SHIP CLASSIFICATION USING TRIPLET CNN

PROJECT REPORT

submitted by

ABIN K JAYAN	**ADR18CS013**
AJMAL LATHEEF	**ADR18CS028**
ASHIL JOHNSON	**ADR18CS035**
SIDDHARDH R NAIR	**ADR18CS045**

in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

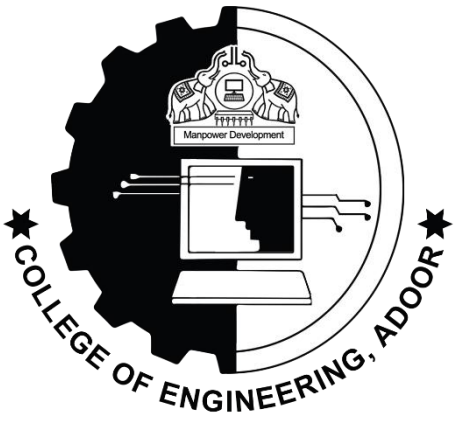
COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Under the guidance of

Mrs. MANJU J



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING ADOOR

2021-2022

SHIP CLASSIFICATION USING TRIPLET CNN

PROJECT REPORT

submitted by

ABIN K JAYAN	**ADR18CS013**
AJMAL LATHEEF	**ADR18CS028**
ASHIL JOHNSON	**ADR18CS035**
SIDDHARDH R NAIR	**ADR18CS045**

in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

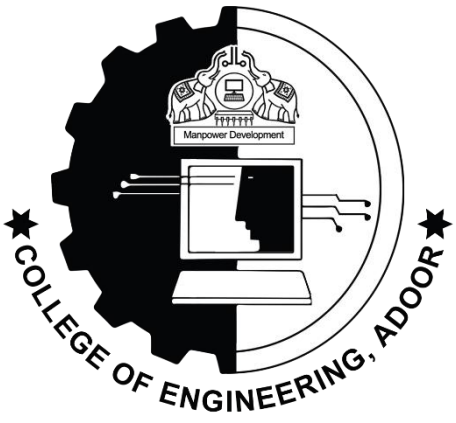
COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Under the guidance of

Mrs. MANJU J

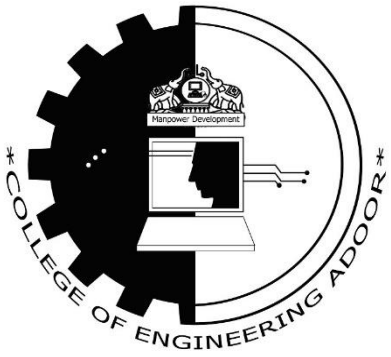


DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING ADOOR

2021-2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING ADOOR



Certificate

Certified that this is a bona fide record of the seminar report entitled

SHIP CLASSIFICATION USING TRIPLET CNN

Submitted by

ABIN K JAYAN	**ADR18CS001**
AJMAL LATHEEF	**ADR18CS003**
ASHIL JOHNSON	**ADR18CS014**
SIDDHARDH R NAIR	**ADR18CS038**

during the year 2021 in partial fulfilment of the requirements for the award of the
Degree of Bachelor of Technology in Computer Science and Engineering
of
APJ Abdul Kalam Technological University, Kerala

Project Guide
Mrs. Manju J
Assistant Professor
Dept of CSE

Head of the Department
Mr. Shibu J
Associate Professor
Dept of CSE, CEA

Project Coordinator
Mrs. Honey Mol
Assistant Professor
Dept of CSE, CEA

External Examiner

ACKNOWLEDGEMENT

We bring out the report of our project work endeavouring our deepest gratitude to God Almighty. We take this opportunity to list a few wonderful individuals.

We would like to extend our gratitude to **Dr K Sunilkumar**, Principal, College of Engineering Adoor, for equipping us with all facilities during all development of our project. Our special thanks to **Mr. Shibu J**, Head of the Department, Computer Science and; Engineering, College of Engineering Adoor.

We make use of this opportunity to express our hearty gratitude to our project guide **Mrs. Manju J** and coordinator**, Mrs. Honey Mol M** and all other staff's support we should have been far from completion of this project. We thank them for assisting us in needs and giving relevant ideas and advice for making this project a success.

This acknowledgement will stand incomplete if our friends and family aren't thanked whose constant encouragement and timely criticism helped us to a great extend and fuelled our destination.

We make this opportunity to thank all who have helped us directly or indirectly through this endeavour.

ABIN K JAYAN

AJMAL LATHEEF

ASHIL JOHNSON

SIDDHARDH R NAIR

ABSTRACT

With the capability to automatically learn discriminative features, deep learning has experienced great success in natural images but has rarely been explored for ship classification in medium-resolution SAR images due to the training bottleneck caused by the small datasets. In existing system the geometric, radiometric, and structural features combined with the Convolutional Neural Network(CNN) methods is used to conduct ship classification in high resolution (HR) SAR images. However, this system show least accuracy due to its limits in network. We propose a new model in which Triplet convolutional neural networks (CNNs) are applied to ship classification by using SAR images with the small datasets. To execute our model, ship chips are constructed from medium-resolution SAR images and split into training and validation datasets. Second, a ship classification model is constructed based on very deep triplet convolutional networks. Then, training and prediction is done using this model. We present a detailed implementation of the proposed model built with python.

CONTENTS

LIST OF FIGURES

LIST OFABBREVIATIONS

CNN	Convolutional Neural Networks
SAR	Synthetic Aperture Radar
HOG	Histogram of Oriented Gradient
SIFT	Scale-Invariant Feature Transform (SIFT)
CFAR	Constant False Alarm Rate
LBP	Linear Bound Pattern
FC	The Fully Connected

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Due to their all-weather, all-day, and high-resolution advantages, synthetic aperture radar (SAR) images have recently been used for ship classification in marine surveillance. There are several satellites that have provided high-resolution SAR images since 2007, such as ASI's COSMO-SkyMed, DLR's TerraSAR-X, Japan's ALOS-2, and China's Gaofen-3, These high-resolution SAR images provide a resolution greater than 3 m that contain rich information about the targets, such as the geometry of ships, which makes discriminating different types of ships possible ['Cargo',' Dredging',' Passenger',' Tanker'].

The methods used for ship classification with SAR images mainly focus on feature selection and optimized classifier techniques. Currently, commonly used features are geometric features, such as ship length, ratio of length to width, distribution of scattering centers, covariance coefficient, contour features , and ship scale; and scattering features, such as 2D comb features , local radar cross section (RCS) density, permanent symmetric scatterers ,and polarimetric characteristics .

Based on the advantages of deep learning, convolutional neural networks (CNNs) are adapted in this paper. To further boost the feature discrimination, the task-specific densely connected CNN is combined with the triplet networks to address the intraclass diversity and interclass similarity via the deep metric learning (DML) scheme. Compared to the normal CNNs, the triplet CNNs comprise three identical networks with each taking input an image of a triplet. One triplet is built by sampling an anchor image, a positive image of the same class to the anchor and a negative image of another class, from the training batch.

Then, the sampled triplets are fed into the triplet CNNs to compute the triplet n, by minimizing which the positive and anchor images are brought closer in the learned feature space and the negative ones are pushed far apart from the anchors such that the intraclass compactness and interclass scatter are explicitly encouraged, rendering the subsequent classification much simpler.

PROBLEM DEFINITION

Despite the positive prospect and trend in SAR Ship Classification, there are still challenges concerned with the efficiency. The common CNN models optimized for classification have achieved impressive results. However, an equal penalization of the classification error for each class and thus is less effective for dealing with the intraclass variety and interclass similarity and only high-resolution images can be used Testing & Predicting. For compensation, we propose a Triplet CNN strategy to learn the classification and deep feature embedding, simultaneously, for ship classification in SAR images and can use medium resolution images for Testing & Predicting. Specifically, different from the traditional CNN pipeline, some tuples of three images derived from a minibatch are sent into a triplet network, and then passed through an embedding transformation module to extract the deep feature embeddings.

CHAPTER 2

LITERATURE SURVEY

Synthetic aperture radar (SAR), an essential aerospace remote sensor, is characterized by its ability to achieve all-weather observation of the ground. Based on this advantage, SAR technology has developed rapidly, and has a wide range of applications in target detection, disaster detection, military operations, and resource exploration.

