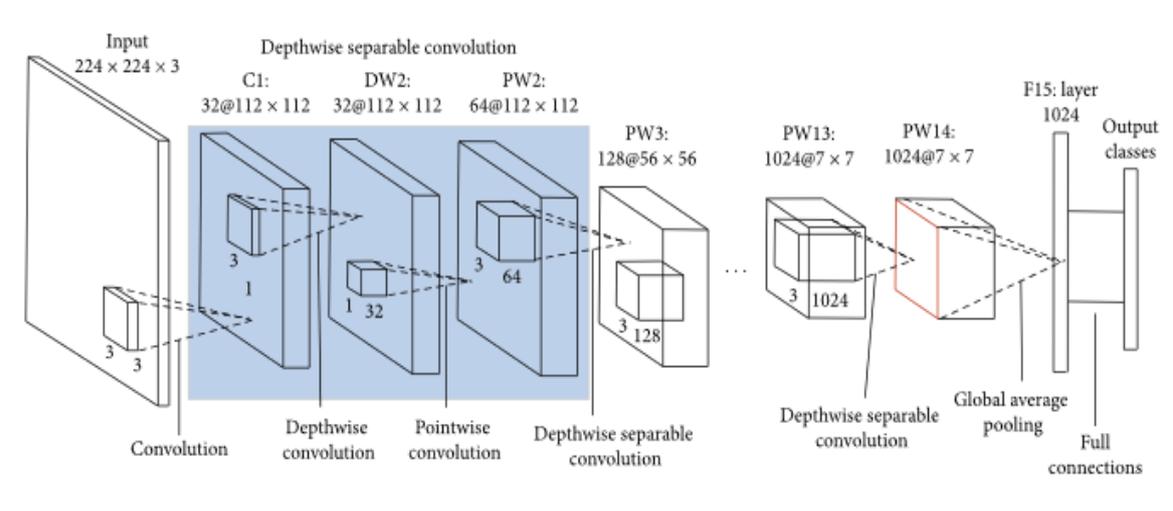


Figure 15. MobileNet architecture

(Pujara, 2020)

DenseNet: This is densely connected-convolutional networks and like a ResNet. They differ as ResNet uses an additive method taking a previous output as an input for a future layer, whereas the DenseNet takes all previous output as an input for a future layer. DenseNet is developed to improve accuracy over vanishing gradient in high-level neural networks. (Huang *et al.*, 2016)

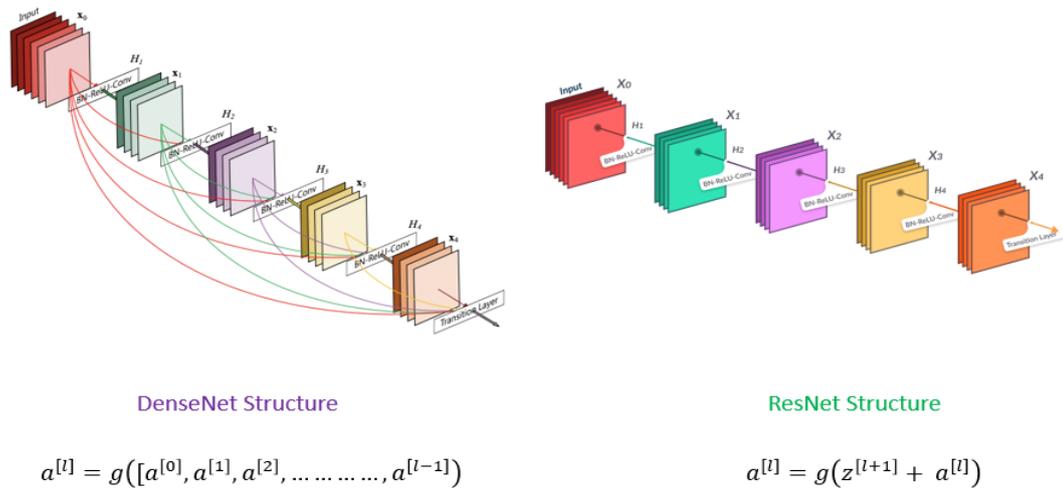


Figure 16. DenseNet Architecture VS ResNet Architecture

(Singhal, 2020)

3.6 Satellite imagery

Satellites with different sensors collect data for remote sensing applications. There are active & passive sensors based on the sensing approach. Passive sensors collect the earth's reflected radiation received from Sun. The Active sensors can emit radiation to earth and collect reflection.

Satellites and the EM Spectrum - Satellites carry sensors that can capture radiation covering broad EM spectrum. These could include human visible light, infrared light, ultraviolet light, or even microwaves. The satellite sensing data which are from non-visible spectrum are then mapped to visible colors to generate visible images. (How does satellite imaging work?, 2021)

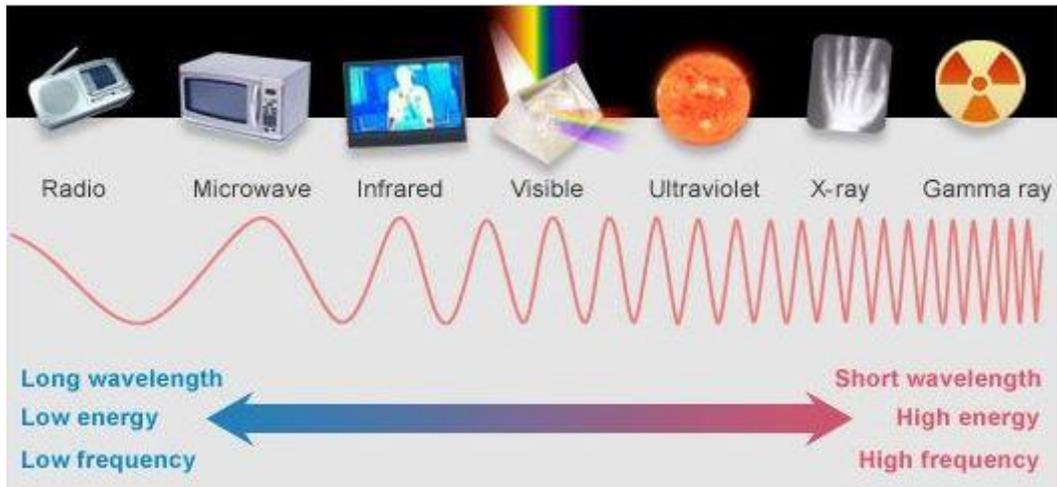


Figure 17. The EM Spectrum

(How does satellite imaging work?, 2021)

3.7 Related Work

The use of satellite images to detect ships has been studied as both segmentation as well as object detection problem statement. The CNN-based SAR ship detection can be divided into region-based methods and regression-based methods. a) In region-based detection process there are two stages. The first stage generates a set of sparse candidate proposals, then classifying these proposals into different categories. Few variants for this process are Faster RCNN, RFCN, Cascade RCNN. b) In regression-based approach, feature pyramids and multiscale prediction structure are used to improve accuracy and speed. It predicts the location and categories of the objects. (Han et al., 2020)

Han *et al.* in their paper "Multi-Size Convolution and Learning Deep Network for SAR Ship Detection from Scratch" provide a SAR ship detector from scratch. The experiments show the design to have good accuracy, speed and robustness. (Han et al., 2020)

Yuan Yao *et al.* in his paper "High-resolution optical satellite image simulation of ship target in large sea scenes" lists the usual image vulnerabilities and simulates the image sensing issues due to different sea states, cloud conditions, target types and imaging conditions which can support the evaluation of ship detection algorithms. (Yao, Jiang and Zhang, 2016)

There has been a lot of research concerning the segmentation of ships from image and detection of ships from image.

4 METHODS AND MATERIAL

The programming language used for preprocessing the data and developing/training model was Python. Commonly used packages such as Numpy, Pandas were used. Also, deep learning models were developed using Keras API with TensorFlow backend. All the experiments were conducted on the CSC's puhti computing GPU partition with NVIDIA Volta V100.

4.1 Dataset

A public dataset provided on the Airbus Ship Detection Challenge website (Airbus ship detection challenge, 2018) is used in this thesis. The data set contains more than 100K images of size 768*768 with a total size exceeding 30Gb. Many images do not contain ships, and those that do may contain multiple ships. Ships within and across images may differ in size (sometimes significantly) and be located in open sea, at docks, marinas, etc. There is actually quite imbalance in the sense that only $\approx 1/4$ of the data images have ships. Along with the images is a CSV file that lists all the images' ids and their corresponding pixels coordinates. These coordinates represent segmentation boxes of ships. Not having pixel coordinates means the image doesn't have any ships.

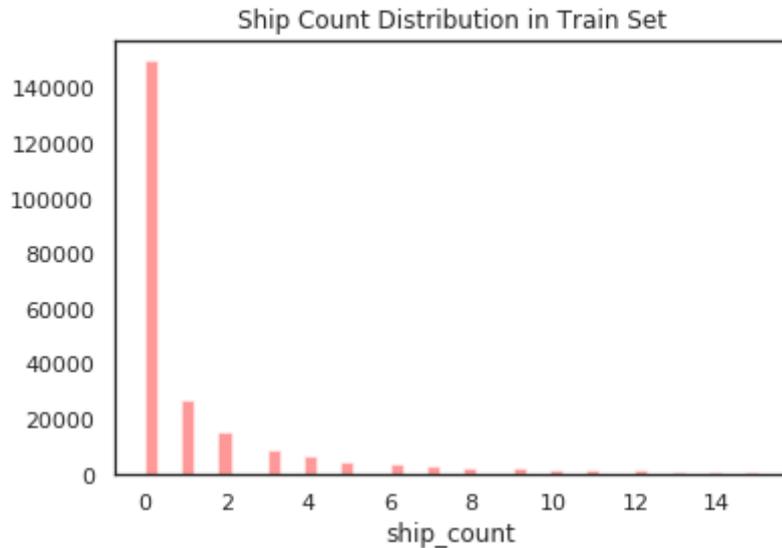


Figure 18. Ship counts in data

The challenge of detecting the ships in the images can be thought as a classification problem for pixels, where, for each image, we need to classify 768×768 pixels in one of two classes: ship and no-ships. Below graph shows the imbalance of classes considering the pixel level granularity, only 0.001% of the pixels are ships, while 99.9% of the pixels are no-ships. And dropping all the images with no ships in them the class imbalance is reduced, but it's still very high: this is, 0.5% of the pixels are ships while 99.5% are no-ships.

