Underwater Pipeline Damage Detection with Pre-Trained Convolutional Network

Jiajun Shi, Wenjie Yin, Yipai Du

KTH, School of Electrical Engineering and Computer Science

Abstract. In recent years, image-based detection technologies have been developed and widely applied in various industries. Damage detection on underwater pipeline is one of the meaningful applications. In this project, we implemented an image-based damage detection system using transfer learning with models including VGG16 and MobileNet pre-trained on ImageNet in Keras. We tested different transfer learning techniques: (1) retrain fully-connected layers and fine-tune the last convolutional block; (2) use pre-trained network as feature extractor with normal SVM or one-class SVM. We also tested a shallow convolutional network for reference. All models were evaluated on the same non-public dataset, with additional visualization techniques for analysis. Most practically promising results were achieved by utilizing MobileNet model. Based on it, a simple demo program is established at the end, to illustrate a standard workflow using our network.

Keywords: Damage Detection, Image Classification, Transfer Learning, Convolutional Networks, MobileNet-V1, VGGNet, SVM

1 Introduction

Underwater pipeline plays a vital role in transportation, communications and other fields. In the complex underwater environment, concrete pipes are vulnerable to corrosion and impact. Crack on the underwater pipeline surface is one of the most obvious indications of structural damage, thus detecting it is critical for the maintenance. Currently, regular manual inspection of underwater pipelines is an important and necessary part of pipeline maintenance work [1]. Nevertheless, due to the rarity of photos containing crack as well as their high variety and low visibility, manual inspection is a huge and difficult task. Moreover, manual inspection methods generally lack objectivity in quantitative analysis as they highly rely on the knowledge and experience of the inspector[2]. Therefore, automatic damage detection methods, especially image-based ones, are proposed as replacement. Latest image processing technique can provide higher efficiency and also higher accuracy compared to manual inspection approaches. With the advancement of digital cameras' resolution, the results on image-based detection can be further improved in the future [3].

Over the years, literatures present various kinds of techniques to automatically identify the damage on the surface, and its depth using image processing techniques. In view of the recent extensive success of deep-learning technologies in different tasks of classification and detection, it is practical and promising to apply deep-learning based method in underwater pipe damage detection problem. In this project, various tests were done on the problem of underwater pipeline crack detection, using different deep-learning models and techniques. Finally a practically useful demo is illustrated here. Although we used a specific non-public dataset, still such tests can provide some deeper understanding on the use of deep-learning methods for general underwater crack detection problem.

2 Background

In the current state of practice, damage detection approaches can be usually divided into two categories: destructive testing method and non-destructive testing incorporating with the visual testing [2]. Traditionally, due to the limitation of sensors and processing technologies, infrared, thermal, ultrasonic and laser based methods drew more research interest[4]. In recent years, with the development of camera and image-based algorithms, there has been a rapidly ascending trend for general image-based crack detection. A wide range of image-based methods have been used for practical crack detection. Among existing approaches, many are based on classic computer vision technologies such as detection by various filtering [5–7], morphology-based method [8] or approach using topological structure [9]. Compared with them, more recent deep-learning methods like the convolutional neural network proposed in [10], show promising results. With the help of deep models, highly discriminative deep features can be learned from raw images by low-cost camera, thus avoiding hand-crafted filter or descriptor structures and auxiliary assumptions in them. While general crack detection problem has been explored deeply, researches or applications with underwater images are relatively few. Though underwater crack is physically similar to general crack, yet compared to problems with normal image, under water pipeline damage detection is more challenging, mainly due to the following problems:

1) Extremely imbalanced dataset. In the real-world, collecting large amounts of underwater images is often difficult and can be time-consuming and resource intensive. And among all images, those containing clear pipe damage is even rarer. Specific numbers of image samples in our dataset, are listed in Section 3.1.

2) Distortion of color and contrast of images. Because of the attenuation of natural light and non-uniform artificial illumination, it can be hard to recognize cracks in underwater images.

3) Difference between positive and negative samples can be so subtle that it is hardly recognizable even with human perception.

In the limited number of researches on underwater crack detection, [11] proposed a detection method based on local and global characteristics of image blocks and domains. [12] introduced a two-dimensional lateral inhibitory network and a border highlighting rule for contrast enhancement and based on them an improved artificial bee colony algorithm for edge extraction. Both schemes above still use classic hand-crafted descriptors and rules. Therefore, our work of applying deep models on underwater crack detection, has its research contribution in the sense of exploration.

3 Approach

The process of image-based damage detection generally comprises 3 steps:

1) Image Acquisition. Collect the image samples of the structure through camera, laser or other sensors.

2) Pre-processing. After acquisition, images are pre-processed with techniques like segmentation, sharpening, contrast enhancement or even manual selection[1]. Furthermore, data augmentation are employed to enlarge training dataset and reduce overfitting.