SAR is an essential tool for maritime surveillance. As the maritime trade and transportation carrier is an important military object, it is of great significance to realize the real-time detection of ship targets in spaceborne SAR images. Due to the satellite's limited computing and storage resources, the accuracy, detection speed, and model size of the target detection algorithm are required simultaneously. At present, the methods of ship target detection in SAR images are mainly divided into two types: traditional detection methods, and target detection methods based on deep learning.

Fig 2.1: optical images of ships and their corresponding SAR images

Traditional ship detection algorithms in SAR images generally detect ship targets by manually selectively extracting features such as gray level, contrast ratio, texture, geometric size, scattering characteristics, histogram of oriented gradient (HOG), and scale-invariant feature transform (SIFT). Generally, they can achieve better detection performance in simple scenes

with less interference. The constant false alarm rate (CFAR) detection algorithm is widely used as a contrast-based target detection algorithm for SAR ship detection. However, each pixel point in the CFAR detection algorithm is involved in the calculation of distribution parameter estimation multiple times, and the calculation of background clutter distribution is extensive. The detection speed cannot meet the demand of real-time.

In recent years, deep learning technology has developed rapidly natural image recognition. Introduction of convolutional neural networks (CNN) into the field of target detection, which has brought new research ideas to target detection, and its application in SAR images has an ample exploration space. A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include layers that perform convolutions. Some of the layers in CNN are;

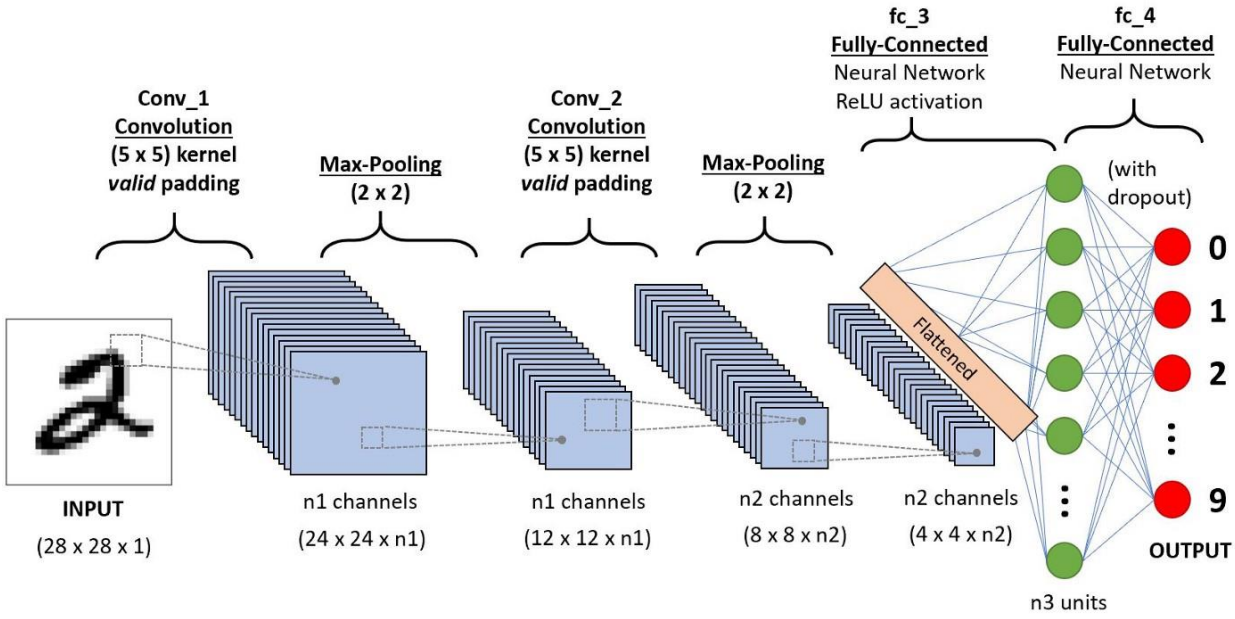


Fig 2.2: CNN Layers

Convolutional layers; the input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels) After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels)

Pooling layers; Convolutional networks may include local and/or global pooling layers along with traditional convolutional layers. Pooling layers reduce the dimensions of data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters, tiling sizes such as 2 x 2 are commonly used. Global pooling acts on all the neurons of the feature map. There are two common types of pooling in popular use: max and average. *Max pooling* uses the maximum value of each local cluster of neurons in the feature map, while *average pooling* takes the average value.

Fully connected layers; Fully connected layers connect every neuron in one layer to every neuron in another layer. It is the same as a traditional multilayer perceptron neural network (MLP). The flattened matrix goes through a fully connected layer to classify the images

A densely connected CNN structure based on the advanced dense convolutional networks (DenseNets) is developed for a more effective representation of MR SAR images. The core idea of DenseNet is the dense connection from the shallow layers to all the deeper ones, by which more new features can be explored to represent the ship targets with less parameters required.
However, in training stage, CNN is always trained to well classify images, without considering intra-class and inter-class distances, while in retrieval stage, the task is almost entirely based on distance metric. It will result in worse retrieval performance if intraclass distances is smaller than inter-class distances for some samples.

Inspired by this, Triplet-CNN, which contains three identical CNNs sharing all the weights and biases, is employed. The net is trained by “telling” machines which samples are of the same class or of different classes, and penalizing it when a sample of a different class is recognized to be

more similar than those of the same class. In training stage, one challenge we may face to is the lack of a large-scale training dataset. Overfitting, a phenomenon that performance on test set is quite terrible in contrast to the perfect performance on training set, will occur if the net is quite huge and complicated while the training set is not large enough. A common method to overcome this challenge is data augmentation with existing labeled data. To overcome the challenge of lacking labeled training data of the same domain in our tasks, we apply several datasets of similar domains in training stage. Datasets in similar domain means datasets sharing similar categories. In feature extraction stage, some useful information of feature representations may be lost after feature activation, although in training stage, feature activation can add nonlinearity to the network model. Therefore, we propose to use non-activation based features to preserve the information. Moreover, samples having similar features at several levels should be regarded as more similar than those having similar features at just one level. Inspired by this, we combine different levels of features for the retrieval tasks.

For Triplet-CNN, three images, including an image selected firstly, an image of the same class (denoted as "+") and an image of another random class (denoted as "−") were separately put into three identical networks, which shares weights and biases, in every training iteration.

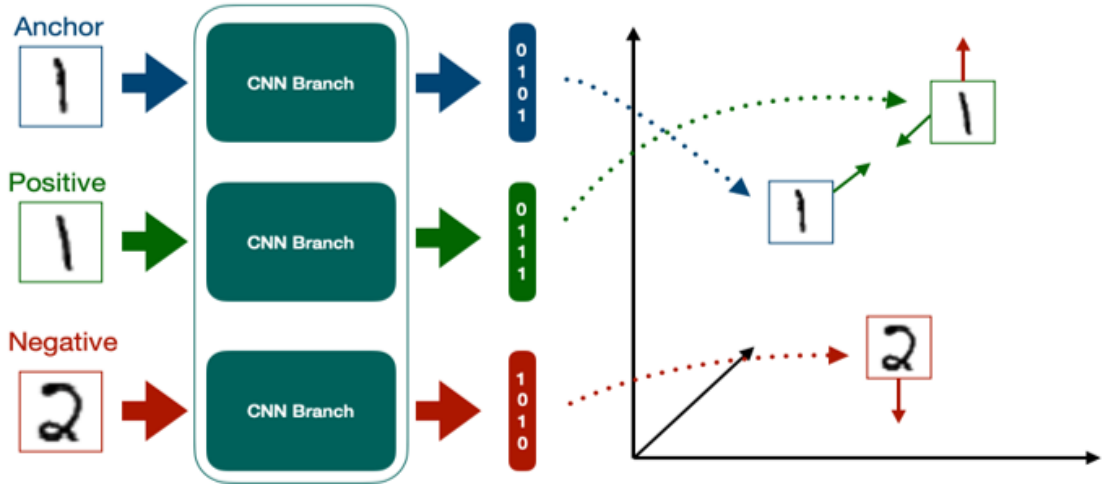


Fig 2.3: Triplet Convolutional neural network

The goal of network training should be ensuring that intraclass distances are at least γ larger than inter-class distances. Therefore, the loss function is $L((\mathbf{x},\mathbf{x}_+,\mathbf{x}_-);\gamma;S)=\max\{0,\gamma-S(\mathbf{y},\mathbf{y}_+)+S(\mathbf{y},\mathbf{y}_-)\}$