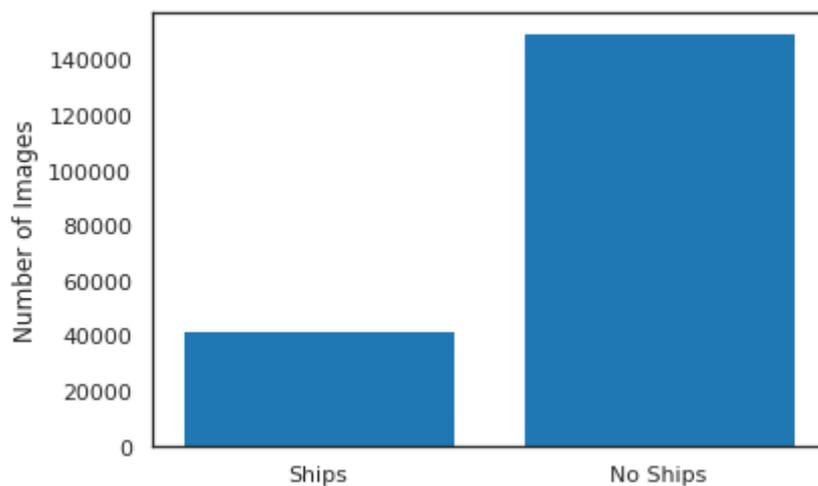


Figure 19. Image count with valid ships

TRAIN SHIP SEGMENTATION (CSV):

	ImageId	EncodedPixels
IMAGE - NO SHIPS	0003e153.jpg	
	0001124c7.jpg	
IMAGE - ONE SHIP	000155de5.jpg	264661 17 265429 33 266197 33 266965 33 267733 33 268501 33 269269 33 270037 33 270805 33 271573 33 272341 33 273109 33 273877 33 274645 33 275413 33 276181 33 276949 33 277717 33 278485 33 279253 33 280021 33 280789 33 281557 33 282325 33 283093 33 283861 33 284629 33 285397 33 286165 33 286933 33 287701 33 288469 33 289237 33 290005 33 290773 33 291541 33 292309 33 293077 33 293845 33 294613 33 295381 33 296149 33 296917 33 297685 33 298453 33 299221 33 299989 33 300757 33 301525 33 302293 33 303061 33 303829 33 304597 33 305365 33 306133 33 306901 33 307669 33 308437 33 309205 33 309973 33 310741 33 311509 33 312277 33 313045 33 313813 33 314581 33 315349 33 316117 33 316885 33 317653 33 318421 33 319189 33 319957 33 320725 33 321493 33 322261 33 323029 33 323797 33 324565 33 325333 33 326101 33 326869 33 327637 33 328405 33 329173 33 329941 33 330709 33 331477 33 332245 33 333013 33 333781 33 334549 33 335317 33 336085 33 336853 33 337621 33 338389 33 339157 33 339925 33 340693 33 341461 33 342229 33 342997 33 343765 33 344533 33 345301 33 346069 33 346837 33 347605 33 348373 33 349141 33 349909 33 350677 33 351445 33 352213 33 352981 33 353749 33 354517 33 355285 33 356053 33 356821 33 357589 33 358357 33 359125 33 359893 33 360661 33 361429 33 362197 33 362965 33 363733 33 364501 33 365269 33 366037 33 366805 33 367573 33 368341 33 369109 33 369877 33 370645 33 371413 33 372181 33 372949 33 373717 33 374485 33 375253 33 376021 33 376789 33 377557 33 378325 33 379093 33 379861 33 380629 33 381397 33 382165 33 382933 33 383701 33 384469 33 385237 33 386005 33 386773 33 387541 33 388309 33 389077 33 389845 33 390613 33 391381 33 392149 33 392917 33 393685 33 394453 33 395221 33 395989 33 396757 33 397525 33 398293 33 399061 33 399829 33 400597 33 401365 33 402133 33 402901 33 403669 33 404437 33 405205 33 405973 33 406741 33 407509 33 408277 33 409045 33 409813 33 410581 33 411349 33 412117 33 412885 33 413653 33 414421 33 415189 33 415957 33 416725 33 417493 33 418261 33 419029 33 419797 33 420565 33 421333 33 422101 33 422869 33 423637 33 424405 33 425173 33 425941 33 426709 33 427477 33 428245 33 429013 33 429781 33 430549 33 431317 33 432085 33 432853 33 433621 33 434389 33 435157 33 435925 33 436693 33 437461 33 438229 33 438997 33 439765 33 440533 33 441301 33 442069 33 442837 33 443605 33 444373 33 445141 33 445909 33 446677 33 447445 33 448213 33 448981 33 449749 33 450517 33 451285 33 452053 33 452821 33 453589 33 454357 33 455125 33 455893 33 456661 33 457429 33 458197 33 458965 33 459733 33 460501 33 461269 33 462037 33 462805 33 463573 33 464341 33 465109 33 465877 33 466645 33 467413 33 468181 33 468949 33 469717 33 470485 33 471253 33 472021 33 472789 33 473557 33 474325 33 475093 33 475861 33 476629 33 477397 33 478165 33 478933 33 479701 33 480469 33 481237 33 482005 33 482773 33 483541 33 484309 33 485077 33 485845 33 486613 33 487381 33 488149 33 488917 33 489685 33 490453 33 491221 33 491989 33 492757 33 493525 33 494293 33 495061 33 495829 33 496597 33 497365 33 498133 33 498901 33 499669 33 500437 33 501205 33 501973 33 502741 33 503509 33 504277 33 505045 33 505813 33 506581 33 507349 33 508117 33 508885 33 509653 33 510421 33 511189 33 511957 33 512725 33 513493 33 514261 33 515029 33 515797 33 516565 33 517333 33 518101 33 518869 33 519637 33 520405 33 521173 33 521941 33 522709 33 523477 33 524245 33 525013 33 525781 33 526549 33 527317 33 528085 33 528853 33 529621 33 530389 33 531157 33 531925 33 532693 33 533461 33 534229 33 534997 33 535765 33 536533 33 537301 33 538069 33 538837 33 539605 33 540373 33 541141 33 541909 33 542677 33 543445 33 544213 33 544981 33 545749 33 546517 33 547285 33 548053 33 548821 33 549589 33 550357 33 551125 33 551893 33 552661 33 553429 33 554197 33 554965 33 555733 33 556501 33 557269 33 558037 33 558805 33 559573 33 560341 33 561109 33 561877 33 562645 33 563413 33 564181 33 564949 33 565717 33 566485 33 567253 33 568021 33 568789 33 569557 33 570325 33 571093 33 571861 33 572629 33 573397 33 574165 33 574933 33 575701 33 576469 33 577237 33 578005 33 578773 33 579541 33 580309 33 581077 33 581845 33 582613 33 583381 33 584149 33 584917 33 585685 33 586453 33 587221 33 587989 33 588757 33 589525 33 590293 33 591061 33 591829 33 592597 33 593365 33 594133 33 594901 33 595669 33 596437 33 597205 33 597973 33 598741 33 599509 33 600277 33 601045 33 601813 33 602581 33 603349 33 604117 33 604885 33 605653 33 606421 33 607189 33 607957 33 608725 33 609493 33 610261 33 611029 33 611797 33 612565 33 613333 33 614101 33 614869 33 615637 33 616405 33 617173 33 617941 33 618709 33 619477 33 620245 33 621013 33 621781 33 622549 33 623317 33 624085 33 624853 33 625621 33 626389 33 627157 33 627925 33 628693 33 629461 33 630229 33 630997 33 631765 33 632533 33 633301 33 634069 33 634837 33 635605 33 636373 33 637141 33 637909 33 638677 33 639445 33 640213 33 640981 33 641749 33 642517 33 643285 33 644053 33 644821 33 645589 33 646357 33 647125 33 647893 33 648661 33 649429 33 650197 33 650965 33 651733 33 652501 33 653269 33 654037 33 654805 33 655573 33 656341 33 657109 33 657877 33 658645 33 659413 33 660181 33 660949 33 661717 33 662485 33 663253 33 664021 33 664789 33 665557 33 666325 33 667093 33 667861 33 668629 33 669397 33 670165 33 670933 33 671701 33 672469 33 673237 33 674005 33 674773 33 675541 33 676309 33 677077 33 677845 33 678613 33 679381 33 680149 33 680917 33 681685 33 682453 33 683221 33 683989 33 684757 33 685525 33 686293 33 687061 33 687829 33 688597 33 689365 33 690133 33 690901 33 691669 33 692437 33 693205 33 693973 33 694741 33 695509 33 696277 33 697045 33 697813 33 698581 33 699349 33 700117 33 700885 33 701653 33 702421 33 703189 33 703957 33 704725 33 705493 33 706261 33 707029 33 707797 33 708565 33 709333 33 710101 33 710869 33 711637 33 712405 33 713173 33 713941 33 714709 33 715477 33 716245 33 717013 33 717781 33 718549 33 719317 33 720085 33 720853 33 721621 33 