Ship Classification Using an Image Dataset

Okan Atalar (okan@stanford.edu), Burak Bartan (bbartan@stanford.edu)

***Abstract*—In this project, we developed three different sets of classification algorithms to classify different ship types based on color (RGB) images of ships. An image of a ship is the input to our classification algorithms, which are mainly bag of features, convolutional neural networks (CNNs), and SVM along with some preprocessing. The best-performing of these is CNN, and is able to classify a given ship into its corresponding category with an accuracy probability of 0.8822 when using 10 different classes of ships.**

I. Introduction

The increased presence of autonomous systems requires reliable classification algorithms to understand their surrounding environment. These autonomous systems have the potential to find widespread use in sea and ocean waters, necessitating a reliable classification of their surrounding. Since ships are the most popular means of transportation and warfare in seas and oceans, they need to be classified by autonomous systems. We are therefore interested in applying machine learning, and computer vision techniques to the problem of reliably classifying ships into different classes using captured ship images in different lighting conditions, image quality, and distance to the ships.

The set of algorithms we use for ship classification take ship images in red, green and blue (RGB) color format as input and outputs the most likely ship category that the input image represents. For the ship classification setting, we applied three different classification algorithms: Preprocessing of images followed by SVM with a set of features that we choose, bag of features method, and a convolutional neural network trained by AlexNet.

II. Related Work

The work [2] uses the same images from the same website we are using. They consider $140,000$ ship images from 26 different ship categories for classification. Their baseline method, which uses Crammer and Singer multi-class SVM with feature vectors extracted from a VGG-F network, achieves an accuracy of 0.54. Their CNN, which utilizes AlexNet, achieves an accuracy of 0.73. They do not however, consider using bag of features or SVM method with preprocessing.

III. Dataset and Features

The dataset that we used consists of 10 different classes of ships: aircraft carriers, bulkers, cruise ships, fire-fighting vessels, fishing vessels, inland dry cargo vessels, restaurant ships, motor yachts, drilling rigs, and submarines. We obtained the dataset by downloading classified ship images from [1]. We used 1000 images for training for the SVM algorithm and 200 for validation with 5 different classes: aircraft carriers, bulkers, cruise ships, fire-fighting vessels, and fishing vessels. The reason being that the accuracy we obtained with preprocesssing followed by SVM was comparable to the other two methods: bag of features method using 10 classes and convolutional neural networks using 10 classes. On average we used 1000 images from each class for training and 200 for testing for bag of features method. Convolutional neural networks used on average 1680 images for training and 360 for testing.

Fig. 1. 4 examples from each of the first 5 image categories: Aircraft carriers, bulkers, cruise ships, fire-fighting vessels, fishing vessels (from left to right).

A. Preprocessing of Images

The dataset consists of ship images taken at different orientations, distances, and with varying background. Since the background of the ship contains limited information regarding the ship category and introduces significant noise to the image, we aim to remove this randomness by cropping a part of the image which contains only the ship. The objective in this step is to crop the ship image, without throwing away pixels related to the ship. We first perform edge detection on the initial image, since the ship has well defined borders. The flow diagram for image cropping based on edge detection is demonstrated in Fig. 2.

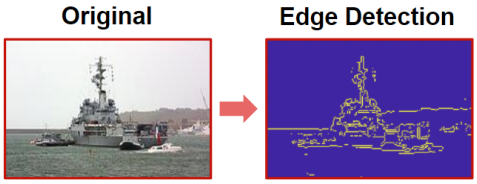

Fig. 2. Edge Detection

Since the ship has well defined borders, our goal is to determine the rows and columns in the image corresponding to these edges. We therefore sum the pixels along a row in the image and compare with the sum in the next row. If there is a big difference between the values for the two rows, this

indicates an edge in that row. Due to the random background, edge detection also detects random artefacts. To overcome the noise and make the algorithm robust against errors, we find the sum over a number of rows, which we define as the bandwidth. After the row and column locations are found where there is a significant change in edges detected, we crop the image. The flow diagram is shown in Fig. 3 for cropping.