defined where S is the similarity function defined, γ is the margin parameter, and (x, x+, x−) is the input triplet set. x+ and x belong to the same class while x− is in a randomly chosen different class. (y, y+, y−) are features of the triplet set. Then,

$$\frac{\partial L}{\partial \mathbf{y}} = \left(\frac{\mathbf{y}^\top \mathbf{y}_+}{\|\mathbf{y}_+\| \|\mathbf{y}\|^3} - \frac{\mathbf{y}^\top \mathbf{y}_-}{\|\mathbf{y}_-\| \|\mathbf{y}\|^3} \right) \mathbf{y} - \frac{1}{\|\mathbf{y}_+\| \|\mathbf{y}\|} \mathbf{y}_+ + \frac{1}{\|\mathbf{y}_-\| \|\mathbf{y}\|} \mathbf{y}_-,$$

$$\frac{\partial L}{\partial \mathbf{y}_+} = -\frac{1}{\|\mathbf{y}_+\| \|\mathbf{y}\|} \mathbf{y} + \frac{\mathbf{y}^\top \mathbf{y}_+}{\|\mathbf{y}\| \|\mathbf{y}_+\|^3} \mathbf{y}_+,$$

$$\frac{\partial L}{\partial \mathbf{y}_-} = \frac{1}{\|\mathbf{y}_-\| \|\mathbf{y}\|} \mathbf{y} - \frac{\mathbf{y}^\top \mathbf{y}_-}{\|\mathbf{y}\| \|\mathbf{y}_-\|^3} \mathbf{y}_-.$$

Considering that there are many triplet sets in each batch, gradients should be computed as follows:

$$\frac{\partial L}{\partial \mathbf{W}^{(l)}} = \sum_{k=1}^{n} \left(\frac{\partial L}{\partial \mathbf{y}^k} \frac{\partial \mathbf{y}^k}{\partial \mathbf{W}^{(l)}} + \frac{\partial L}{\partial \mathbf{y}_+^k} \frac{\partial \mathbf{y}_+^k}{\partial \mathbf{W}^{(l)}} + \frac{\partial L}{\partial \mathbf{y}_-^k} \frac{\partial \mathbf{y}_-^k}{\partial \mathbf{W}^{(l)}} \right),$$

$$\frac{\partial L}{\partial b^{(l)}} = \sum_{k=1}^{n} \left(\frac{\partial L}{\partial \mathbf{y}^k} \frac{\partial \mathbf{y}^k}{\partial b^{(l)}} + \frac{\partial L}{\partial \mathbf{y}_+^k} \frac{\partial \mathbf{y}_+^k}{\partial b^{(l)}} + \frac{\partial L}{\partial \mathbf{y}_-^k} \frac{\partial \mathbf{y}_-^k}{\partial b^{(l)}} \right),$$

where n is batch size. Then weights and bias can be updated according to mini-batch gradient descent. Features are extracted from the last three fully connected layers so that they represent different semantic levels.

CHAPTER 3

PROPOSED SYSTEM

The aim of a synthetic aperture radar (SAR) is to form an image which represents the radar reflectivity of the scene being imaged. Using this mechanism, the images can be captured all-day and in various weather conditions. In our project this SAR is used to take images of ship and using this image classification is performed. Our project classifies four types of ships, they are:

Cargo

Passenger

Tanker

Dredging

SAR images of these ships are passed to the triplet CNN model and result is predicted.

3.1 SYSTEM MODEL

The model consists of three entities user, user interface and backend. For an image classification, user enters the image into the user interface, which will be displayed in the user interface window.

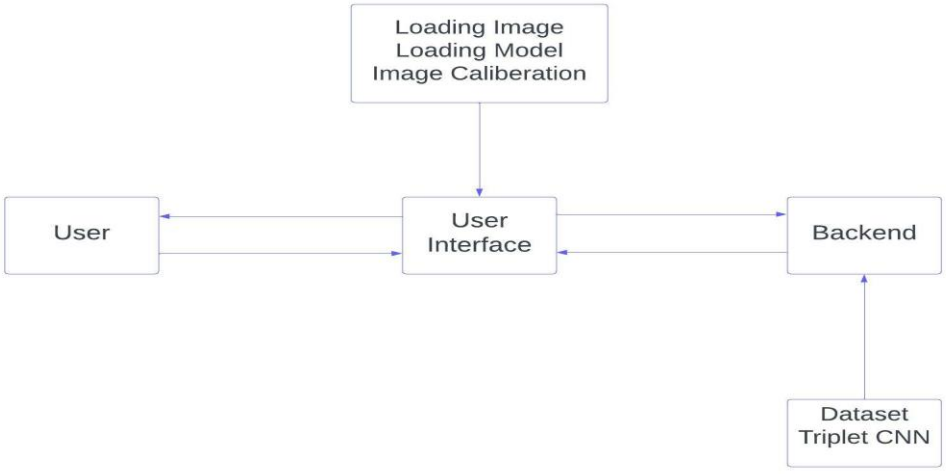


Fig 3.1: System model for ship classification

At the same time, it loads the model and image calibration is done. Then the image is passed to the model and model finds the resultants classification from the backend. For this result classification Triplet CNN is used. From the detailed System Architecture we will get a clear idea about the system.

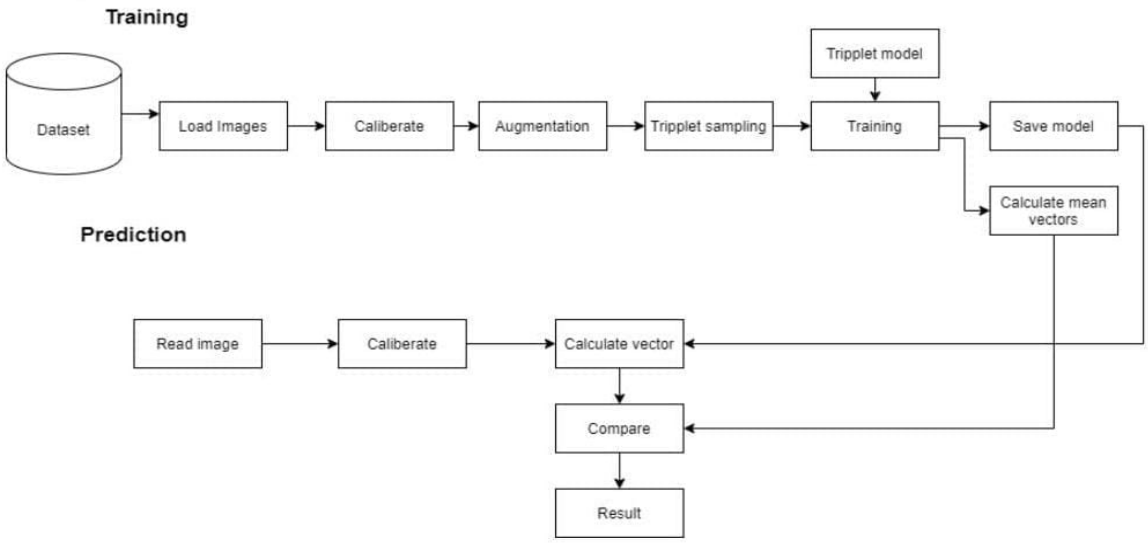


Fig 3.2: Detailed System Architecture of Ship Classification

In training section, the main aim is to reduce the distance vector as much as possible and find a mean vector for each classification. Firstly, the dataset is prepared and image calibration is done. Image calibration is done using Linear Bound Pattern(LBP). Then the image is augmented and the triplet sampling is followed. In this phase our training dataset is converted into triplets, one anchor image, one positive image and one negative image. Using this images, the training is done and the model is saved. And at the same time mean vector of each classification is calculated using the Euclidian distance equation. The mean value is reduced during each classification as the training goes on.

In the prediction phase the image is read by our model via the user interface. Then this image is calibrated to make the size that fit for our bounding box size 28x28 and also the clarity of the image is increased. Then the distance of this image is obtained and is compared with the mean

vector values of the previously trained four classifications. The distance with smallest gap is the resultant classification.

The Level 0 Data flow Diagram(DFD) provide a broad view that is easily digestible but offers little detail. Level 0 data flow diagram of ship classification show a single process node and its connections to external entities.