722389 33 723157 33 723925 33 724693 33 725461 33 726229 33 726997 33 727765 33 728533 33 729301 33 730069 33 730837 33 731605 33 732373 33 733141 33 733909 33 734677 33 735445 33 736213 33 736981 33 737749 33 738517 33 739285 33 740053 33 740821 33 741589 33 742357 33 743125 33 743893 33 744661 33 745429 33 746197 33 746965 33 747733 33 748501 33 749269 33 750037 33 750805 33 751573 33 752341 33 753109 33 753877 33 754645 33 755413 33 756181 33 756949 33 757717 33 758485 33 759253 33 760021 33 760789 33 761557 33 762325 33 763093 33 763861 33 764629 33 765397 33 766165 33 766933 33 767701 33 768469 33 769237 33 770005 33 770773 33 771541 33 772309 33 773077 33 773845 33 774613 33 775381 33 776149 33 776917 33 777685 33 778453 33 779221 33 779989 33 780757 33 781525 33 782293 33 783061 33 783829 33 784597 33 785365 33 786133 33 786901 33 787669 33 788437 33 789205 33 789973 33 790741 33 791509 33 792277 33 793045 33 793813 33 794581 33 795349 33 796117 33 796885 33 797653 33 798421 33 799189 33 799957 33 800725 33 801493 33 802261 33 803029 33 803797 33 804565 33 805333 33 806101 33 806869 33 807637 33 808405 33 809173 33 809941 33 810709 33 811477 33 812245 33 813013 33 813781 33 814549 33 815317 33 816085 33 816853 33 817621 33 818429 33 819197 33 819965 33 820733 33 821501 33 822269 33 823037 33 823805 33 824573 33 825341 33 826109 33 826877 33 827645 33 828413 33 829181 33 829949 33 830717 33 831485 33 832253 33 833021 33 833789 33 834557 33 835325 33 836093 33 836861 33 837629 33 838397 33 839165 33 839933 33 840701 33 841469 33 842237 33 843005 33 843773 33 844541 33 845309 33 846077 33 846845 33 847613 33 848381 33 849149 33 849917 33 850685 33 851453 33 852221 33 852989 33 853757 33 854525 33 855293 33 856061 33 856829 33 857597 33 858365 33 859133 33 859901 33 860669 33 861437 33 862205 33 862973 33 863741 33 864509 33 865277 33 866045 33 866813 33 867581 33 868349 33 869117 33 869885 33 870653 33 871421 33 872189 33 872957 33 873725 33 874493 33 875261 33 876029 33 876797 33 877565 33 878333 33 879101 33 879869 33 880637 33 881405 33 882173 33 882941 33 883709 33 884477 33 885245 33 886013 33 886781 33 887549 33 888317 33 889085 33 889853 33 890621 33 891389 33 892157 33 892925 33 893693 33 894461 33 895229 33 895997 33 896765 33 897533 33 898301 33 899069 33 899837 33 900605 33 901373 33 902141 33 902909 33 903677 33 904445 33 905213 33 905981 33 906749 33 907517 33 908285 33 909053 33 909821 33 910589 33 911357 33 912125 33 912893 33 913661 33 914429 33 915197 33 915965 33 916733 33 917501 33 918269 33 919037 33 919805 33 920573 33 921341 33 922109 33 922877 33 923645 33 924413 33 925181 33 925949 33 926717 33 927485 33 928253 33 929021 33 929789 33 930557 33 931325 33 932093 33 932861 33 933629 33 934397 33 935165 33 935933 33 936701 33 937469 33 938237 33 939005 33 939773 33 940541 33 941309 33 942077 33 942845 33 943613 33 944381 33 945149 33 945917 33 946685 33 947453 33 948221 33 948989 33 949757 33 950525 33 951293 33 952061 33 952829 33 953597 33 954365 33 955133 33 955901 33 956669 33 957437 33 958205 33 958973 33 959741 33 960509 33 961277 33 962045 33 962813 33 963581 33 964349 33 965117 33 965885 33 966653 33 967421 33 968189 33 968957 33 969725 33 970493 33 971261 33 972029 33 972797 33 973565 33 974333 33 975101 33 975869 33 976637 33 977405 33 978173 33 978941 33 979709 33 980477 33 981245 33 982013 33 982781 33 983549 33 984317 33 985085 33 985853 33 986621 33 987389 33 988157 33 988925 33 989693 33 990461 33 991229 33 991997 33 992765 33 993533 33 994301 33 995069 33 995837 33 996605 33 997373 33 998141 33 998909 33 999677 33 1000000
	000194a2d.jpg	360486 1 361252 4 362019 5 362785 8 363552 10 364321 10 365090 9 365858 10 366627 10 367396 9 368165 9 368933 10 369702 10 370471 9 371240 9 372009 9 372777 10 373546 9 374315 9 375084 9 375853 9 376622 9 377391 9 378160 9 378929 9 379698 9 380467 9 381236 9 382005 9 382774 9 383543 9 384312 9 385081 9 385850 9 386619 9 387388 9 388157 9 388926 9 389695 9 390464 9 391233 9 392002 9 392771 9 393540 9 394309 9 395078 9 395847 9 396616 9 397385 9 398154 9 398923 9 399692 9 400461 9 401230 9 401999 9 402768 9 403537 9 404306 9 405075 9 405844 9 406613 9 407382 9 408151 9 408920 9 409689 9 410458 9 411227 9 411996 9 412765 9 413534 9 414303 9 415072 9 415841 9 416610 9 417379 9 418148 9 418917 9 419686 9 420455 9 421224 9 421993 9 422762 9 423531 9 424300 9 425069 9 425838 9 426607 9 427376 9 428145 9 428914 9 429683 9 430452 9 431221 9 431990 9 432759 9 433528 9 434297 9 435066 9 435835 9 436604 9 437373 9 438142 9 438911 9 439680 9 440449 9 441218 9 441987 9 442756 9 443525 9 444294 9 445063 9 445832 9 446601 9 447370 9 448139 9 448908 9 449677 9 450446 9 451215 9 451984 9 452753 9 453522 9 454291 9 455060 9 455829 9 456598 9 457367 9 458136 9 458905 9 459674 9 460443 9 461212 9 461981 9 462750 9 463519 9 464288 9 465057 9 465826 9 466595 9 467364 9 468133 9 468902 9 469671 9 470440 9 471209 9 471978 9 472747 9 473516 9 474285 9 475054 9 475823 9 476592 9 477361 9 478130 9 478899 9 479668 9 480437 9 481206 9 481975 9 482744 9 483513 9 484282 9 485051 9 485820 9 486589 9 487358 9 488127 9 488896 9 489665 9 490434 9 491203 9 491972 9 492741 9 493510 9 494279 9 495048 9 495817 9 496586 9 497355 9 498124 9 498893 9 499662 9 500431 9 501200 9 501969 9 502738 9 503507 9 504276 9 505045 9 505814 9 506583 9 507352 9 508121 9 508890 9 509659 9 510428 9 511197 9 511966 9 512735 9 513504 9 514273 9 515042 9 515811 9 516580 9 517349 9 518118 9 518887 9 519656 9 520425 9 521194 9 521963 9 522732 9 523501 9 524270 9 525039 9 525808 9 526577 9 527346 9 528115 9 528884 9 529653 9 530422 9 531191 9 531960 9 532729 9 533498 9 534267 9 535036 9 535805 9 536574 9 537343 9 538112 9 538881 9 539650 9 540419 9 541188 9 541957 9 542726 9 543495 9 544264 9 545033 9 545802 9 546571 9 547340 9 548109 9 548878 9 549647 9 550416 9 551185 9 551954 9 552723 9 553492 9 554261 9 555030 9 555799 9 556568 9 557337 9 558106 9 558875 9 559644 9 560413 9 561182 9 561951 9 562720 9 563489 9 564258 9 565027 9 565796 9 566565 9 567334 9 568103 9 568872 9 569641 9 570410 9 571179 9 571948 9 572717 9 573486 9 574255 9 575024 9 575793 9

