UNIVERSITEIT VAN AMSTERDAM

BACHELOR OPLEIDING KUNSTMATIGE INTELLIGENTIE

Detecting Plastic Soup automatically

Using pre-trained Convolutional Neural Networks

Bachelor thesis
Credits: 18EC

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Author:
Ysbrand M. A. A. GALAMA
UvAID: 10262067

Supervisor:
Thomas MENSINK

Intelligent System Lab Amsterdam
University of Amsterdam

23rd June 2015

Contents

Abstract

Large amounts of plastic end up in the world's oceans. This project uses state-of-the-art Computer Vision techniques to detect Plastic Soup. Recent studies have shown that Convolutional Neural Networks (CNN) can distinguish images within the classes the network was trained on. This project will not train a CNN, however a CNN will be used as a feature extractor for a Support Vector Machine (SVM). The tests in this project show a high accuracy on the dataset that was constructed for this project. There are even several test resulting in the localisation of Plastic Soup within the images, however more research is needed before this can be used as a real-world application.

1 Introduction

The environment is constantly being influenced by humanity and this influence has been debated since the hippie movement in the '60s. The debate on the weight humanity has on the environment has also been fuelled in recent years; the full impact of oil-spills and climate change are still being debated about. [citation needed?]

This project will focus on one of the environmental hazards: Plastic Soup. However, the project will not focus on the problem, or how Plastic Soup affects nature, but how Artificial Intelligence can help to clean up the waste.

1.1 Plastic Soup: Environmental problem

Large amounts of plastic waste end up in the world's oceans and have a significant impact on marine life (Barnes and Milner, 2005). Plastic Soup or Plastic Ocean are both a collective noun for this environmental problem. As Barnes and Milner indicate, the amount of flotsam in the oceans grows annually as does the increasing proportion of plastic.

When plastic objects are disposed of in rivers, they float through rivers into the ocean. The currents in the oceans transport the flotsam around the world to influence marine life. In figure 1.1 the probable location of the Great Pacific Ocean Patch is shown[1]. Because the plastic does not decay in nature, a lot of marine life ingests the plastic into their digestive system. Figure 1.2a shows how much plastic can end up in a bird.

In addition to the dangers for marine life, large amounts of plastic end up on beaches, as shown in figure 1.2b. This has a sizeable impact on the

The most significant problem is probably micro-plastic. When plastic floats in water for some time, sunlight and minerals break the plastic objects down into grains. These grains of plastic are usually several micro meters in size and can infect the ecosystem to a large extend Moore and Phillips (2011).

The weight of Plastic Soup on the environment has caused several organisations to form in order to address this problem and find solutions to the problem. The nascency of organisations addressing the problem and searching for solutions has gained media attention and thus cause the growing awareness for the Plastic Soup.

These organisations would benefit with systems that can detect plastic automatically. Therefore, this project will focus on techniques that are able to distinguish plastic from marine life in ocean water.

[1] source: `http://education.nationalgeographic.com/education/encyclopedia/great-pacific-garbage-patch`

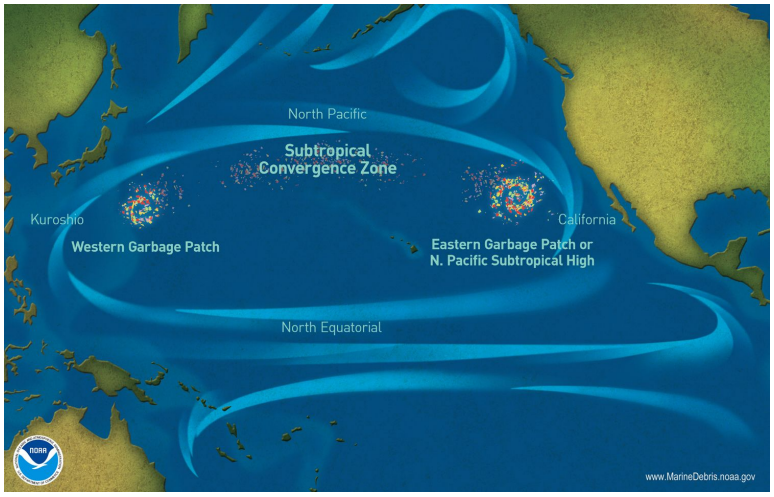

Figure 1.1: Ocean currents in the Pacific that 'collect' the Plastic Soup. Map by NOAA

(a) The stomach contents of a bird that ate plastic. Photograph by Chris Jordan, U.S. Fish and Wildlife Service

(b) A polluted beach in Mumbai, India. Photograph by EPA

Figure 1.2: Impact of Plastic Soup on the environment and society

1.2 Project outline

To develop a system that can detect plastic, state-of-the-art imaging techniques will be used. In recent years the accuracy of imaging techniques on object recognition have been growing, as section 2 describes. This project will research how these techniques will perform on Plastic Soup detection.