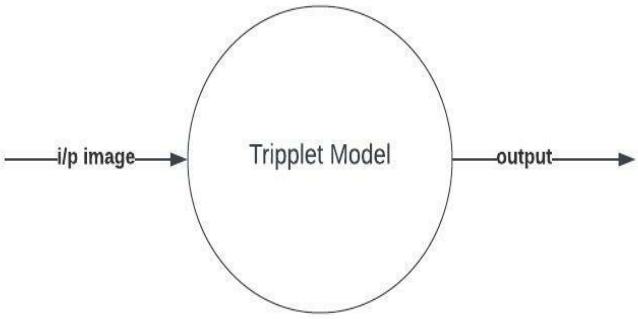


Fig 3.3 Level Zero DFD for ship Classification

In a level 1 data flow diagram, the single process node from the context diagram is broken down into subprocesses. As these processes are added, the diagram needs additional data flows and data stores to link them together.

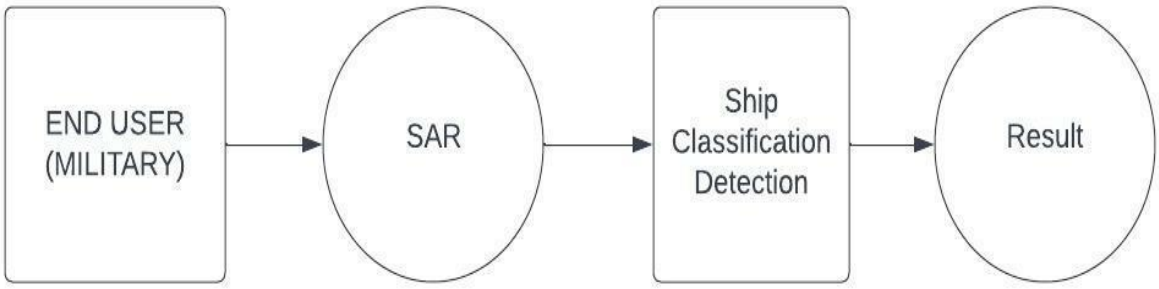


Fig 3.4 Level 1 DFD for Ship Classification

Level 2 DFD give a more detailed information than Level 1. It contains more data nodes and data flow.

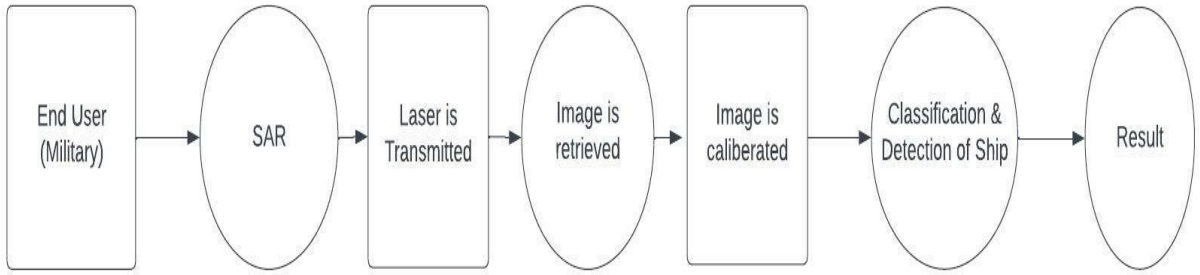


Fig 3.5 Level 2 DFD for Ship Classification

Level 3 DFD gives some more information than Level 2. Its look similar to system architecture and contains several node and data flow.

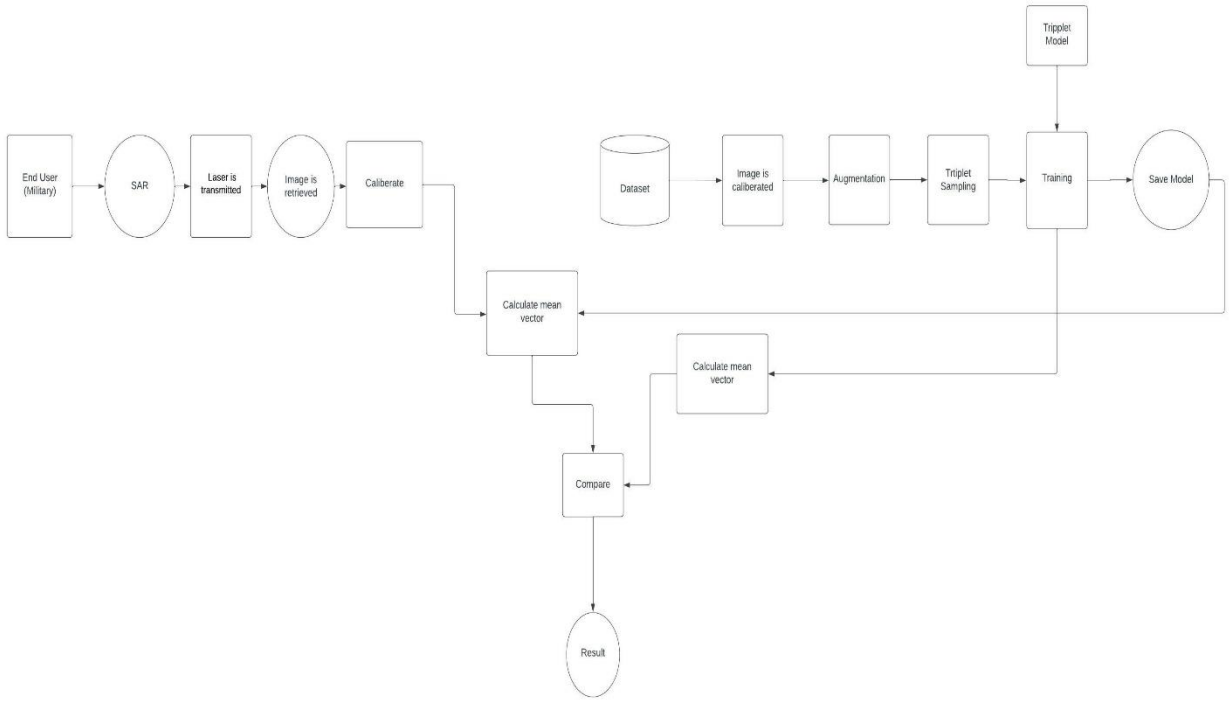


Fig 3.6 Level 3 DFD for Ship Classification

3.1.1 DATASET PREPARATION

We used OpenSARShip database, which made possible to implement ship classification across large-scale SAR data set. However, there are some drawbacks for this data set. First, the sizes of the ship chips are different for various types, which is cumbersome to be applied to the normal CNNs. And for some ship chips, more than two targets exist in one chip that might impair the recognition performance. Most importantly, to some extent, the classified ship types are a bit coarse, which makes it infeasible to perform fine-grained ship classification. The dataset contains SAR images of different ships in different colour. It includes, grey colour, RGB colour etc. So the main challenge is to make all these images into one colour. Then the next challenge is that the size of the image. Normally we are using medium-resolution images in our Project. Our dataset contains image with higher resolution, medium resolution and also low resolution. So the task is to make the images in one resolution. We use bounding box as the size 28x28 pixel, which is considered as an average.