In the below Figure we can see the image and the masks for ships in the image and overlay.

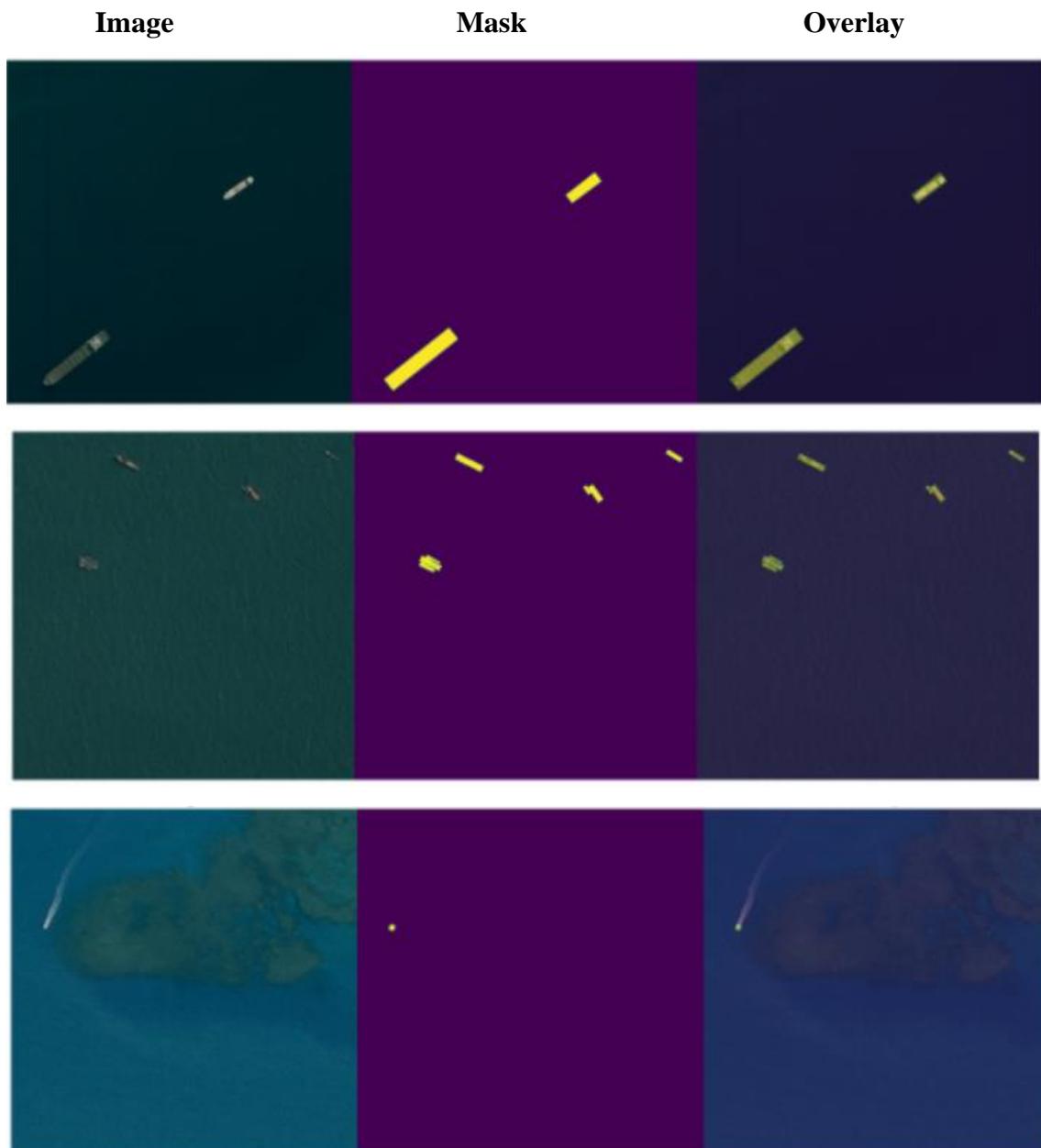


Figure 21. Images with masks

4.2 U-Net: Convolutional Networks for Image Segmentation

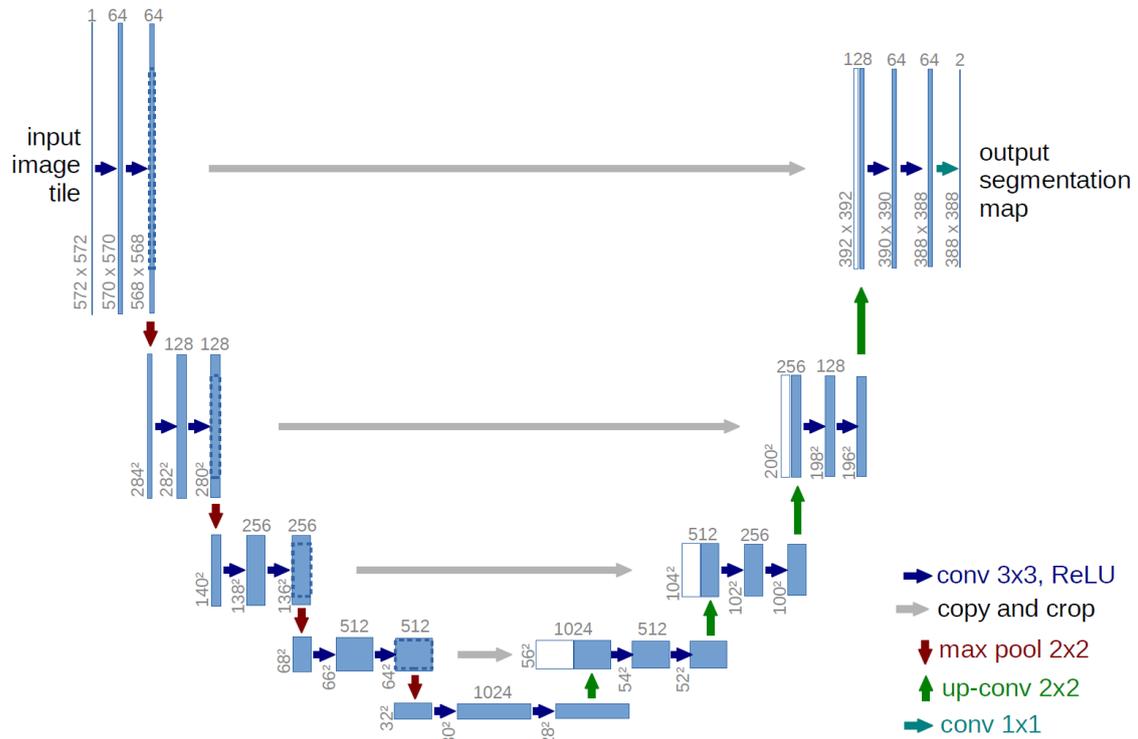


Figure 22. U-Net architecture - Example for 32x32 pixels in the lowest resolution

(Ronneberger, Fischer and Brox, 2015)

4.2.1 U-Net Network Architecture

U-net is an encoder-decoder network architecture which is U-shaped. Four encoder blocks and four decoder blocks are connected via a bridge. The encoder network halves the spatial dimensions and doubles the encoder block filters. And the decoder network doubles the spatial dimensions and halves the number of feature channels. (Tomar, 2021b)

4.2.2 Encoder Network

The encoder network extracts the features and learns the abstract representation of the input image through the encoder blocks. Each encoder block consists of 3*3 convolutions which are followed by ReLU (Rectified Linear Unit) activation function. The output of the activation function ReLU acts as a skip connection for the decoder block. (Tomar, 2021b)

Next, follows a 2x2 max pooling, where the spatial dimensions (height and width) of the feature maps are reduced by half. This reduces the computational cost by decreasing the number of trainable parameters. (Tomar, 2021b)

4.2.3 Skip Connections

The skip connections help the decoder to generate better semantic features. Skip connections helps to learn better representation by helping in better flow of gradient while back propagation. (Tomar, 2021b)

4.2.4 Bridge

The bridge completes the flow of information by connecting encoder and decoder networks. It consists of 3*3 convolutions, which is followed by ReLU activation function. (Tomar, 2021b)

4.2.5 Decoder Network

The decoder network takes the abstract representation and generates the segmentation masks. The decoder block starts with 2*2 transpose convolution. Then it concatenates with the skip connection feature map from the encoder block. the skip connections provide features from earlier layers which get lost sometimes due to the depth of the network. Then two 3*3 convolutions are used, which are followed by activation function ReLU. The last decoder output passes through 1*1 convolution with sigmoid activation function which gives pixel-wise classification of segmentation masks. (Tomar, 2021b)

4.3 Backbone of Network

The backbone defines how these layers are arranged in the encoder network and how decoder network should be built. The backbones used in this thesis are VGG, ResNet, Inception, DenseNet, EfficientNet and MobileNet. These CNNs perform encoding and

down sampling itself and their counter parts are built to perform decoding and up sampling to form a final U-net model. (Alsabhan and Alotaiby, 2022)

4.4 Evaluation Metrics

The task of semantic segmentation is simply to predict the class of each pixel in an image. The prediction output shape matches the input's spatial resolution (width and height) with a channel depth equivalent to the number of possible classes to be predicted. Each channel consists of a binary mask which labels areas where a specific class is present.

4.4.1 Intersection over Union

Intersection over Union (IoU) metric is also known as Jaccard Index and closely related to Dice coefficient which is used as loss function during training.

The IoU metrics is measured by the number of common pixels in target and prediction masks divided by the total number of pixels present in both target and prediction masks. (Jordan, 2018)

$$IoU = \frac{A \cap B}{A \cup B}$$

4.4.2 F1 Score

F1 score, also known as Dice coefficient, is doubled the area of overlap between the prediction and ground truth divided by the total number of pixels in both images. This metric is very similar to IoU, and the results of these metrics are positively correlated. (Jordan, 2018)

$$F1 = \frac{2||A \cap B||}{||A|| + ||B||}$$

4.4.3 Precision

To evaluate the predicted mask, we have to compare each predicted mask with the target mask for the given input image.

- A **true positive** is observed when IoU score of the prediction-target pair exceeds the predefined value.
- A **false positive** indicates a predicted object mask has no corresponding ground truth object mask.
- A **false negative** indicates that there is no predicted object mask for the ground truth object mask. (Jordan, 2018)



Figure 23. Precision prediction possibilities

(Jordan, 2018)

$$Precision = \frac{TP}{TP + FP}$$

4.4.4 Recall

Recall describes the completeness of our positive predictions relative to the ground truth. (Jordan, 2018)

$$Recall = \frac{TP}{TP + FN}$$

5 RESULTS AND CONCLUSIONS

Experiments were performed using U-Net with different backbones using Keras to develop and train the models. (Iakubovskii, 2020) The backbones used in this thesis are ResNet50, DenseNet-121, InceptionV3, VGG-16, MobileNetV2 and EfficientB2. The dataset has 193K images and every image has a resolution of 768x768. All the models were trained using 768*768 resolution images.

Experiments are performed on Keras models. Keras models are trained with early stopping callback being monitored on validation loss. All the encoders used in the experiments are pre-trained models on Imagenet. The Keras models were initially trained using BCE_jaccard loss plus dice loss for 15 epochs using images with ships. Models were trained by freezing the encoder and only training the decoder.

5.1 ResNet-50 as Backbone

ResNet-50 is a convolutional Neural Network that is 50 layers deep. U-Net with ResNet-50 as backbone was trained for 15 epochs for images with ships. Below graph shows the training vs. validation metrics for 15 epochs.