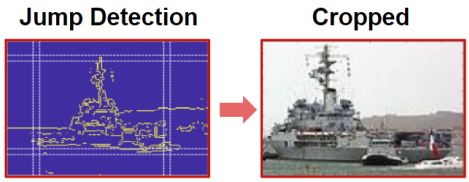

Fig. 3. Image Cropping

We first determine the minimum and maximum row locations for the ship by finding the row such that the ratio of row summed over the row bandwidth to the row a bandwidth away summed over the same bandwidth is maximized. The row locations are estimated first rather than column locations since most ship images are oriented horizontally (parallel to sea surface) and therefore longer in the horizontal direction. We then use the row estimates when determining the column locations for robustness. Let $|Image(r,c)|$ denote the gray color converted from RGB by taking the magnitude with respect to RGB, and where r denotes the row number c the column number. We are trying to estimate r_{min} and r_{max} by summing over the row bandwidth r_{BW}. After the minimum and maximum row locations have been estimated, we use these estimates for determining the minimum and maximum column locations, c_{min} and c_{max} by summing again over a column bandwidth of c_{BW}. The equations used for determining the minimum and maximum row and column locations are shown in Eq. 1-4.

$$r_{min} = \arg\max_{r_x} \left(\frac{\sum_{r=r_x-0.5r_{BW}}^{r_x+0.5r_{BW}} \sum_{c=1}^{c_{max}} |Image(r,c)|}{\sum_{r=r_x-1.5r_{BW}}^{r_x-0.5r_{BW}} \sum_{c=1}^{c_{max}} |Image(r,c)|} \right) \quad (1)$$

$$r_{max} = \arg\min_{r_x} \left(\frac{\sum_{r=r_x-0.5r_{BW}}^{r_x+0.5r_{BW}} \sum_{c=1}^{c_{max}} |Image(r,c)|}{\sum_{r=r_x-1.5r_{BW}}^{r_x-0.5r_{BW}} \sum_{c=1}^{c_{max}} |Image(r,c)|} \right) \quad (2)$$

$$c_{min} = \arg\max_{c_y} \left(\frac{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_y-0.5c_{BW}}^{c_y+0.5c_{BW}} |Image(r,c)|}{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_y-1.5c_{BW}}^{c_y-0.5c_{BW}} |Image(r,c)|} \right) \quad (3)$$

$$c_{max} = \arg\min_{c_y} \left(\frac{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_y-0.5c_{BW}}^{c_y+0.5c_{BW}} |Image(r,c)|}{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_y-1.5c_{BW}}^{c_y-0.5c_{BW}} |Image(r,c)|} \right) \quad (4)$$

Due to noise artefacts in the background, this algorithm may not properly crop the image. To capture such mistakes, the dimensions of the cropped image are checked. If the ship image is smaller than 10 pixels in the vertical direction and 75 pixels in the horizontal direction, then the cropped image is probably not containing the ship. In this case, we do not crop the image and work with the whole picture. It should be noted that these numbers were based on the dataset that we used and will vary based on the number of pixels used to represent each image in the dataset.

B. Feature Extraction

After the ship image has been cropped, the related features must be extracted for classification. Relevant features that we identified include the power spectrum samples of the two dimensional image after normalizing the cropped image size to 70 rows by 140 columns, the RGB color moment generating function, and the ratio of the number of rows to the columns in the cropped image.

The power spectrum for an image is calculated by normalizing the size of the cropped image to 70 rows by 140 columns, followed by taking the two dimensional Fourier Transform of the normalized image (calculated as in Eq. 5), and taking the absolute squared. Since the cruise ship contains many windows and variations within its body, high frequency components show up in its two dimensional power spectrum, shown in Fig. 5, unlike the cruise ship, shown in Fig. 4. To prevent overfitting, we only use "some" samples of the two dimensional power spectrum. In particular, we sample the Fourier Transform at regular intervals (we used 7 samples with regular spacing from the power spectrum). Additionally, since the Fourier Transform also contains phase information which we do not care about, we only pass on the magnitude square of the Fourier Transform coefficients after sampling with regular intervals to capture 7 samples. Since the ship images contain different lighting conditions, the total amount of power present in each image varies. To normalize this variation in power, we normalize the power in each image and therefore compute the power normalized power spectrum for each image. The power normalized power spectrum is calculated as in Eq. 6 by using the two dimensional Fourier transform of the image computed by Eq. 5. $F_s(k_1,k_2)$ is then used as a set of features, where n = 5 and m = 10 (sampling the power spectrum with a spacing of 5 in the row and 10 in the column directions, respectively.