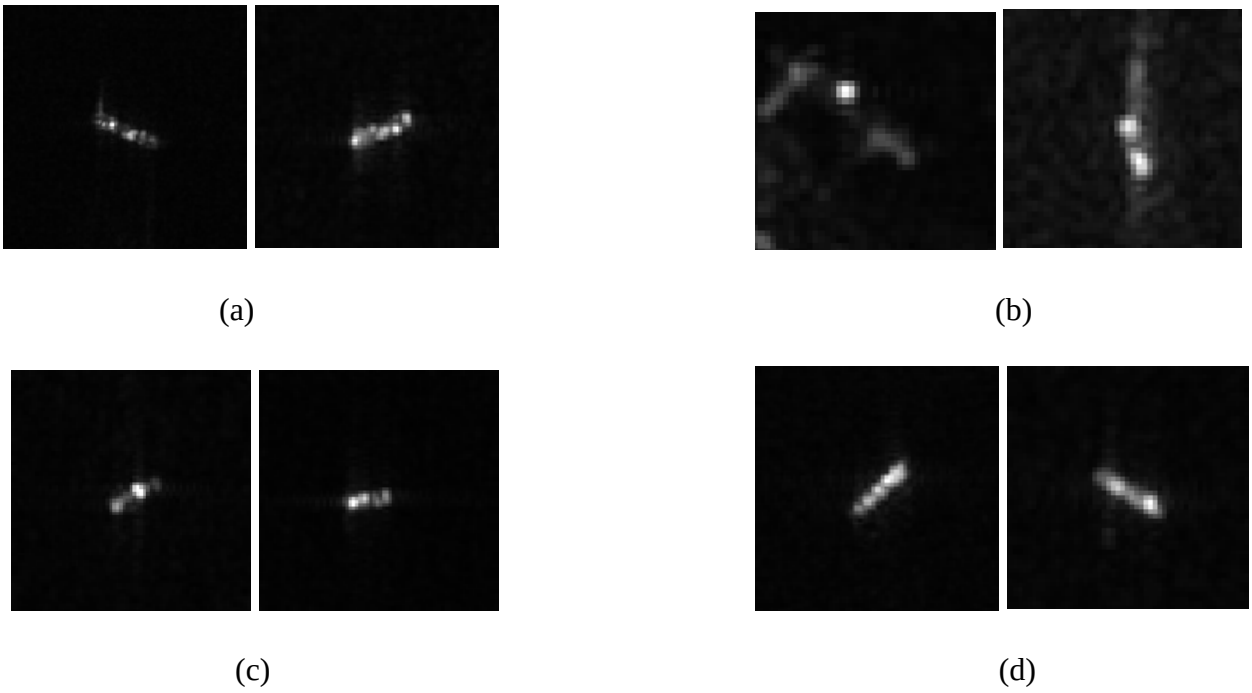


Fig 3.7: SAR images of cargo(a), dredging(b), passenger(c) and tanker(d)

3.1.2 IMAGE LOADING & CALIBERATION

As we all know, image is also known as a set of pixels. When we store an image in computers or digitally, it's corresponding pixel values are stored. So, when we read an image to a variable using OpenCV in Python, the variable stores the pixel values of the image. There are lots of different types of texture descriptors are used to extract features of an image. Local Binary Pattern, also known as LBP, is a simple and grey-scale invariant texture descriptor measure for classification. In LBP, a binary code is generated at each pixel by thresholding it's neighbourhood pixels to either 0 or 1 based on the value of the center pixel.

The rule for finding LBP of an image is as follows:

1. Set a pixel value as center pixel.
2. Collect its neighbourhood pixels (In our project we are taking a 3 x 3 matrix so; total number of neighbourhood pixel is 8)
3. Threshold it's neighbourhood pixel value to 1 if its value is greater than or equal to center pixel value otherwise threshold it to 0.
4. After thresholding, collect all threshold values from neighbourhood either clockwise or anti-clockwise. The collection will give you an 8-digit binary code. Convert the binary code into decimal.
5. Replace the center pixel value with resulted decimal and do the same process for all pixel values present in image.

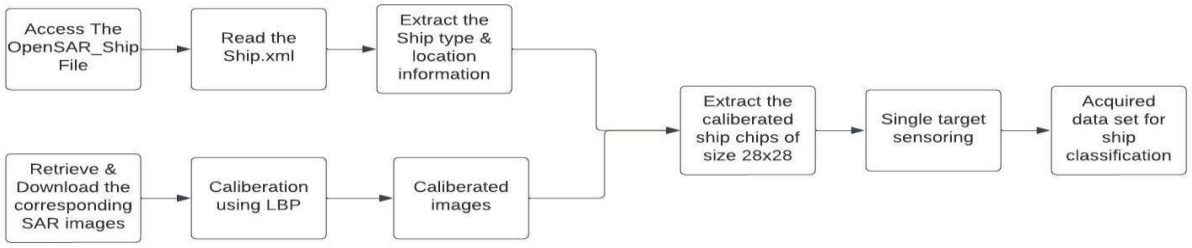


Fig 3.8: DFD for Load Images & Calibration

3.1.3 AUGMENTATION

Since the data volume of each ship type in our data set is unbalanced, we implement data augmentation for the training data set and the transformations are as follows:

1) Horizontally and vertically flip the ship chips.

2) Rotate the ship chips by 90◦, 180◦, and 270◦

3) Randomly translate the ship chips with the translation value of pixels selected from −5 to 5.

4) Add the Gaussian noise with the mean and standard deviation set as 0 and 0.001, respectively

3.1.4 TRIPLET SAMPLING

In triplet sampling, the objective is to build triplets <anchor, positive, negative> consisting of an anchor image, a positive image (which is similar to the anchor image), and a negative image (which is dissimilar to the anchor image). There are different ways to define similar and dissimilar images. If you have a dataset having multiple labels as the target class, then images of the same class can be considered as similar, and images across different classes can be considered as dissimilar.

In our dataset we have the images of different ships. To generate triplets, first two images are selected from one type of a ship and one image is selected from the other one. Now, images of the same classes are considered similar, so one of them is used as an anchor and the other one as positive whereas images from the other class is considered a negative image.

In our work we have selected triplet sampling method called Batch Hard. It is crucial to select appropriate triplets for network training. If the triplets selected satisfy the constraint easily, it will lead to slow convergence. However, to mine the hardest triplets across the whole data set, it might be trapped in bad local minima.

Specifically, suppose there are C classes with K samples per class in a training batch. For each sample in one training batch being selected as an anchor image, which is denoted by

{xa i } a=1,...,K i=1,...,C , and then correspondingly, the hardest positive sample x p i and the hardest negative sample xn j can be, respectively, sampled as follows:

$$\begin{aligned} \boldsymbol{x}_i^p &= \arg\max_{\boldsymbol{x}_i^p} D\left(\boldsymbol{x}_i^a, \boldsymbol{x}_i^p\right) \\ &= \left\| \boldsymbol{f}\left(\boldsymbol{x}_i^a\right) - \boldsymbol{f}\left(\boldsymbol{x}_i^p\right) \right\|_2 \quad \forall p = 1, \ldots, K \\ \boldsymbol{x}_j^n &= \arg\min_{\boldsymbol{x}_j^n} D\left(\boldsymbol{x}_i^a, \boldsymbol{x}_j^n\right) \\ &= \left\| \boldsymbol{f}\left(\boldsymbol{x}_i^a\right) - \boldsymbol{f}\left(\boldsymbol{x}_j^n\right) \right\|_2 \quad \forall j = 1, \ldots, C, \quad j \neq i, \; n = 1, \ldots, K. \end{aligned}$$

Thus, for a training batch with N samples, we can totally get N triplets

3.1.5 TRIPLET MODEL

Our architecture is mainly built with triplet CNN. In this there are three identical CNN network are used. Anchor, Positive and Negative images are passed through their respective CNN network and during backpropagation weight vectors are updated using shared architecture. Each CNN layer consist of:

- A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction
- A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.

CNN Layers

There are three types of layers that make up the CNN which are the convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked, a CNN architecture will be formed. In addition to these three layers, there are two more important parameters which are the dropout layer and the activation function which are defined below.

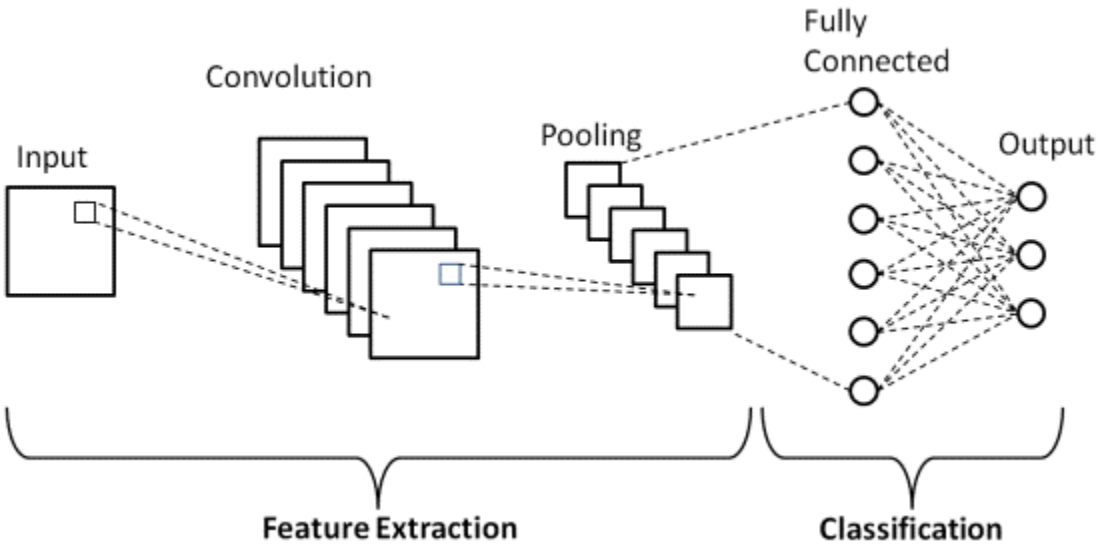


Fig 3.9: CNN Layers

1. Convolutional Layer

This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM).
The output is termed as the Feature map which gives us information about the image such as the corners and edges. Later, this feature map is fed to other layers to learn several other features of the input image.