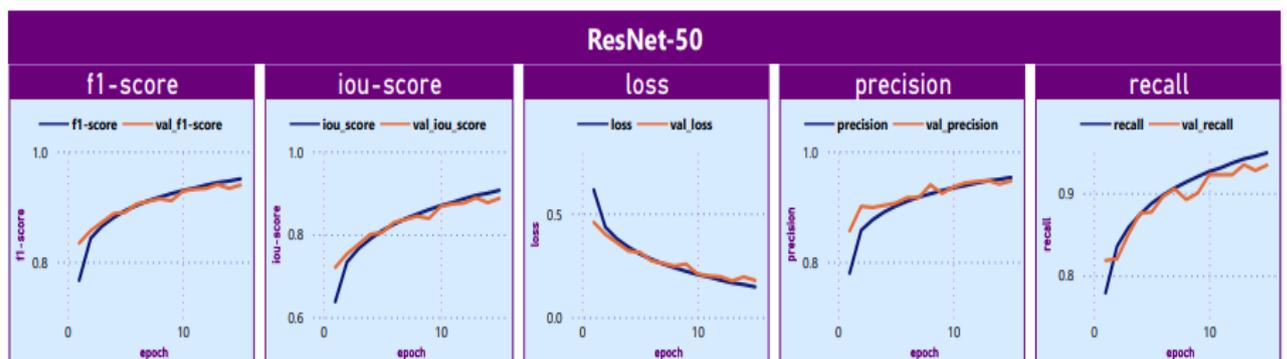


Figure 24. ResNet-50 metrics, 1-15 epoch

5.2 DenseNet-121 as Backbone

U-Net with DenseNet-121 as backbone was trained for 15 epochs and the training vs. validation metrics were plotted as below

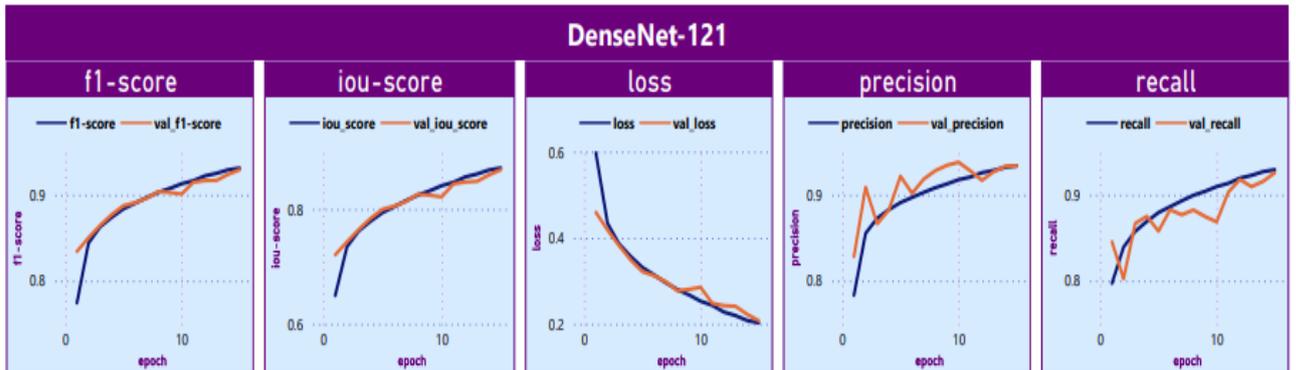


Figure 25. DenseNet-121 metrics, 1-15 epoch

5.3 InceptionV3 as Backbone

U-Net with InceptionV3 as backbone was trained for 15 epochs and the training vs. validation metrics were plotted as below

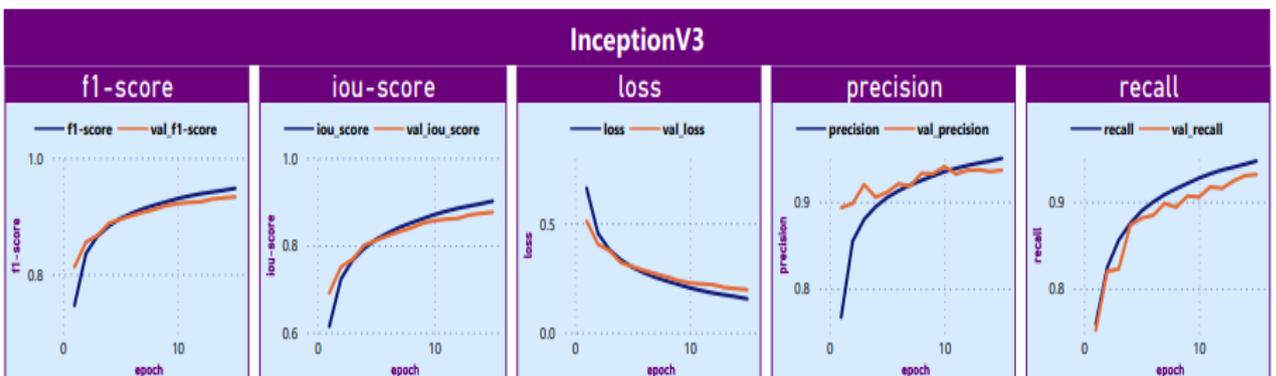


Figure 26. InceptionV3 metrics, 1-15 epoch

5.4 VGG-16 as Backbone

U-Net with VGG-16 as backbone was trained for 15 epochs and the training vs. validation metrics were plotted as below

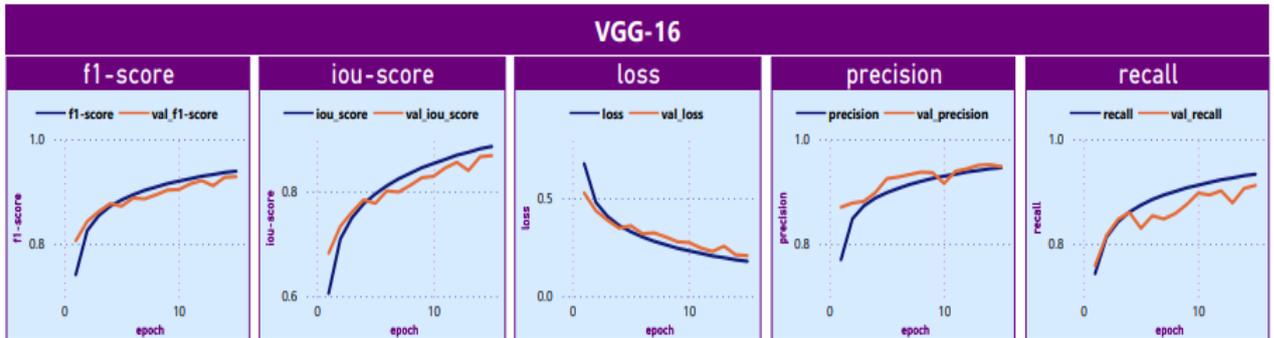


Figure 27. VGG-16 metrics, 1-15 epoch

5.5 MobileNetV2 as Backbone

U-Net with MobileNetV2 as backbone was trained for 15 epochs and the training vs. validation metrics were plotted as below

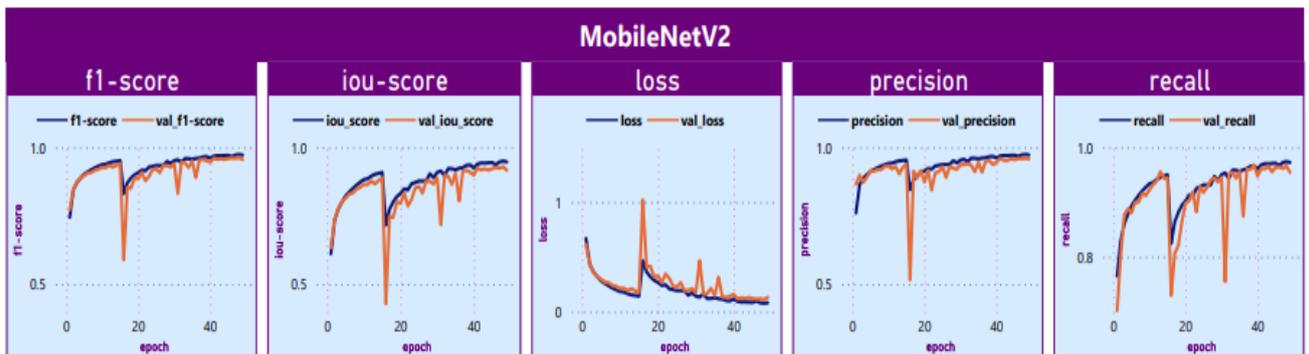


Figure 28. MobileNetV2 metrics, 1-15 epoch

5.6 EfficientNetB2 as Backbone

U-Net with EfficientNetB2 as backbone was trained for 15 epochs and the training vs. validation metrics were plotted as below

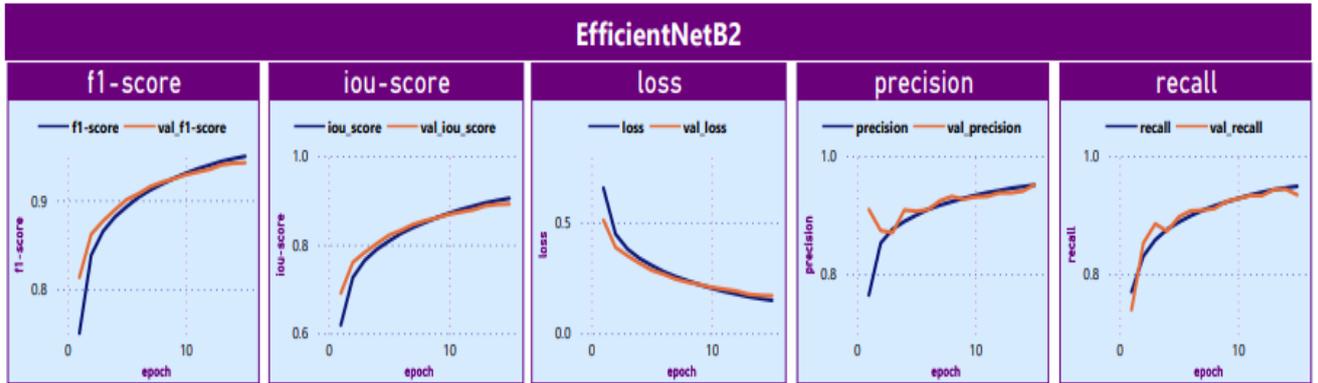


Figure 29. EfficientNetB2 metrics, 1-15 epoch

5.7 Model Test Result

U-Net with different backbones were trained for 15 epochs and the models were tested for the test data. As shown in the table below, F1 score, IoU score, loss, precision and recall were compared. The IoU scores were 0.8676 and 0.8681 and F1 scores were 0.9282 and 0.9280 for DenseNet-121 and VGG-16 respectively, which were the least among other backbones.