$$F(k_1,k_2) = \sum_{r=r_{min}}^{r_{max}} \sum_{c=c_{min}}^{c_{max}} Image(r,c) e^{-i2\pi(k_1 r/r_{max} + k_2 c/c_{max})} \quad (5)$$

$$F_s(k_1,k_2) = \frac{|F(nk_1,mk_2)|^2}{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_{min}}^{c_{max}} (Image(r,c))^2} \tag{6}$$

Another feature we use is the color distributions. Different ship categories are predominantly the same color. For instance, aircraft carriers are usually gray, fire-fighting vessels are usually red etc. The RGB colors for a firefighter-vessel is shown in Fig. 6. We also use higher orders of color distributions (moment generating functions), since color distributions is also critical. This is based on the mathematical fact that knowing all the moment generating functions of a distribution is equivalent to knowing the actual distribution. Calculating the n^{th} order moment for red color (denoted R_n) is expressed in Eq. 7. Taking the first 5 orders yielded the best accuracy. This choice depends on the dataset and may show variability when tested with different datasets. The n^{th} order moment for green and blue colors is also done in the same way with using the green and blue values for the image. Similar to the power normalization when computing the samples of the power spectrum, we also normalize the moment generating function with respect to the total power in the image to achieve consistency among different images captured under different lighting conditions.

$$R_n = \frac{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_{min}}^{c_{max}} (Image_R(r,c))^n}{\sum_{r=r_{min}}^{r_{max}} \sum_{c=c_{min}}^{c_{max}} |Image_R(r,c)|^2} \tag{7}$$

The color distributions and the power spectrum for the image show great variability among value. To compensate for this huge difference, similar to preprocessing before Principal Component Analysis (PCA), we normalize the mean and variance. Let $x^{(i)}$ represent the features extracted pertaining to the i^{th} image and j the j^{th} feature. Let $\mu = \frac{1}{m}\sum_{i=m} x^{(i)}$. Replace each $x^{(i)}$ with $x^{(i)} - \mu$ to have zero mean. Let $\sigma_j^2 = \frac{1}{m}\sum_{i=m}(x_j^{(i)})^2$. Replace each $x_j^{(i)}$ with $\frac{x_j^{(i)}}{\sigma_j}$ to normalize the variance.

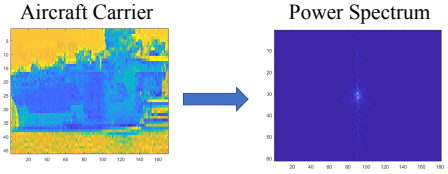

Fig. 4. Aircraft Carrier Power Spectrum

IV. Methods

We used three different learning algorithms: preprocessing followed by SVM, bag of features, and convolutional neural networks trained from scratch and using transfer learning (AlexNet).

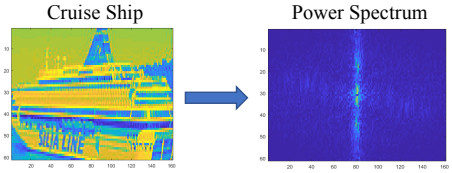

Fig. 5. Cruise Ship Power Spectrum

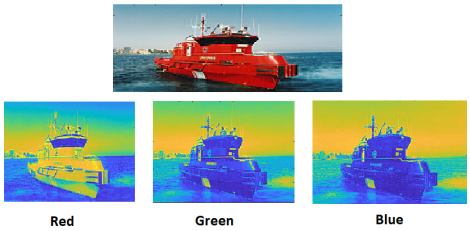

Fig. 6. RGB Representation of Image

A. Preprocessing Followed by SVM

The relevant features of the image are extracted after the image has been cropped. As aforementioned, the relevant features are: two dimensional power spectrum of image, moment generating functions for red, green, and blue and the ratio of number of rows to columns in the cropped image. These features are vectorized and used with a conventional SVM. The SVM algorithm is shown in Eq. 8, where w are the parameter we are trying to learn and ϵ is the error margin.

$$\begin{aligned} \underset{\gamma,w,b}{\text{minimize}} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\epsilon_i^2 \\ \text{subject to} \quad & y^{(i)}(w^T x_b^{(i)}) \geq 1 - \epsilon_i, i = 1,...,m, \\ & \epsilon_i \geq 0, i = 1,...,m. \end{aligned} \tag{8}$$

The SVM algorithm tries to separate the data by hyperplanes in the high dimensional space of the feature vectors. Based on the samples from the training data, the algorithm separates the data points by hyperplanes determined by the number of classes present in the classes. During the testing stage, the algorithm outputs the class of the input image by extracting the relevant features as aforementioned and then outputting the ship class based on the point defined by the features in the hyperspace. The parameter C in Eq. 8 is equal to $1/n$, where n is the number of parameters we have. For the preprocessing algorithm followed by SVM, we have a total of 37 features: 7x3 from samples of the power spectrum, 5x3 from samples of the color moment generating function, and 1 from ratio of number of rows to columns. Therefore, $C = 1/37$.