A dataset of annotated images has been constructed to be used in this project. This dataset can be used to train and test the algorithm. To

recognise the plastic containing images, a Convolutional Neural Network (CNN) will be used. The CNN will not be trained on the data itself, however a pre-trained network will be used to construct a feature-vector of the image. These feature-vectors will be used to train a classifier to classify the images in ether containing or not containing plastic. The classifier used in this project is a Support Vector Machine (SVM); to distinguish between plastic and marine-life, two SMVs will be trained.

Both CNNs and SVMs are known to reach high accuracy in Computer Vision, therefore it is hypothesised that this method for plastic recognition should give an appropriate baseline.

Besides the detection of images containing plastic, this project will also research if the location of the plastic within an image can be detected. As a prove of concept, the pipeline of the plastic recognition will be adapted a small amount to give results for localisation. To truly localise the plastic in the image, more research will be necessary.

1.3 Outline of the thesis chapters

The performance of the plastic-soup detector made in this project are described in section 6 from the results in section 5. In section 4 the method of the project is shown, section 3 shows how the data used in this project was acquired, and in section 7 I will go into the parts that can be improved in further research. But first, in section 2 the current state-of-the-art techniques in the field of Computer Vision.

2 Theoretical framework

The history of Image Recognition begins as a summer school of Stanford. As was the case in that age, Artificial Intelligence made some wild assumptions on the reachability of their projects. [citation needed] The recent developments show how mistaken they were.

Through the years multiple algorithms have been made to accomplice Computer Vision. The model used to describe images has been a grid with a value (the pixels). This results in a multidimensional function, on which several linear and geometric algebra algorithms can be used on. Usually three steps are taken to identify an image. First using the gradient in images, key-points of the image are identified. Thereafter, these key-points are used to construct features or models, describing the image. Finally these models are tested against a classifier. [citation needed]

2.1 Image representations

As stated above, local image gradients are used to make models on which classifiers can be trained. Several techniques exists that extract features from the image and make a descriptor. A wildly used technique for extracting the features is a 'Bag of Features'; the frequency of the features result in a multidimensional vector of the size of the number of features. From the data-points representing the images, a classifier can be trained. [citation needed]

Another possibility is using a Convolutional Neural Network. The CNN does a similar trick with the features. However, it also constructs the features itself from the image-data and has the possibility to be used as a classifier. In other words, a CNN can be used for each of the steps needed for classifying images.

The CNN technique is currently popular with image recognition tasks. From large amounts of data, an accurate recognition system can be trained (Girshick et al., 2014), (Razavian et al., 2014).

The biggest difference between 'classical' imaging techniques –such as Bag of Features– and Convolutional Neural Networks is the self-learning of the CNN. The neurons of a CNN are trained on the data and construct their own representation that describes the images. In contrary to classical techniques, from which the representations are constructed by humans that build the algorithms.

In the Machine Learning the Convolutional Neural Network (CNN) has existed for quite some time (Fukushima, 1980). However, due to computational complexity is was only recently that Krizhevsky et al. (2012) showed an implementation to compute large CNNs. Krizhevsky et al. shows how to implement a CNN on a computer's GPU that can work with the massive parallelism necessary to compute many layers of nodes.

Neural Networks consist of nodes, connected with weighted edges. When data is presented to the input of the network, each node calculates with its input an output, which results in an output of the network. When the output of the network is incorrect according to the label of the input, a backward calculation in the network is performed. This calculation makes an adaptation to the weights in such a manner, that the network will output the correct label in the next run.

A CNN is a neural network that consists of several layers of nodes, with sometimes many millions of neurons. Figure 2.3 shows the layers of neurons the CNN used by Krizhevsky et al..

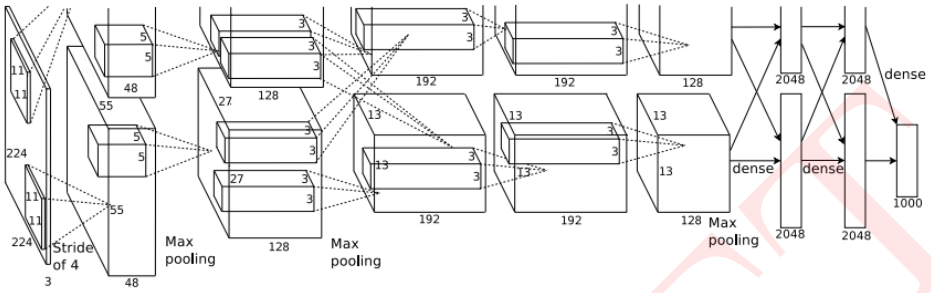

Figure 2.3: Model of the layers of neurons in the CNN as used by Krizhevsky et al. (2012)

Though a CNN can be used to classify a dataset of images, training such a network uses much computing power. A pre-trained CNN has the weights of the neurons adapted to the dataset used for training. This results in a network trained on other classes than used for this project. However, a method that can be applied to work around this problem is inherent to the workings of a CNN.