Test scores for U-Net+Backbone - for test data					
U-Net+Backbone	f1-score	iou_score	loss	precision	recall
Densenet-121	0.9282	0.8676	0.2104	0.9337	0.9242
EfficientNetB2	0.9431	0.8936	0.1681	0.9520	0.9354
InceptionV3	0.9317	0.8739	0.2005	0.9356	0.9294
MobileNetV2	0.9415	0.8902	0.1740	0.9393	0.9443
ResNet-50	0.9393	0.8869	0.1793	0.9469	0.9330
VGG-16	0.9280	0.8681	0.2106	0.9472	0.9117

Table 1. Comparing model metrics.

As shown in the above table MobileNetV2 and EfficientNetB2 outperformed other backbones with f1 scores 0.9415 and 0.9431 and IoU scores 0.8902 and 0.8936 respectively.

Below plot compares the F1 scores, IoU scores, loss, precision and recall for the U-Net with the backbone - DenseNet-121, EfficientNetB2, InceptionV3, MobileNetV2, ResNet-50 and VGG-16.

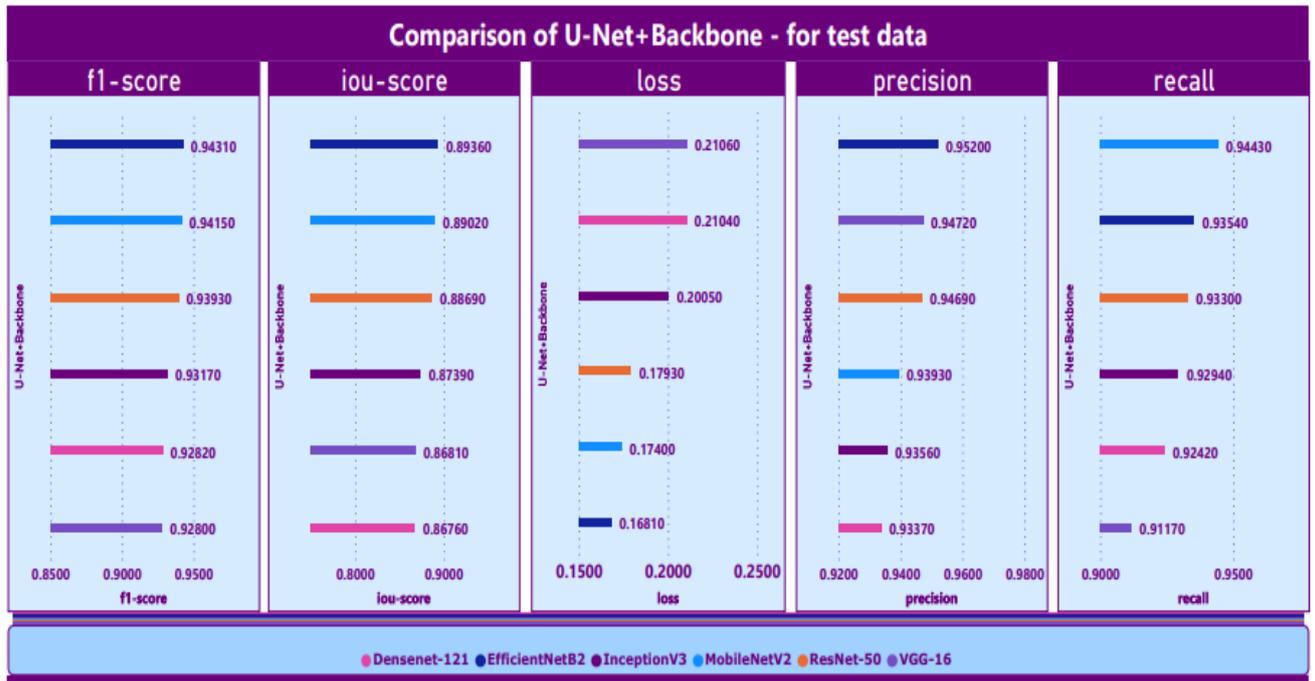


Figure 30. Comparison Metrics, at epoch 15

As seen in Table 1 U-Net with backbone MobileNetV2 and EfficientNetB2 performed well among other backbones when trained for 15 epochs by freezing the encoder and training only the decoder for images with ships. Further MobileNetV2 and EfficientNetB2 were trained for another 34 epochs (total 49 epochs) without freezing the encoder.

5.7.1 U-Net+MobileNetV2

After training for 49 epochs the metrics is plotted as shown in the below graph. Since we trained for the full model with encoder and decoder we can see in the below graphs how f1 score and IoU score goes down from 16th epoch and gradually increases.

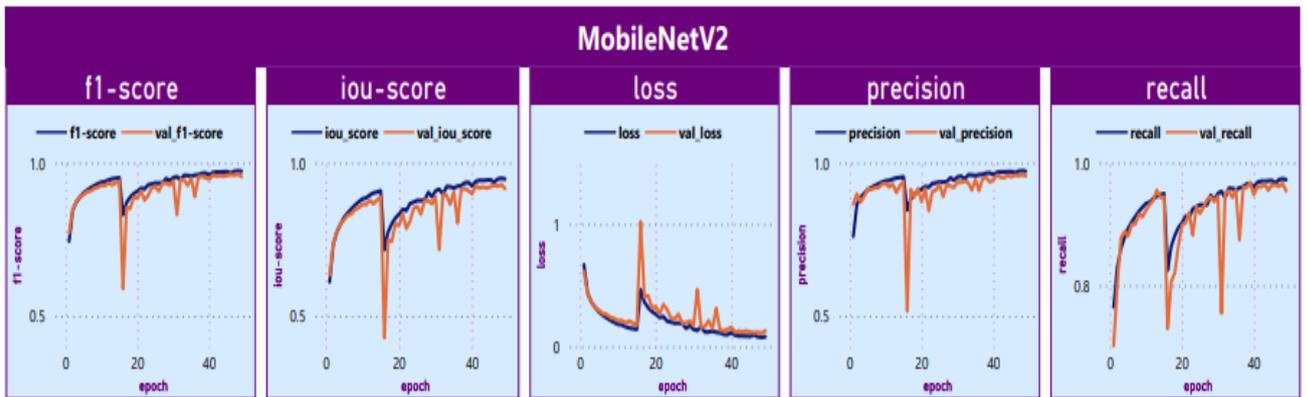


Figure 31. MobileNetV2 metrics. 1-49 epochs

5.7.2 U-Net+EfficientNetB2

U-Net with EfficientNetB2 is also further trained for 34 epochs with encoder and decoder and the metrics for training vs. validation is plotted below for the 49 epochs. Here also we can see F1 and IoU scores drop from 16th epoch and gradually increases.

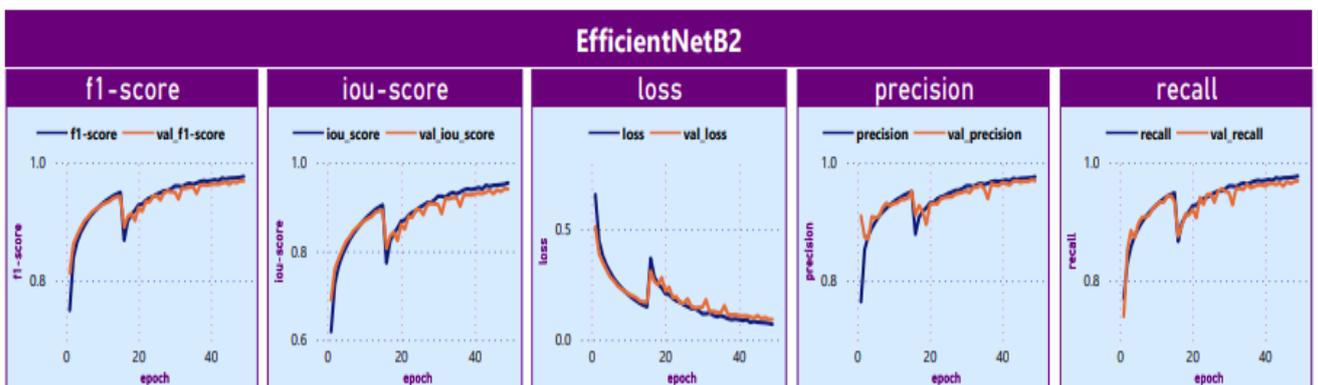


Figure 32. EfficientNetB2 metrics, 1-49 epochs

5.7.3 Comparison of U-Net+EfficientNetB2 and U-Net+MobileNetV2

Below table shows the model test results for U-Net+mobileNetV2 and U-Net+EfficientB2 after training for 49 epochs. The F1 score for U-Net+EfficientNetB2 is 0.9682 and IoU score is 0.9392 which performed better than MobileNetV2. The F1 score for MobileNetV2 is 0.9562 and IoU score is 0.9167.

Test scores for U-Net+Backbone - MobileNetV2 & EfficientNetB2					
U-Net+Backbone	f1-score	iou_score	loss	precision	recall
EfficientNetB2	0.9682	0.9392	0.0958	0.9692	0.9679
MobileNetV2	0.9562	0.9167	0.1318	0.9578	0.9550

Table 2. Comparing EfficientNetB2 and MobileNetV2 after 49 epochs

Below graph show the comparison of U-Net+MobileNetV2 and U-Net+EfficientB2 after the training the models for 49 epochs and testing on test data.