B. Bag of Features

This method [3] is based on SURF feature description. SURF is an algorithm that can be used for both feature detection and description. In our method, we do not use the

feature detection; instead we use grids of various sizes as features to be described. For feature description, however, we use the SURF algorithm. Feature description is the process of obtaining a numerical vector for every feature; in our case, for every grid. The steps of the bag of features method is visually summarized in Fig. 7, taken from [4].

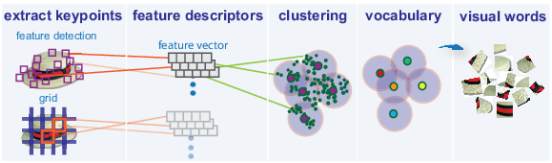

Fig. 7. Summary of bag of features method

After obtaining a feature vector for every image, we then drop 25% of the feature vectors. We then apply k-means clustering to the remaining 75% of the feature vectors found so that we can have a feature basis that we can represent our images in. The basis vectors are the centroids from k-means algorithm, and they are referred to as vocabulary, as in the bag of words method for language processing.

The next step of the algorithm is to project every image into this feature space, and the resulting projections are our inputs to the multi-class SVM classifier.

C. Convolutional Neural Networks (CNNs)

We have experimented with two different networks: A network from scratch, and another that uses transfer learning (namely, AlexNet). First let us describe the common properties of both approaches, and then we will talk about the differences of the methods. A representative figure of the CNNs we considered is visually depicted in Fig. 8, taken from [5].

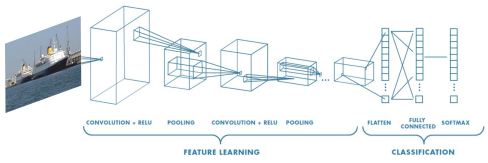

Fig. 8. Layers of the CNN

In both of the methods, we use the rectifier function as the activation function given in Eq. 9.

$$g(z) = max(0, z) \tag{9}$$

Rectifier function is desirable in CNNs because they make the computations faster, and are widely preferred in practice.

We use dropout layers in both methods. This is because we found that there was a considerable amount of overfitting. Dropout layers make the network less complicated by dropping some portion of the neurons from the previous layer. A less complicated network implies less variance, which helps reduce overfitting.

We have max pooling layers in both methods to speed up the training, and somewhat control overfitting. Batch normalization layer is to normalize the intermediate layer outputs, and it prevents the neuron outputs from getting too big. It also leads to faster learning.

The ending layers in both methods are also the same. We use a softmax layer as the last layer for classification. Softmax function is as given in Eq. 10.

$$\phi_i = \frac{exp(\theta_i^T z)}{\sum_{j=1}^{K} exp(\theta_j^T z)} \tag{10}$$

The output of the softmax layer is a K-dimensional vector of probabilities. Namely, the i'th entry of this vector gives the probability that the input ship image belongs to class i.

1) From scratch: In this method, we train a network from scratch. We consider a network with 5 convolutional layers. Each convolutional layer is followed by a batch normalization layer, a ReLu layer, and a maxpooling layer. Connected to the last maxpooling layer is a dropout layer to reduce the overfitting. A fully connected layer with 10 neurons (corresponding to 10 ship categories) follows. Then, finally we have the softmax layer for classification.

In MATLAB, we also specify a classification layer after the softmax layer. This layer is to indicate what type of loss we want to use. We used the cross entropy function as the cost function, which is given in Eq. 11 for a single example:

$$CE = -\sum_{k=1}^{K} y_k log(\hat{y_k}), \tag{11}$$

where y_k is the true label in one-hot representation, and $\hat{y_k}$ is the probability that the given example belongs to class k.

2) AlexNet: In this method, we use transfer learning (AlexNet, [6]) to decrease the training time, and obtain higher performance. We copy the first 22 layers of the AlexNet network, and attach three layers to it: A dropout layer, a fully connected layer, and a softmax layer. The initial layers of CNNs, even for very different classification tasks, are similar in that they detect similar features such as edges. The transfer learning basically aims to transfer the learning done before on a large dataset to a new network. Because the weights from a pre-trained network will likely be closer to the optimal weights than a randomly chosen set of weights, transfer learning is quite a useful technique.