The bottom layers of a CNN trained on large image datasets detect basic visual concepts, for instance lines, corners and simple shapes. Only higher in the network the concepts become more abstract and less local (Zeiler and Fergus, 2014). Therefore, using the second-to-last layer of a pre-trained CNN an abstract representation of the image can be obtained.

This second-to-last layer can be represented as a multidimensional vector in a space the size of the number of abstract features. As stated before, a feature-vector can be used to train a classifier. Effectively, this project uses a pre-trained CNN as a feature extractor of an image to train an SVM classifier.

2.2 Image classification

Classifying feature-vectors has also several possible algorithms. The Support Vector Machine (SVM) is one of the classification algorithms in Machine

Learning that is wildly used in supervised image classification.

The SVM uses the data-points as vectors in a high dimensional space. In this space, several Linear Algebra techniques are used to fit a hyperplane in this space, where the distance (i.e. a projection of the vector on the plane) of each data-point is maximised. Figure 2.4 shows an example in 2D of possible manners to divide the data-points in the classes; the solid line shows a maximised classifier. The SVM results in a classifier where the sign of the projection vector classifies the data-points (i.e. on which side of the plane the data-point is).

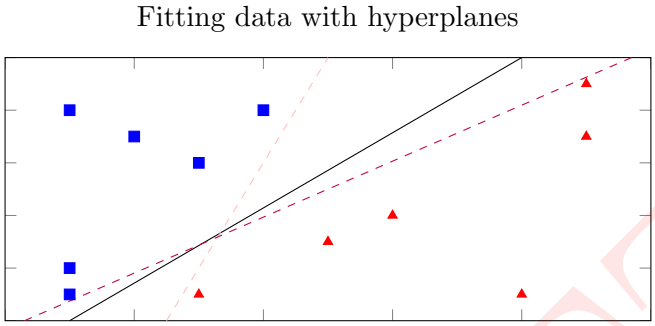

Figure 2.4: Example of possible possible classifications in an algorithm. The dotted lines are a worse fit than the continuous one.

The hyperplane that describes the classifier can be described by, besides linear, other, more complex, formulae. However, these planes usually need more calculation time than a linear plane. Figure 2.5 shows the three hyperplanes used in this project. The data in these examples show the necessity of the different hyperplanes for different data.

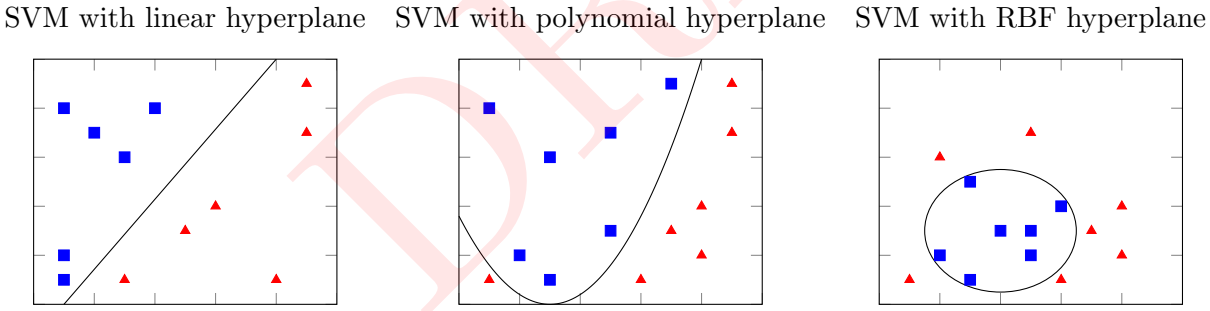

Figure 2.5: Examples of different hyperplanes of a two dimensional SVM

3 Dataset of Plastic Soup images

Because no dataset was available for this project, one was constructed by hand.

The dataset used in this project is made from short films by Bill MacDonnald[2]. Several of these clips consist of floating plastic from a viewpoint both above and below water. These clips have been segmented in single frames and selected for use in this project. A total of 37165 images were left to annotate. The annotation has been done by my hand, classifying each image on viewpoint and if plastic or animals were visible.

To facilitate the annotation, a small Java-application was made. With the use of several key-strokes, large amounts of image-data could be annotated with relative ease. Even so, the annotation of the data took a considerable amount of time.

In total 20635 images show plastic only, 6972 images show animals only and 8502 images show both. The above (16553) and below (20612) viewpoints were separated in two datasets, to train and test on both separately. In the above-set a total of 14588 images show plastic, 6341 images show animals and 863 images show none of both. In the below-set a total of 14549 images show plastic, 9133 images show animals and 193 images show none of both. Figure 3.6 shows several examples of images in the dataset. Both above and below water viewpoints are shown, as well as the labels the images were annotated with.

Because the images are constructed from films, many images are similar in appearance. This fact should be taken in account when chances of overtfitting occurs.