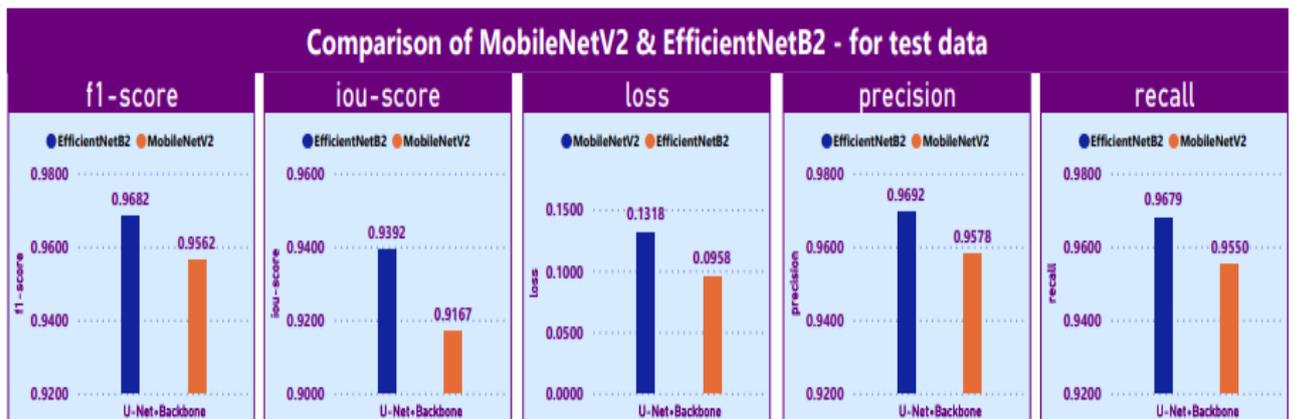


Figure 33. Comparing Metrics - MobileNetV2 & EfficientNetB2 - for test data

5.7.4 Predictions

Below are the images from test data and predictions by U-Net with EfficientNetB2 and MobileNetV2.

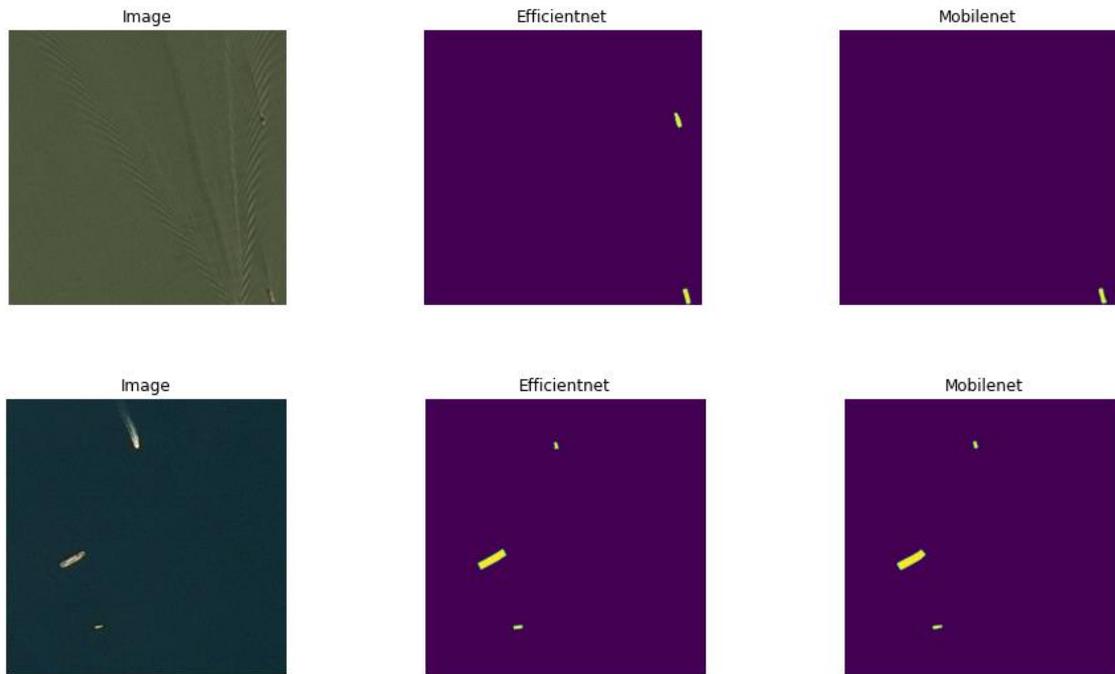


Figure 34. Predictions

5.8. Conclusions

The goal of this study was to evaluate the ship detection using U-Net with well-known transfer learning backbone networks. The U-Net with backbone DenseNet-121, ResNet-50, VGG-16, MobileNetV2, EfficientNetB2, and InceptionV3 were trained initially for 15 epochs with freezing the encoder. U-Net with EfficientNetB2 and MobileNetV2 outperformed the other models.

U-Net with EfficientNetB2 and MobileNetV2 were further trained for 34 epochs (total 49 epochs) for the same data and EfficientNetB2 with F1 score 0.9682 and IoU score 0.9392 performed better than MobileNetV2.

6 FURTHER IMPROVEMENTS

- ❖ Further improvements can be done using U-Net with different backbones and versions.
- ❖ Also we can further train the model to extract more features and predict finer boundary.
- ❖ Other image segmentation model like PSPNet, DeepLab etc. can be used with different backbones.
- ❖ Data augmentation can be done to make the model robust by passing augmented images. The different data augmentation techniques like RGB shift, distort image, zoom out etc. can be used.
- ❖ Ablation study can be done by adding/removing few layers from the backbone.

7 REFERENCES

Airbus ship detection challenge (2018) *Kaggle.com*. Available at: <https://www.kaggle.com/c/airbus-ship-detection/data> (Accessed: June 7, 2022).

Alghazo, J. *et al.* (2021) “Maritime ship detection using convolutional neural networks from satellite images,” in *10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*. *IEEE*; 2021, pp. 432–437.

Alsabhan, W. and Alotaiby, T. (2022) “Automatic building extraction on satellite images using unet and ResNet50,” *Computational intelligence and neuroscience*, 2022, p. 5008854. doi: 10.1155/2022/5008854.

Balodi, T. (2021) *Convolutional neural network with python code explanation*, *Analyticssteps.com*. Available at: <https://www.analyticssteps.com/blogs/convolutional-neural-network-cnn-graphical-visualization-code-explanation> (Accessed: May 9, 2022).

Chen, L.-C. *et al.* (2018) “Encoder-decoder with atrous separable convolution for semantic image segmentation,” *arXiv [cs.CV]*. Available at: https://openaccess.thecvf.com/content_ECCV_2018/papers/Liang-Chieh_Chen_Encoder-Decoder_with_Atrous_ECCV_2018_paper.pdf (Accessed: June 8, 2022).

Chen, Y., Zheng, J. and Zhou, Z. (2018) *Airbus ship detection -traditional v.s. Convolutional neural network approach*, *Stanford.edu*. Available at: <https://cs229.stanford.edu/proj2018/report/58.pdf> (Accessed: May 9, 2022).

Chollet, F. (2016) “Xception: Deep learning with depthwise separable convolutions,” *arXiv [cs.CV]*. Available at: <http://arxiv.org/abs/1610.02357>.

Fei-Fei Li & Justin Johnson & Serena Yeung (2017) “*Detection and Segmentation*,” 10 May. Available at: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf.

Gaudenz Boesch (2021) *VGG Very Deep Convolutional Networks (VGGNet) - What you need to know* - *viso.ai*. Available at: <https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/> (Accessed: May 9, 2022).

Gupta, T. (2017) *Deep learning: Feedforward neural network, Towards Data Science*. Available at: <https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7> (Accessed: May 9, 2022).

Han, L. *et al.* (2020) “Multi-size convolution and learning deep network for SAR ship detection from scratch,” *IEEE access: practical innovations, open solutions*, 8, pp. 158996–159016. doi: 10.1109/access.2020.3020363.

He, K. *et al.* (2016) "Identity mappings in deep residual networks," *arXiv [cs.CV]*. Available at: <http://arxiv.org/abs/1603.05027>.

How does satellite imaging work? (2021) *Globalforestlink.com*. Available at: <https://globalforestlink.com/how-does-satellite-imaging-work> (Accessed: June 8, 2022).

Huang, G. *et al.* (2016) "Densely connected convolutional networks." doi: 10.48550/ARXIV.1608.06993.

Iakubovskii, P. (2020) *Segmentation_models*.

IBM Cloud Education (2020) *What is deep learning?* *Ibm.com*. Available at: <https://www.ibm.com/cloud/learn/deep-learning> (Accessed: May 9, 2022).

Image Segmentation Guide (2021) *Fritz.ai*. Available at: <https://www.fritz.ai/image-segmentation/> (Accessed: June 8, 2022).

Jordan, J. (2018) *Evaluating image segmentation models*, *Jeremy Jordan*. Available at: <https://www.jeremyjordan.me/evaluating-image-segmentation-models> (Accessed: June 8, 2022).

Kanjir, U., Greidanus, H. and Oštir, K. (2018) "Vessel detection and classification from spaceborne optical images: A literature survey," *Remote sensing of environment*, 207, pp. 1–26. doi: 10.1016/j.rse.2017.12.033.

Kurama, V. (2020) *A guide to ResNet, Inception v3, and SqueezeNet*, *Paperspace Blog*. Available at: <https://blog.paperspace.com/popular-deep-learning-architectures-resnet-inceptionv3-squeezenet/> (Accessed: May 9, 2022).

Lendave, V. (2021). A Comparison of 4 Popular Transfer Learning Models [Online]. *Analytics India Magazine*. Available at: <https://analyticsindiamag.com/a-comparison-of-4-popular-transfer-learning-models> (Accessed: 27 May 2022)

Leonardblier, P. (2016) *A brief report of the Heuritech Deep Learning Meetup #5*, *Tech Blog*. Available at: <https://heuritech.wordpress.com/2016/02/29/a-brief-report-of-the-heuritech-deep-learning-meetup-5> (Accessed: June 8, 2022).

Li, L. *et al.* (2020) "A novel CNN-based method for accurate ship detection in HR optical remote sensing images via rotated bounding box," *arXiv [cs.CV]*. Available at: <https://hi.booksc.eu/dl/82170139/ab86b1> (Accessed: June 8, 2022).