V. Experiments/Results/Discussion

We evaluated the performance of our learning algorithms using the ratio of the correct labels to the number of examples. We call this ratio the accuracy of the algorithm. To evaluate the accuracy of our results, we extracted the confusion matrix from the data and the overall accuracy for test samples..

A. Preprocessing with SVM

The SVM learning method obtained an overall training accuracy of 0.4685 and test accuracy of 0.457. Since the training and test accuracies are very similar, we can conclude that there is very little bias in our learning algorithm. To prevent overfitting, we only used samples of the power spectrum for each image by regular sampling. Nearby points were not used

TABLE I
CONFUSION MATRIX FOR THE METHOD OF PREPROCESSING WITH SVM

	$\hat{y}$				
y	1	2	3	4	5
1	0.5350	0.1700	0.2050	0.0150	0.0750
2	0.2500	0.3100	0.1400	0.0850	0.2150
3	0.2200	0.1400	0.4400	0.0200	0.1800
4	0.1150	0.1250	0.1100	0.5050	0.1450
5	0.0750	0.1000	0.1450	0.1850	0.4950

since they would be very similar in value, but highly prone to noise. The same also applied for the color moment generating function, orders higher than 5 were not used. The confusion matrix for the test sample is shown in Table I. The columns are the probabilities of classification. Diagonal entries correspond to the probability of correct classification for a class. From the confusion matrix we see that the algorithm was most successful in classifying aircraft images and least successful for classifying bulkers.

B. Bag of Features

The training accuracy of this method is 0.57, and the test accuracy is 0.44 therefore we observe overfitting. We tried reducing the bias, but this was the best we could get. There is a lot of noise in the images, and the background can vary greatly from one image to another. For example, a considerable number of images have mountains in the background. This could be misleading the bag of features method especially if this is a common feature among images from different classes. The confusion matrix for this method is given in Table II, which illustrates that the images from classes $4, 5, 7$ (namely, fire-fighting vessels, fishing vessels, and tour boats) are the ones that get misclassified the most often. This observation is reasonable since the variation in the images among these classes is the highest.

TABLE II
CONFUSION MATRIX FOR THE METHOD OF BAG OF FEATURES

	$\hat{y}$									
y	1	2	3	4	5	6	7	8	9	10
1	0.3800	0.1700	0.1350	0.0300	0.0250	0.0400	0.0350	0.0400	0.1000	0.0450
2	0.0450	0.7400	0.0500	0.0100	0.0300	0.0400	0.0050	0.0100	0.0550	0.0150
3	0.0750	0.0850	0.4750	0.0700	0.0100	0.0700	0.0450	0.0450	0.0600	0.0650
4	0.0700	0.0900	0.1100	0.1750	0.0750	0.1400	0.1050	0.0550	0.0850	0.0950
5	0.0550	0.0700	0.0800	0.0750	0.2350	0.0600	0.1100	0.1450	0.0300	0.1400
6	0.0450	0.0950	0.0200	0.0650	0.0250	0.6450	0.0150	0.0500	0.0100	0.0300
7	0.0500	0.0100	0.0400	0.0850	0.1250	0.0450	0.1800	0.2900	0.0700	0.1050
8	0.0200	0.0450	0.1250	0.0300	0.0800	0.1450	0.0950	0.4050	0.0300	0.0250
9	0.0300	0.0500	0.0150	0.0100	0.0350	0.0050	0.0100	0.0100	0.8050	0.0300
10	0.0450	0.0750	0.0500	0.0700	0.0500	0.1800	0.0300	0.0650	0.0450	0.3900

C. Convolutional Neural Networks (CNNs)

We will mostly talk about the results of the CNN with AlexNet since it outperforms the CNN from scratch. Specifically, the test accuracy is 0.7633 for the CNN from scratch, and 0.8822 for the CNN with AlexNet. To be able to use AlexNet, we resized every image into 227×227 to use AlexNet. We used bicubic interpolation for resizing.

The confusion matrix of the CNN with AlexNet for the test set is given in Table III. The confusion matrix illustrates that categories that cause the highest misclassification probabilities are the $5, 7, 8$'th classes. These are fishing vessels, tour boats, and motor yachts, respectively. A visual inspection over the images of these types of images make it clear why they are difficult to classify, namely, it is because they do not have a lot of distinctive features.