2. Pooling Layer

In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations.
In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the

predefined section is computed in Sum Pooling. The Pooling Layer usually serves as a bridge between the Convolutional Layer and the FC Layer

3. Fully Connected Layer

The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.
In this, the input image from the previous layers are flattened and fed to the FC layer. The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place.

4. Activation Functions

Finally, one of the most important parameters of the CNN model is the activation function. They are used to learn and approximate any kind of continuous and complex relationship between variables of the network. In simple words, it decides which information of the model should fire in the forward direction and which ones should not at the end of the network.
It adds non-linearity to the network. There are several commonly used activation functions such as the ReLU, Softmax, tanH and the Sigmoid functions. Each of these functions have a specific usage. For a binary classification CNN model, sigmoid and softmax functions are preferred an for a multi-class classification, generally softmax us used.

And finally Loss Function is calculated. Triplet loss is a function for machine learning algorithms where a reference input (called anchor) is compared to a matching input (called positive) and a non-matching input (called negative). The distance from the anchor to the positive is minimized, and the distance from the anchor to the negative input is maximize. Here the network is trained (using a contrastive loss) to output a distance which is small if the image belongs to a known person and large if the image belongs to an unknown person. However, if we want to output the closest images to a given image, we would like to learn a ranking and not just a similarity.

Loss function:

The cost function for Triplet Loss is as follows:

L(a, p, n) = max(0, D(a, p) — D(a, n) + margin)

where **D(x, y):** the distance between the learned vector representation of x and y. As a distance metric L2 distance or (1 - cosine similarity) can be used. The objective of this function is to keep the distance between the anchor and positive smaller than the distance between the anchor and negative.

3.1.6 TRAINING

We have a small dataset for the project. So this small dataset is divided into training dataset and validation set. In all image 80%images are used for training set and 20% images is used for validation set. With the training images it is split into triplet sample and the resultant sample is fed into the network. The aim in the training phase is to calculate the mean vector for each classification. To find the minimum mean repeated training procedure to be followed.

3.1.7 CALCULATE MEAN VECTORS

The mean image and positive image is calculated for each classification. Then after repeated training procedure the distance between the anchor and positive in decreased and the distance between anchor and negative image is increased. Finally, the smallest distance between the anchor and positive image is selected as the mean of that classification.

3.1.8 COMPARISON OF MEAN VECTORS

When the user given an input image the model will finds its vector format which can be useful to find the distance. Then the Euclidian distance between each classification should be calculated. The resultant classification should be the class which have minimum distance with the input image vector. If the distance between Cargo class and image vector is less compared to other class, then the output result will be the class cargo. The formula used to find the Euclidian distance between the class is as follows.

$$\text{Euclidean Distance} = |X - Y| = \sqrt{\sum_{i=1}^{i=n} (x_i - y_i)^2}$$

CHAPTER 4

SYSTEM SPECIFICATIONS

4.1 HARDWARE SPECIFICATION

- ❖Processor: i3
- ❖RAM: 4GB
- ❖Hard Disk: 256GB or above

4.2 SOFTWARE SPECIFICATIONS

- ❖Tool: Sublime text, Anaconda, tkinter
- ❖Language: Python
- ❖Operating System: Windows 7 or later

CHAPTER 5

SYSTEM DEVELOPMENT

5.1 USER INTERFACE

The user interface is built using the tool tkinter. Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

The first page of the tkinter windows enables the user to enter the login credentials. The password and username of the user is already fixed. When the user enters the correct credentials. He will be forwarded to next page and if the user entered the wrong credentials then alert message Wrong Credentials will be displayed.

```
def logcheck():
    global username_var,pass_var
    uname=username_var.get()
    pass1=pass_var.get()
    if uname=="admin" and pass1=="admin":
      showcheck()
    else:
       messagebox.showinfo("alert","Wrong Credentials")
```