To evaluate the results of the pipeline, the dataset is divided in a train (70%), validate (10%) and test (20%) set. Because consecutive frames are similar, the division is done by using a pseudo-random randomise algorithm with each time the same seed. Resulting in three sets of images that have a high distribution of the original data while remaining the same on each run. The results of the algorithms can therefore be tested on accuracy with the amount of labels computed correctly.

[2]The videos are online on youtube: `www.youtube.com/007bmac`. Thank you Bill for letting me use your films in this research

Figure 3.6: Several images from the dataset: Left images are from the above water viewpoint and right images are from below water. A red border indicates an image classified as showing plastic, where a green border indicates an image classified as showing animals.

4 Method

This section describes the steps taken in this project. First the pipeline is explained, after which the details of the implementation are stated.

4.1 Pipeline

As stated in section 2 a Convolutional Neural Network has been used as a feature-vector of the image, after which a Support Vector Machine was used for classification.

Several libraries exist that implement a CNN. In this project the choice is made to use Caffe (Jia et al., 2014). This framework is open source and free to use. Moreover it has a large community maintaining and developing, can run both on CPU and GPU and has an interface to `Python`.

The Caffe framework has several pre-trained CNNs. In this project the Caffenet network is used, which is an implementation of Alexnet (Krizhevsky et al., 2012). Caffenet is trained on the *ilsvrc12* dataset, however in contrary to Alexnet, the order of pooling and normalisation is swapped. [citation needed]

The output of Caffenet is a 1×1000 normalised vector, representing the confidence of the respective class. However, the second-to-last layer of the network is used as feature-vector. This layer is a 10×4096 vector; 10 rows because the network uses five overlapping 'sub-images' (four corners and one centre) and their mirrored image, to improve accuracy. Each row of 4096 numbers is an abstract representation of the image, trained by the network. To reduce complexity, the mean over the 'sub-images' is used to compress the second-to-last layer data in a 1×4096 vector.

This 4096 dimensional vector is then used as the space for the SVM. The three possible hyperplanes as described in section 2.2 were tested with several parameters. Tests were conducted by training the SVM on the test-data, and using the validate-data to find the best parameters.

Three different types of hyperplanes were used. A polygonal plane was tested with degrees of 2 till 9. An RBF plane was tested with gammas of 0.0 till 1.0 with steps of 0.1. Lastly a linear plane was tested. Each of the twenty SVM models were trained on part of the train-set, with intervals of order of magnitude; the sizes of the train-set were: 14, 140, 1400 and 14000 images.

4.2 Impelemtation

`Python`has been used to write the code necessary for the pipeline. Not only because Caffe has an interface in `Python`, but also because of the many libraries available in `Python`.

The library that allows the use of matrices and basic linear algebra operations is `numpy`. Another library called `scikit.learn` has an implementation of an SVM, which has been used in this project. The code used in this project can be found in appendix A

All models are trained and tested on a laptop with 2^{nd} generation i5 CPU, 4GB RAM and without using a GPU. Therefore time testing and training took in this project should be evaluated relatively to each other.

Furthermore, because no GPU was used, running Caffenet took a considerable amount of time. Therefore the output of the network had been saved for each image, to speed up the training of the SVM.

4.3 Detecting location of plastic

After testing the pipeline to detect plastic and marine life on images, small adaptations were made to find the location of the classes in the image. When the system developed in this project is used for practical applications, location of the plastic is important.

To distinguish blobs in an image, several techniques could be used; however, as a prove of feasibility, no complex algorithms were used in this project. A new piece of code was written that segments the image in sections and treats each of them as an image in the original pipeline.

The outcome of each segmented piece of the image are cumulated to show the confidence of detection of the classes at the given location in the image.

Because there is not labelled data of these segmented images, the evaluation could not be done on a large scale. Therefore no sizeable experiments could be conducted to test the accuracy of locating of the classes. However, in section 5.2 the outcome of several tests is shown.

5 Results

5.1 Pipeline

The outcome of the different parameters the pipeline was tested with, can be found in appendix B. Several details of the raw output are visualised and shown in this section.

The accuracy shown in this section is calculated by dividing the correct labled images by the total amount of images. Or in formula form:

$$\frac{\#(Outcome_{True}\, and\, Label_{True}) + \#(Outcome_{False}\, and\, Label_{False})}{\#tested\, images}$$

The time that has been taken into account is the difference in *System time* from beginning the training or testing and finishing. Because the data from the CNN was saved on a hard drive, that time is not included in these figures.

Figure 5.7 shows how each of the SVM model performes and what time it took to train and test. All these tests were fitted on the data of the under-water viewpoint. The blue dots represent the accuracy, that can be read out on the left y-axis. The green and red bars represent time taken to train and test respectively, that can be read out on the right y-axis. Each instance on the x-axis is one of the twenty different hyperplanes.