TABLE III
CONFUSION MATRIX FOR THE METHOD OF CNN WITH ALEXNET

	$\hat{y}$									
y	1	2	3	4	5	6	7	8	9	10
1	0.9103	0	0.0160	0	0.0032	0.0032	0.0096	0.0353	0.0032	0.0192
2	0.0160	0.9291	0.0046	0.0114	0	0.0137	0.0137	0	0.0023	0.0092
3	0.0502	0	0.8995	0.0046	0	0	0.0091	0.0228	0	0.0137
4	0.0106	0.0160	0	0.8830	0.0160	0.0213	0.0213	0.0053	0.0106	0.0160
5	0.0202	0.0058	0.0029	0.0462	0.8179	0.0058	0.0867	0.0058	0.0058	0.0029
6	0.0200	0.0044	0.0022	0.0089	0.0111	0.9000	0.0356	0.0067	0	0.0111
7	0.0102	0.0081	0	0.0305	0.0265	0.0204	0.8208	0.0794	0	0.0041
8	0.0225	0.0023	0.0068	0.0023	0.0023	0.0045	0.1149	0.8401	0.0023	0.0023
9	0.0188	0.0150	0	0.0038	0	0.0038	0.0113	0.0038	0.9398	0.0038
10	0.0308	0.0066	0	0	0.0066	0.0132	0.0132	0.0088	0.0066	0.9143

The training progress of the CNN that uses AlexNet is given in Fig. 9. This figure illustrates that there is a little overfitting. There was initially more overfitting, but we added a dropout layer which drops its outputs with probability 0.5. Some more parameters: Minibatch size was set to 32, learning rate was 0.0001. The parameters of the first 22 layers are all coming from the pre-trained AlexNet network. Furthermore, we have used the validation set accuracy to choose the hyper parameters.

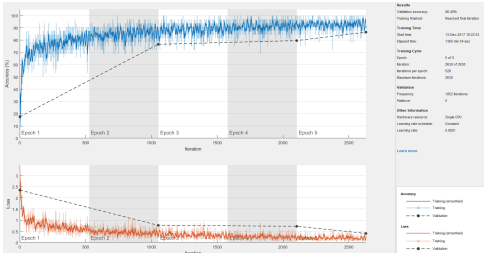

Fig. 9. Training progress of the CNN with AlexNet

VI. CONCLUSION/FUTURE WORK

It is clear that CNN with AlexNet significantly outperforms the other methods, which is expected given CNNs' performance in image classification tasks are superior to other known methods. Combined with transfer learning, the performance went up even further.

There is still some overfitting in the CNN method, and even more in the bag of features method. Even though we tried to reduce it by adding regularization terms, making the models simpler by reducing the number of parameters, in the end, we are still left with some overfitting. This is mainly due to the fact that we do not have enough data. CNN method used 16800 training samples, which is too little for CNN. This problem could be solved through data augmentation as future work.

Although there is no overfitting for the preprocessing followed by SVM method, the attained accuracy is low compared to the other two methods. To improve the performance, wavelet analysis could be used to generate additional features.

VII. Contributions

- Okan Atalar: Worked on Preprocessing, SVM, experiments and downloading dataset.
- Burak Bartan: Worked on bag of features and CNN methods, and their experiments.

References

[1] Shipspotting.com, Home - ShipSpotting.com - Ship Photos and Ship Tracker, *ShipSpotting.com*. [Online]. Available: http://www.shipspotting.com/. [Accessed: 21-Oct-2017].

[2] E. Gundogdu, B. Solmaz, V. Ycesoy, and A. Ko, MARVEL: A Large-Scale Image Dataset for Maritime Vessels, *Computer Vision ACCV 2016 Lecture Notes in Computer Science*, pp. 165180, 2017.

[3] C. Dance, J. Willamowski, L. Fan, C. Bray, and G. Csurka. Visual categorization with bags of keypoints. In ECCV International Workshop on Statistical Learning in Computer Vision., Prague, 2004.

[4] Mathworks.com. Available: https://www.mathworks.com/help/vision/ug/image-classification-with-bag-of-visual-words.html

[5] Mathworks.com. Available: https://www.mathworks.com/discovery/convolutional-neural-network.html.

[6] Mathworks.com. Available: https://www.mathworks.com/help/nnet/ref/alexnet.html