Li, Y. *et al.* (2019) "SAR ship detection based on resnet and transfer learning," in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE pp 1188-1191 Available at: <https://doi.org/10.1109/IGARSS.2019.8900290>

Oligschläger, C. (2020) *Detect ships in critical areas*, *EARSC and OGEO Portal: Bringing EO user communities together*. Available at: <https://earsc-portal.eu/display/EOWiki/Detect+ships+in+critical+areas> (Accessed: June 8, 2022).

Pujara, A. (2020) *MobileNet Convolutional neural network Machine Learning Algorithms, Analytics Vidhya*. Available at: <https://medium.com/analytics-vidhya/image-classification-with-mobilenet-cc6fbb2cd470> (Accessed: May 9, 2022).

Pawan, S. (2020) *Residual networks (ResNet) - deep learning, GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/> (Accessed: May 9, 2022).

Ronneberger, O., Fischer, P. and Brox, T. (2015) “U-Net: Convolutional Networks for Biomedical Image Segmentation,” *arXiv [cs.CV]*. Available at: <https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/> (Accessed: May 9, 2022).

Schulz, H. and Behnke, S. (2012) “Deep learning: Layer-wise learning of feature hierarchies,” *KI - Künstliche Intelligenz*, 26(4), pp. 357–363. doi: 10.1007/s13218-012-0198-z.

Seth, N. (2021) *How does Backward Propagation Work in Neural Networks, Analytics Vidhya*. Available at: <https://www.analyticsvidhya.com/blog/2021/06/how-does-backward-propagation-work-in-neural-networks/> (Accessed: May 9, 2022).

Shanmugamani, R. (2018) *Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras*. Birmingham, England: Packt Publishing.

Simonyan, K. and Zisserman, A. (2014) “Very deep convolutional networks for large-scale image recognition,” *arXiv [cs.CV]*. Available at: <http://arxiv.org/abs/1409.1556>.

Singhal, G. (2020) *Introduction to DenseNet with TensorFlow, Pluralsight.com*. Available at: <https://www.pluralsight.com/guides/introduction-to-densenet-with-tensorflow> (Accessed: June 8, 2022).

Szegedy, C. *et al.* (2014) “Going deeper with convolutions,” *arXiv [cs.CV]*. Available at: <http://arxiv.org/abs/1409.4842>.

Tan, M. and Le, Q. V. (2019) “EfficientNet: Rethinking model scaling for convolutional Neural Networks,” *arXiv [cs.LG]*. Available at: <http://arxiv.org/abs/1905.11946>.

Tomar, N. (2021a) *What is residual network or ResNet? — idiot developer, Analytics Vidhya*. Available at: <https://medium.com/analytics-vidhya/what-is-residual-network-or-resnet-idiot-developer-6a1daa7c3b09> (Accessed: June 8, 2022).

Tomar, N. (2021b) *What is UNET? - Analytics Vidhya - Medium, Analytics Vidhya*. Available at: <https://medium.com/analytics-vidhya/what-is-unet-157314c87634> (Accessed: May 9, 2022).

Understanding of MobileNet - EN (2022) Available at: <https://wikidocs.net/165429> (Accessed: June 8, 2022).

Yao, Y., Jiang, Z. and Zhang, H. (2016) “*High-resolution optical satellite image simulation of ship target in large sea scenes,*” in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE.

Zou J., Han Y., So SS. (2008) *Overview of Artificial Neural Networks*. In: Livingstone D.J. (eds) *Artificial Neural Networks. Methods in Molecular Biology*, vol 458. pp.14-22. Humana Press.

8 APPENDICES

8.1 CSC-Puhti sample batch script

Example of batch script for training the image segmentation model with EfficientNetB2.

```
#SBATCH --job-name=EfficientNet-ship

#SBATCH --nodes=1

#SBATCH --ntasks-per-node=1

#SBATCH --cpus-per-task=30

#SBATCH --partition=gpu

#SBATCH --gres=gpu:v100:1,nvme:10

#SBATCH --time=35:00:00

#SBATCH --mem-per-cpu=12G

#SBATCH --account=project_2001220

Export PYTHONPATH=~/.local/lib/python3.8/site-packages/:$PYTHONPATH

#module purge

module load python-data/3.7.6-1

module load tensorflow/2.6

module list

module load plotly

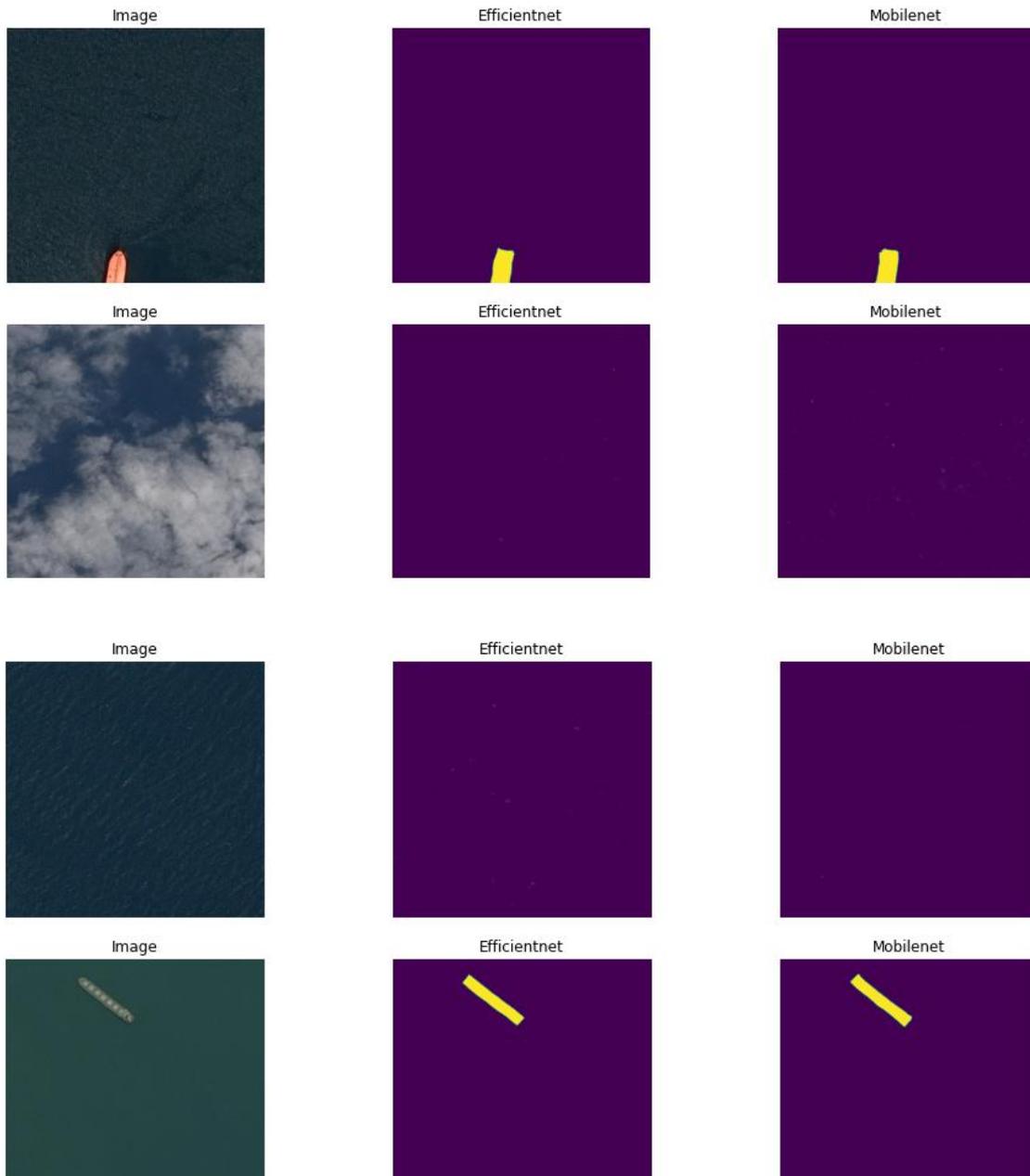
module load segmentation-models==1.0.1

set -xv

srun python3 new_efficientnet.py
```

8.2 Image prediction results

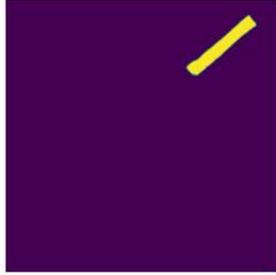
Below images show the original image and the predictions using U-Net with EfficientNetB2 and MobileNetV2 for the test images.



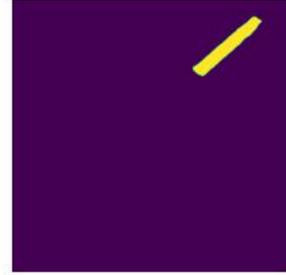
Image



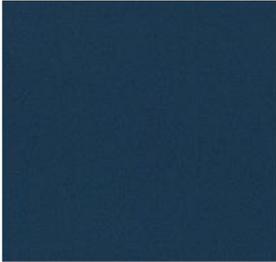
Efficientnet



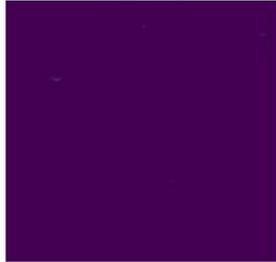
Mobilenet



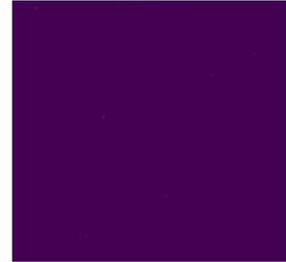
Image



Efficientnet



Mobilenet



Image



Efficientnet



Mobilenet



Image



Efficientnet



Mobilenet



Image



Efficientnet



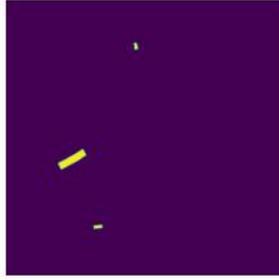
Mobilenet



Image



Efficientnet



Mobilenet