In figure 5.8 the results of the orders of magnitude of the train size are shown when a linear SVM was used. The figure consists of two graps, the left one shows the accuracy on each class individually and the total accuracy, the right one shows the time taken to test and train. The x-axis shows the size of the train-set on a logarithmic scale. In other words, the first column of the first four graphs is plotted in a single image, only using a different set of test-data.

In figure 5.9 there is moreover a graph of a linear SVM tested on several orders of magnitude of train data. The tests in this figure were conducted on a combination of both viewpoints.

5.2 Detecting location of plastic

As stated in section 4.3 the localisation of the plastic within the image could not be evaluated truly. However, several images have been pulled through the pipeline and the results can be seen in figure 5.10. The middle column in the figure shows the tested images. The more red in the figure on the right, the more confidence the program has in detecting image on that location. The images on the left represent the same with the greenness for locating animals. Testing the localisation took a considerable amount of time, because a run through the pipeline could took up to 5 seconds per sub-image. Because these tests were conducted on a pyramid-segmentation

of depth 4, a total amount of 341 images were run through the pipeline per test.

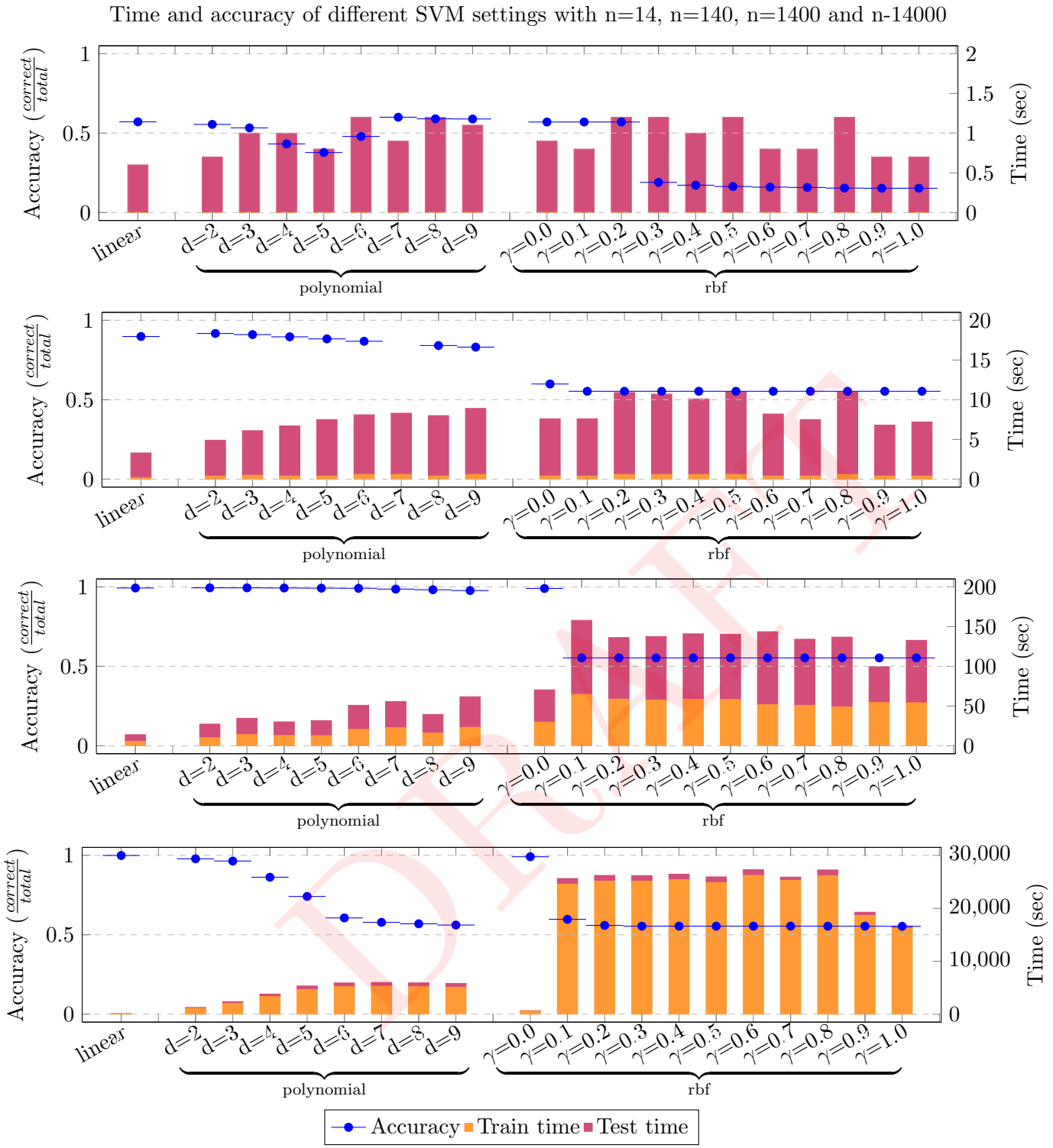

Figure 5.7: Graph of the different parameters of the SVM

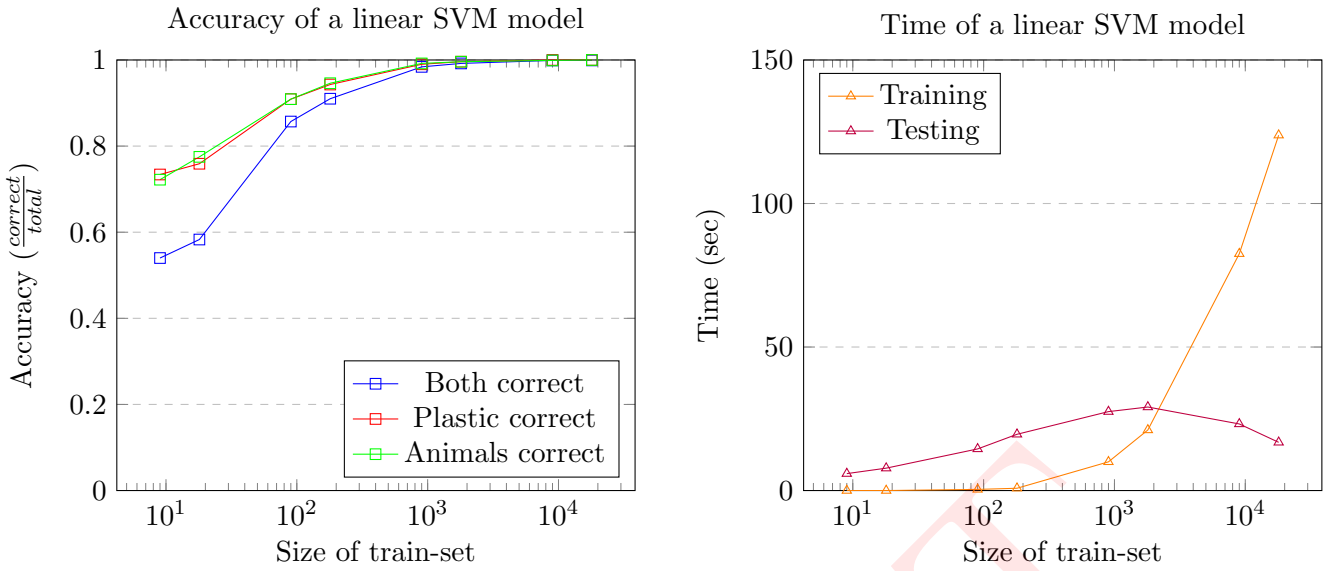

Figure 5.8: Graph of time and accuracy of a linear SVM on different sizes data of the under water viewpoint. The test-data consisted of 2061 images.

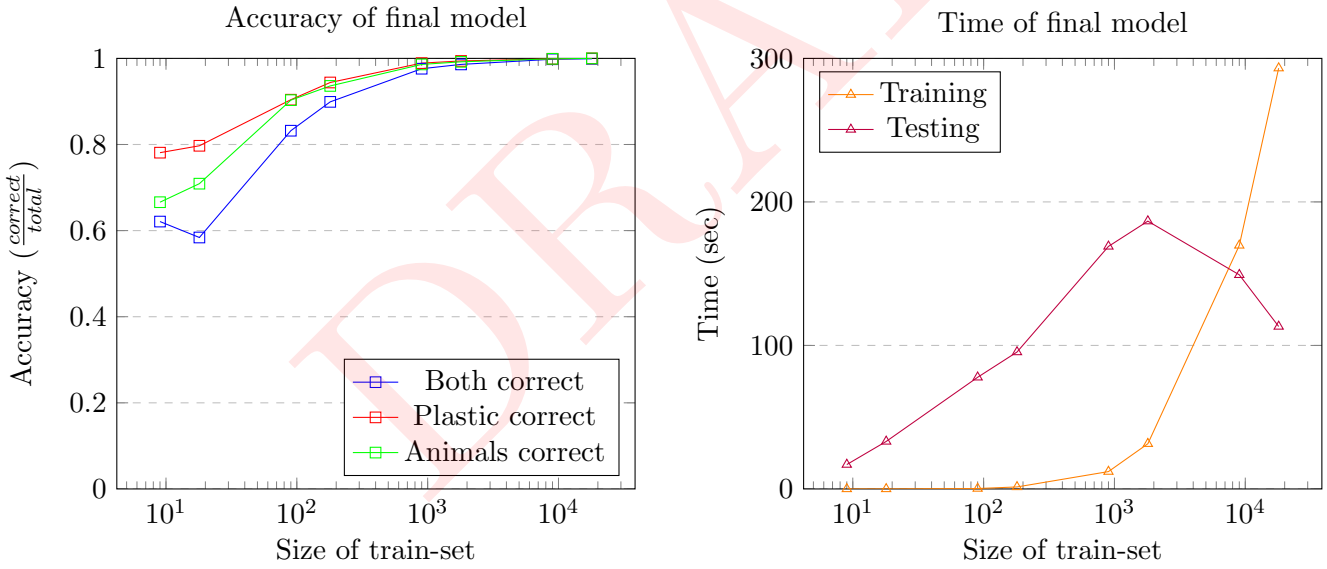

Figure 5.9: Graph of time and accuracy of a linear SVM on different sizes data of the under water viewpoint. The test-data consisted of 18583 images.