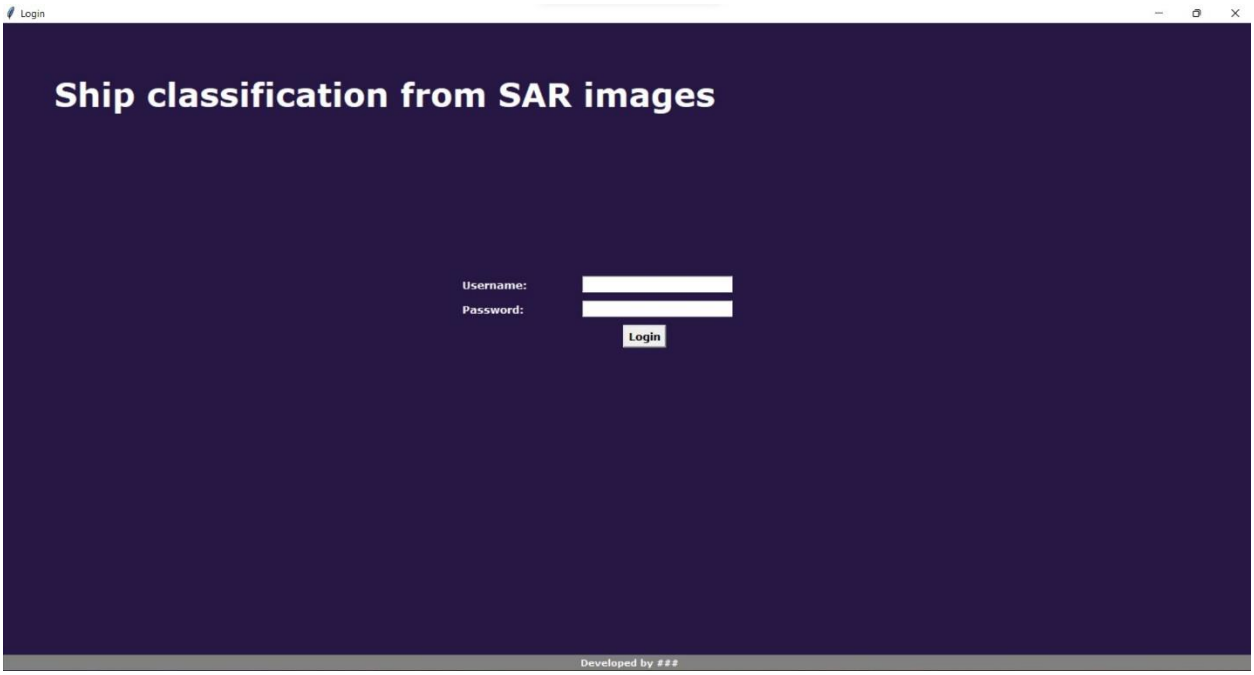


Fig 5.1 Login Page of User Interface

After the successful login then the user will be redirected to the next page

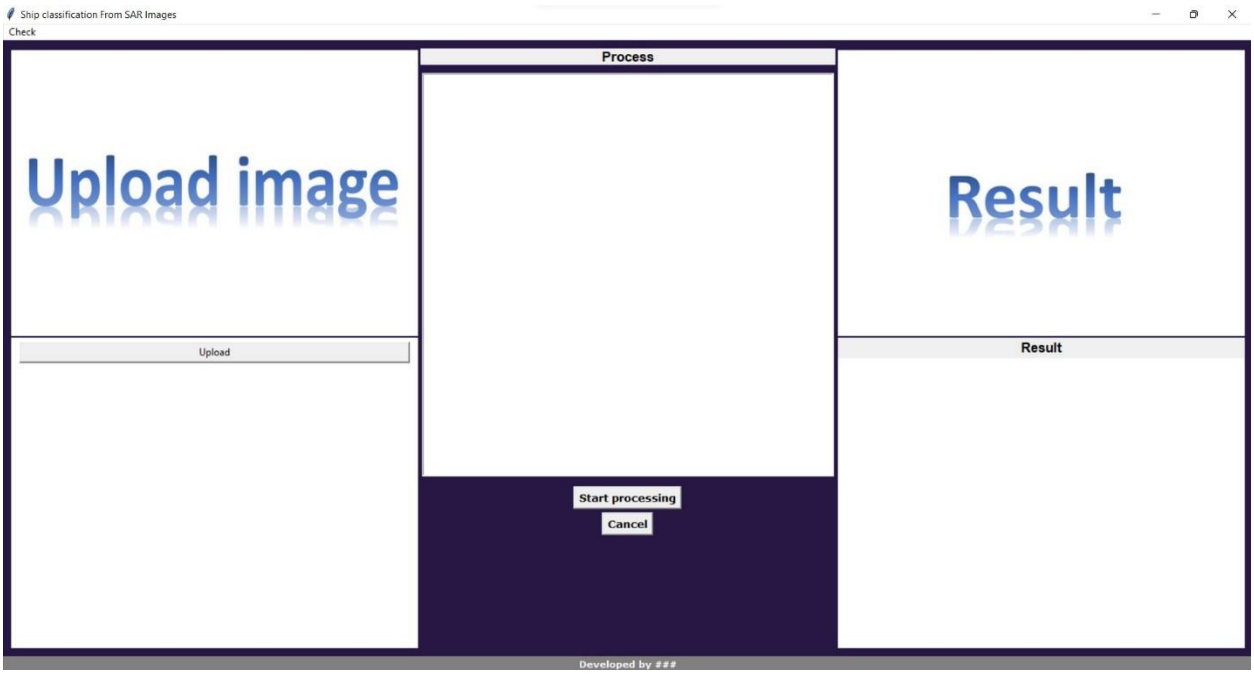


Fig 5.2 Uploading and result display page of User Interface

2nd page of user interface contains many frames in the first frame the user can upload the input image and in the second frame the processes that happened before the result prediction will be displayed and in the final frame the result will be displayed.

5.2 IMAGE CALIBRATION

```
def lbp2(path):

    img_bgr = cv2.imread(path, 1)
    #print(img_bgr)
    img_bgr=cv2.resize(img_bgr,(28,28))
    height, width, _ = img_bgr.shape
    img_gray = cv2.cvtColor(img_bgr,
                            cv2.COLOR_BGR2GRAY)
    img_lbp = np.zeros((height, width),
                       np.uint8)

    for i in range(0, height):
        for j in range(0, width):
            img_lbp[i, j] = lbp_calculated_pixel(img_gray, i, j)

    return img_lbp



# Function for calculating LBP

def lbp_calculated_pixel(img, x, y):
    center = img[x][y]
    val_ar = []
    val_ar.append(get_pixel(img, center, x-1, y-1))

    val_ar.append(get_pixel(img, center, x-1, y))

    val_ar.append(get_pixel(img, center, x-1, y + 1))

    # right
    val_ar.append(get_pixel(img, center, x, y + 1))
```

```
    # bottom_right
    val_ar.append(get_pixel(img, center, x + 1, y + 1))

    # bottom
    val_ar.append(get_pixel(img, center, x + 1, y))

    # bottom_left
    val_ar.append(get_pixel(img, center, x + 1, y-1))

    # left
    val_ar.append(get_pixel(img, center, x, y-1))

    # Now, we need to convert binary
    # values to decimal
    power_val = [1, 2, 4, 8, 16, 32, 64, 128]

    val = 0

    for i in range(len(val_ar)):
        val += val_ar[i] * power_val[i]

    return val
```

5.3 TRAINING PHASE

```
def build_network(input_shape, embeddingsize):
    '''
    Define the neural network to learn image similarity
    Input :
            input_shape : shape of input images
            embeddingsize : vectorsize used to encode our picture
    '''
     # Convolutional Neural Network
    network = Sequential()
    network.add(Conv2D(128, (7,7), activation='relu',
                       input_shape=input_shape,
                       kernel_initializer='he_uniform',
                       kernel_regularizer=l2(2e-4)))
    network.add(MaxPooling2D())
    network.add(Conv2D(128, (3,3), activation='relu', kernel_initializer='he_uniform',
                       kernel_regularizer=l2(2e-4)))
```

```
    network.add(MaxPooling2D())
    network.add(Conv2D(256, (3,3), activation='relu', kernel_initializer='he_uniform',
                kernel_regularizer=l2(2e-4)))
    network.add(Flatten())
    network.add(Dense(4096, activation='relu',
                kernel_regularizer=l2(1e-3),
                kernel_initializer='he_uniform'))


    network.add(Dense(embeddingsize, activation=None,
                kernel_regularizer=l2(1e-3),
                kernel_initializer='he_uniform'))

    #Force the encoding to live on the d-dimentional hypershpere
    network.add(Lambda(lambda x: K.l2_normalize(x,axis=-1)))

    return network



class TripletLossLayer(Layer):
    def __init__(self, alpha, **kwargs):
        self.alpha = alpha
        super(TripletLossLayer, self).__init__(**kwargs)

    def triplet_loss(self, inputs):
        anchor, positive, negative = inputs
        p_dist = K.sum(K.square(anchor-positive), axis=-1)
        n_dist = K.sum(K.square(anchor-negative), axis=-1)
        return K.sum(K.maximum(p_dist - n_dist + self.alpha, 0), axis=0)

    def call(self, inputs):
        loss = self.triplet_loss(inputs)
        self.add_loss(loss)
        return loss

def build_model(input_shape, network, margin=0.2):

     # Define the tensors for the three input images
    anchor_input = Input(input_shape, name="anchor_input")
    positive_input = Input(input_shape, name="positive_input")
    negative_input = Input(input_shape, name="negative_input")
```

```
    # Generate the encodings (feature vectors) for the three images
    encoded_a = network(anchor_input)
    encoded_p = network(positive_input)
    encoded_n = network(negative_input)

    #TripletLoss Layer
    loss_layer =
TripletLossLayer(alpha=margin,name='triplet_loss_layer')([encoded_a,encoded_p,encod
ed_n])

    # Connect the inputs with the outputs
    network_train =
Model(inputs=[anchor_input,positive_input,negative_input],outputs=loss_layer)

    # return the model
    return network_train
```

5.4 PREDICTION PHASE

```
def predict_type(img):
    global loaded_model,m0,m1

    m0=mean1(0)
    m1=mean1(1)
    m2=mean1(2)
    m3=mean1(3)

    li=[]

    preds = loaded_model.predict(img)[0]
    d0=compute_dist(m0,preds)
    print(d0)
    li.append(d0)
```

```
        d1=compute_dist(m1,preds)
        print(d1)
        li.append(d1)
        d2=compute_dist(m2,preds)
        print(d2)
        li.append(d2)
        d3=compute_dist(m3,preds)
        print(d3)
        li.append(d3)

        return np.argmin(li)



#To Compute the distance

def compute_dist(a,b):
    return np.sum(np.square(a-b))

#To compute the mean
def mean1(n):
    model=keras.models.load_model('model')
    if n==0:
        f1=open('df0.pkl','rb')
        df=pickle.load(f1)
        f1.close()
        preds=model.predict(df)
        m=np.mean(preds,axis=0)
        print(m)
```

```
        return m
    if n==1:
        f1=open('df1.pkl','rb')
        df=pickle.load(f1)
        f1.close()
        preds=model.predict(df)
        m=np.mean(preds,axis=0)
        print(m)
        return m
    if n==2:
        f1=open('df2.pkl','rb')
        df=pickle.load(f1)
        f1.close()
        preds=model.predict(df)
        m=np.mean(preds,axis=0)
        print(m)
        return m

    if n==3:
        f1=open('df3.pkl','rb')
        df=pickle.load(f1)
        f1.close()
        preds=model.predict(df)
        m=np.mean(preds,axis=0)
        print(m)
        return m
```

CHAPTER 6

RESULT AND DISCUSSION

A total of 6 scenes from the COSMO-SkyMed level 1A HIMAGE (single-look complex Slant (SCS) product) images in the X-band with single HH or VV polarization are used to evaluate our approach, and the detailed information is shown in Table 6.1. The data are calibrated and converted into grey in the range [28,28]. The ships are found automatically or manually on the SAR images. The Cargo have repeating textures due to the box-like hatches in the longitudinal direction. The Passenger also have repeating characteristics due to truck-size containers. The tanker has symmetric features because of the intense backscattering of pipelines along the centerline.

Sensors	Imaging time	Pixel Spacing (Azi × Rg, m)	Near Range Incidence (°)	Band	Orbit
COSMO-SkyMed12	12 July 2018	1.90 × 2.23	58.79	X	Descending
COSMO-SkyMed12	12 July 2018	0.93 × 2.22	24.87	X	Descending
COSMO-SkyMed12	12 July 2018	1.67 × 2.05	48.63	X	Ascending
COSMO-SkyMed12	13 July 2018	1.90 × 2.23	58.79	X	Descending
COSMO-SkyMed12	13 July 2018	1.66 × 2.05	48.63	X	Ascending

Table 6.1: Details of SAR Images

To evaluate the effectiveness of our proposed method, other models are also used for comparison. Other models are built using CNN and its result is verified. The comparison between triplet CNN and normal CNN is done and efficiency is compared. The CNN model use the input image as 80*80 pixel and use CNN layers for the feature extraction. The CNN model requires high-resolution image where the triplet CNN only require medium-resolution image. The classification accuracy and validation accuracy of triplet CNN is shown in the table. In

addition, precision, recall, and F_1 score are also used to evaluate the influence of units in the fully connected layers.

Number of Testing	Training Accuracy	Validation Accuracy	Average Precision	Average Recall	Average F1 Score
80	100	97.66	97.85	0.9774	0.9779

Table 6.2: Results of Triplet CNN

In the below figure we can see the result of a classification. When we upload an image of tanker the result displayed is also Tanker. We can see that the image contains process frame where all the processes are listed and the output is displayed in result frame.

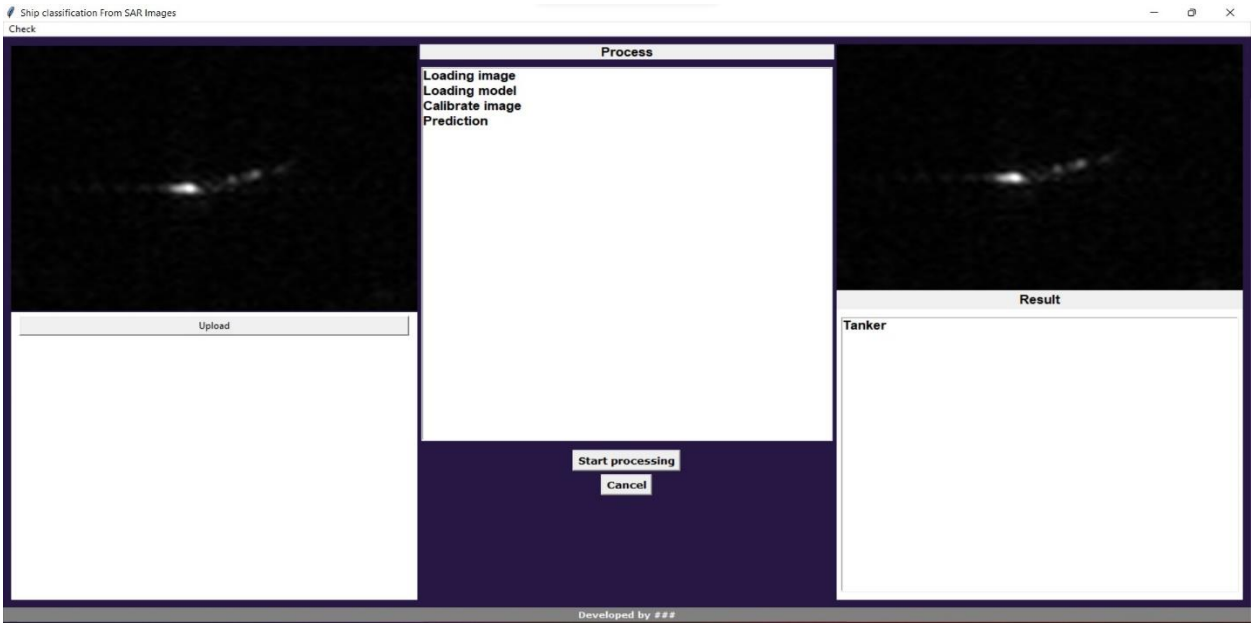


Fig 6.1 Output of a prediction

From above results we can say that the model produces a higher accuracy. The triplet CNN model yields better results. Now the result of normal CNN model is needed to compare with the above results. The result of normal CNN is

Number of Testing	Training Accuracy	Validation Accuracy	Average Precision	Average Recall	Average F1 Score
80	100	89.66	89.13	0.9134	0.9213

Table 6.3: Results of CNN

CNN shows a very low performance compared to triplet CNN. In the above table the normal CNN result is depicted. The results say that its accuracy is very low and precision is also very low.

Through the above experiments and analysis, it can be clearly seen that among the above two models, triplet CNN shows greater accuracy. The model based on triplet CNN achieves the best F_1 score. It will be our future goal to adapt ensemble learning to take advantage of these models and achieve the best performance. This may be characteristic of this dataset and more ship classification datasets are needed to verify this in the future. In addition, due to the differences between SAR images and optical images, large SAR image datasets will be constructed to better exploit the methodology of deep learning, and future research will be conducted on pre-trained models with SAR images, which may show the benefits of transfer learning and combined SAR inherent characteristics (e.g., statistical distributions with convolutional neural networks) to enhance the classification results of ship classifications. Not only does fine tuning achieve promising results, it also results in overfitting of the present model. The increase in datasets is likely to solve this problem.

CHAPTER 7

CONCLUSION

In this paper, convolutional neural networks are used to classify ships in medium-resolution SAR images with small datasets. Specifically, firstly, a SAR ship dataset is constructed from 6 COSMO-SkyMed images and, considering the characteristics of SAR images, a ship classification model based on triplet CNN is constructed. Second, fine tuning is utilized to train the model, which is pre-trained on an ImageNet dataset. The experimental results reveal that our proposed ship classification model achieves the best performance. It shows the better accuracy compared to the CNN model. Our model is mainly focused for the military purposes. When a ship is entered to the border of our country, the SAR will take the images of the ship. Then using these SAR image, the user can classify whether the ship is tanker or not. So these project have a greater role in the field of security.

REFERENCES

[1] Wang C., Zhang H., Wu F., Jiang S., Zhang B., Tang Y. A novel hierarchical ship classifier for COSMO-SkyMed sar data. IEEE Geosci. Remote Sens. Lett. 2014;11:484–488. doi: 10.1109/LGRS.2013.2268875. [CrossRef] [Google Scholar].

[2] Zhang L., Zhang L., Du B. Deep learning for remote sensing data: A technical tutorial on the state of the art. IEEE Geosci. Remote Sens. Mag. 2016;4:22–40. doi: 10.1109/MGRS.2016.2540798. [CrossRef] [Google Scholar]

[3] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, "A tutorial on synthetic aperture radar," IEEE Geosci. Remote Sens. Mag., vol. 1, no. 1, pp. 6–43, Mar. 2013.

[4] D. J. Crisp, "The state-of-the-art in ship detection in synthetic aperture radar imagery," DSTO Inf. Sci. Lab., Edinburgh, SA, Australia, Tech. Rep. DSTO-RR-0272, 2004

[5] Y. Wang and H. Liu, "A hierarchical ship detection scheme for highresolution SAR images," IEEE Trans. Geosci. Remote Sens., vol. 50, no. 10, pp. 4173–4184, Oct. 2012

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. Int. Conf. Learn. Represent., 2015, pp. 1–14.

[7] M. Wilmanski, C. Kreucher, and J. Lauer, "Modern approaches in deep learning for SAR ATR," Proc. SPIE, vol. 9843, May 2016, Art. no. 98430N

[8] J. Ding, B. Chen, H. Liu, and M. Huang, "Convolutional neural network with data augmentation for SAR target recognition," IEEE Geosci. Remote Sens. Lett., vol. 13, no. 3, pp. 364–368, Mar. 2016.

[9] T. Chen, K.-F. Ji, X.-W. Xing, H.-X. Zou, and H. Sun, "Ship recognition in high resolution SAR imagery based on feature selection," in Proc. Int. Conf. Comput. Vis. Remote Sens., Xiamen, China, Dec. 2012, pp. 301–305

[10] E. Hoffer and N. Ailon, “Deep metric learning using triplet network,” in Proc. Int. Conf. Learn. Represent. Workshops, 2015, pp. 1–8.

[11] Y. Wang, C. Wang, and H. Zhang, “Ship classification in high-resolution SAR images using deep learning of small datasets,” Sensors, vol. 18, no. 9, p. 2929, Sep. 2018.

[12] Z. Zhao, K. Ji, X. Xing, W. Chen, and H. Zou, “Ship classification with high resolution TerraSAR-X imagery based on analytic hierarchy process,” Int. J. Antennas Propag., vol. 2013, no. 1, 2013, Art. no. 698370.