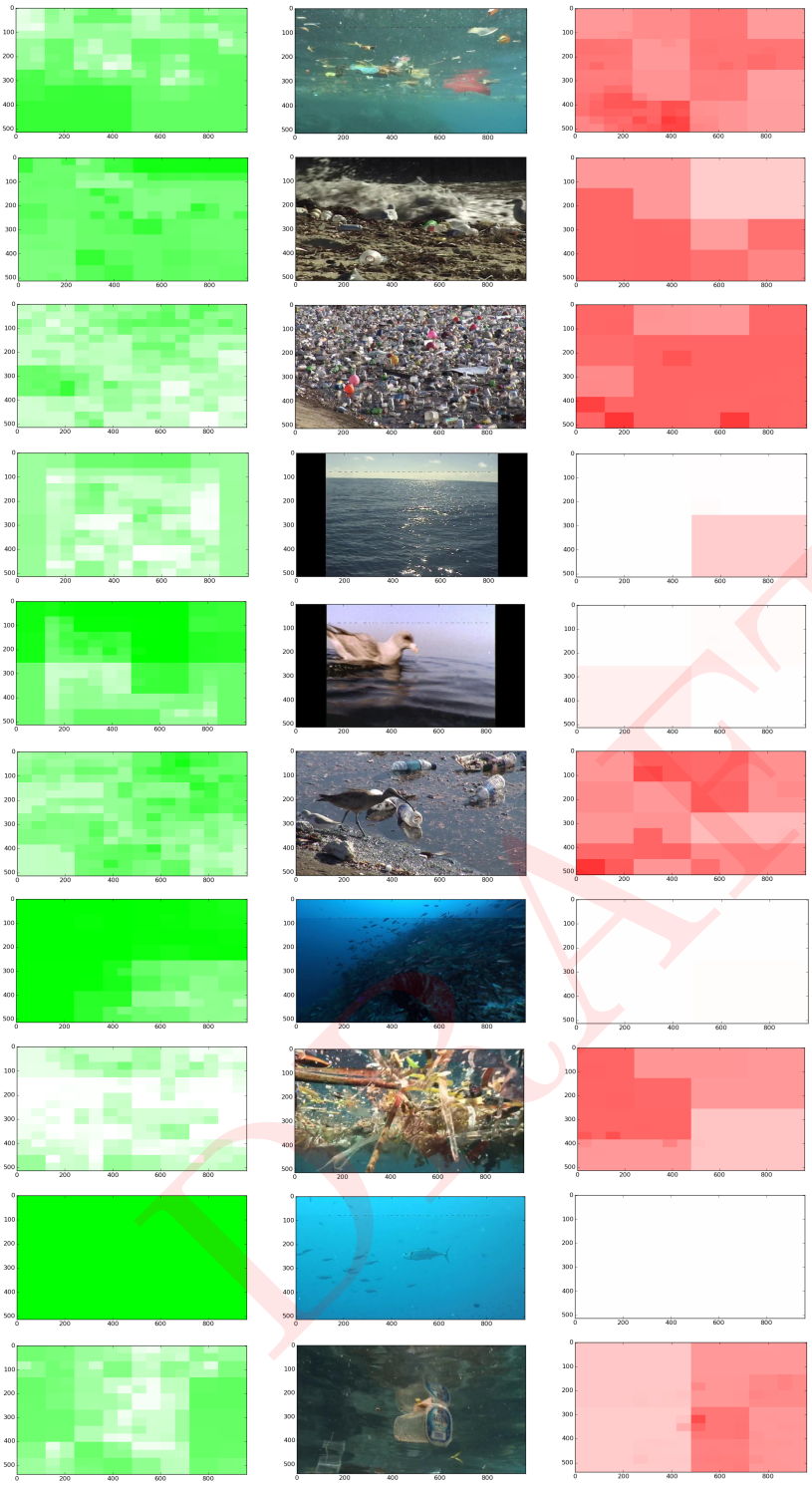

Figure 5.10: Example of outcome of running an image through the segmented image pipeline. Green locates the animals and red the plastic

6 Conclusion

Several conclusions can be made from this project. Firstly, the Support Vector Machine had promising results. The tests of figure 5.7 show that the linear model has one of the highest accuracies of the different hyperplanes, while also having the smallest time to train and test. The RBF model shows less satisfying results. It takes a large amount of time to train and test on the model, while not being able to gain a high accuracy. The polynomial model trained on the train-set of 14k images shows an interesting trent: the higher the degree of the polynome, the lower the accuracy becomes. In other words, for this problem – detecting two classes from a large amount of feature-vectors – the usage of a linear SVM works best.

The test of figure 5.8 shows an accuracy on plastic and animal detection of less than 1‰. However, because the model is trained on 70% the images in the dataset, which consists mostly of consecutive frames of short film clips, there is a substantial amount of overfitting possible.

Therefore, as shown in figure 5.9 another test using the complete dataset from both above and below water viewpoints, shows how the linear SVM performs on different amounts of train data. In this case overfitting when half the dataset is used is also probable, nevertheless, the model also shows accuracy of more than 90% when trained on merely a small part of the train-set. Therefore it can be concluded that this method, using the second-to-last layer of a pre-trained Convolutional Neural Network on an SVM, makes it possible to detect Plastic Soup in images.

Concluding from the localisation of plastic within the images is difficult. The conducted test shows some results. Several of the images show a higher confidance on locations where the consentration of plastic is higher. However, this is not always the case; especialy with the location of marine-life. More tests should be conducted with this technique, with possibly more annotated data, before a conclusion can be made.

Finally the usability of the promising method of using pre-trained CNNs to train on classes that were not in the original CNN is shown in this project. Using a pre-trained CNN as a feature extractor and training another classifier is a simple technique to gain high accuracy while training on a small dataset.

7 Discussion

This project tried to contribute on solving Plastic Soup. The amount of floating plastic in the world's oceans could be very dangerous to the environment and society Moore and Phillips (2011). Automating the process could help with the clean-up, therefore a system that can detect plastic was researched in this project.

A dataset was constructed for this purpose, on which a Convolutional Neural Network was used for feature extraction. On these features a SVM classifier was trained and tested against the labels of the dataset. As shown in section 5, the SVM had a high accuracy, even while trained on small amounts of data.

The accurate results of the Support Vector Machine could be expected. SVMs are known to work well in high dimensional data, as long as the amount of classes to be trained on is small. The 4096 vector of the second-to-last layer of the CNN needed to be trained on two classes, for which both had an SVM trained on ether showing or not.

As also stated before, the high accuracy on the test set could be caused by the similarity of the frames. Nevertheless, as high accuracy also occurs with small amounts of train-data, it is possible to conclude the system truly detects plastic in images.

The localisation however has less satisfying results. The confidence of detecting plastic or animals does not conform consequently with what is shown on the image. This is possibly explained by the fact that the system learned to detect plastic from a distance. In experiments of image classification many pixels described the existence of plastic, while in these localisation experiments, a smaller amount of pixels could be used for classification. Besides that, it is needed in the localisation to view the plastic more close-up. Instead of detecting much plastic together, single plastic objects now need to be recognized.

In this project, several aspects could have been improved for improvement of the results. Firstly the used dataset consisted of many similar frames, which increased the chance of overfitting to this particular dataset. Besides that, the classes that were trained on were not fairly distributed in the dataset. An improved dataset could be used to improve the results of this project. An improved dataset should also contain labels on parts of the images, which could improve the training and testing of the localisation of plastic within the image.

This project did not research the usage of different Convolutional Neural Networks. A standard CNN was used to construct the feature-vector; no time was spent on researching if other networks would perform better.

Not only was assumed that the CNN worked properly that it was not further tested, also the parameters with which the SVM was tested were not statistical analysed; no cross-fold validation was conducted.

Obviously, further research in this domain could correct the mistakes made in this project. A better dataset and more cross-validation could improve the localisation of detecting plastic. Even so, training the system to recognise plastic on several scales could increase the performance on localisation.

Other manners to improve the localisation of the plastic within the images, are the use of other algorithms. If more complex algorithms are used for object detection in the image, the blobs resulting from those could be input for the pipeline, instead of the now used segmentation-pyramid. One of the algorithms that could be used for this is BING [citation needed] ; this project did not have the time to implement one of these algorithms.

Before the system proposed in this project could be used for real-world applications, more research is needed. However, this project showed the possibility of using state-of-the-art Computer Vision techniques to detect Plastic Soup. More research building on the results of this project could result in applications that can detect Plastic Soup.

This project is one of the many recent projects that show the possibilities of Convolutional Neural Networks. Besides that, it also shows how the Artificial Intelligence can be used to help solving environmental problems. I do not think the hippie movement in the '60s could imagine that AI could solve their problems for them today.

References

D. Barnes and P. Milner. Drifting plastic and its consequences for sessile organism dispersal in the atlantic ocean. *Marine Biology*, 146(4):815–825, 2005.

K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

C. C. Moore and C. Phillips. Plastic ocean. *How a sea captain's chance discovery launched a determined quest to save the oceans. Avery, New York*, 2011.

A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014*, pages 818–833. Springer, 2014.

A `Python` code of the project

B Output of `Python` code

B.i SVM