3) Training and predicting. With the image samples processed by the previous procedure, classifiers can be trained using different models. Then the classifier verified by validation data, is applied on pre-processed samples of new data to detect the crack, or even estimate the condition of the structures.

3.1 Dataset

The dataset we use for this project is an underwater image pack for outer surface of concrete pipelines photographed by underwater vehicle. It is provided by a company in collaboration with RPL, KTH. As Fig. 1 to Fig. 4 shows, most images were taken with the pipe aligned height-wise and seabed as background. According to the original taxonomy from the company, there are 4 main categories: (1) clean pipe line[2]; (2) connection, typically one or two straight thin connection seams; (3) anode, an attached cuboid-like structure on the pipe; (4) damage, pipelines with crack, regional defect etc. In the project, same categorization is used and the ideal goal is to distinguish each type. The sizes of sub-datasets (1)-(4) are 6383, 1299, 3140, 83. Provided that the images were taken while the camera slowly moved along the pipe, the number of actually unique shots of the datasets can be much smaller than their size. Thus this is an extremely imbalanced dataset, especially w.r.t. the number of damage samples. As the images were photographed under different water quality and light conditions, the color distortion is severe. For instance, some images can be blue or yellow as a whole while the original color diversity among objects is lost.

[1] For example, we filtered out images with too many irrelevant items like marine organism.

[2] "clean" in the sense of no extra structures or crack, but probably with marine organism or debris

Fig. 1. clean pipe

Fig. 2. damage

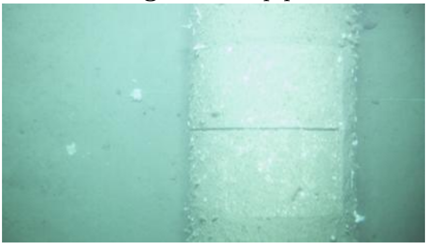


Fig. 3. connection

Fig. 4. anode

3.2 Preprocessing

We processed each image in the dataset in the following sequence respectively.

3.2.1 Sorting

In the provided image packs, many images are actually from closely consecutive shots, which means there are many overlaps among images. To avoid inaccurate assessment, the sets of test, training and validation are divided before cropping and overlapping between any two sets is avoided as much as possible by manual selection. This operation is proved to be necessary as our previous tests done without it showed overly nice performance, compared to our final tests.

3.2.2 Simple Segmentation

Because the classification is aimed at pipelines, the background besides pipe is irrelevant. Therefore, a crude method is designed for segmenting the pipes. Utilizing the fact that pipes in all images are aligned height-wise, our method simply detects the two strongest local minima[3] of height-wise averaged grayscale curve. Note that gradient magnitude is not used, simply because methods using it show no better result. Since the main topic is not segmentation, this practically acceptable method is used throughout the project and all following procedures mentioned are operated on segmented results.

[3] In Fig.5 it looks like maxima because of the reversed y axis for image

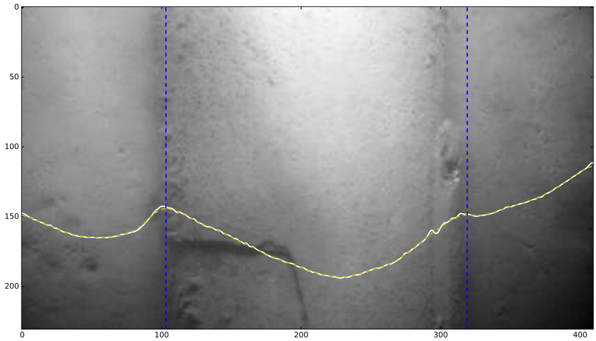


Fig. 5. Example of segmentation. White line: averaged grayscale; blue lines: segmenting lines

3.2.3 RGB to Gray

As mentioned in section 3.1, the images bear the problem of color distortion, so it can become an extra disturbance in the classification process, especially when such information is largely distorted. Therefore, all samples originally in RGB are preprocessed to be gray-scale images. Intuitively, color is not the essential feature that distinguishes the four classes thus it is expected that the loss of effective information is trivial.

3.2.4 Sharpening

Due to underwater conditions, the outline of some damaged pipes and connection parts are vague, which are even difficult for human to notice. To enhance the relatively vague curves of connection and damaged samples, the images are moderately sharpened, making the structures of interest clearer.

3.2.5 Cropping

A fundamental pre-processing step in this project is cropping, with two main motivations. Firstly, considering that in images of connection, anode or damage, the unique pattern merely occupies a small part of the pipe while the main region is the same among all types, we applied cropping only on region of interest to filter out the large amount of common information in different categories. Yet this common information is still preserved, in the samples from clean pipe category. Secondly, With as few as 83 samples for the damage dataset, applying cropping can help to raise the amount of data to an acceptable level for training of several layers in a large network, e.g. two fully connected layers in VGG16 model [13]. During the process, over-sampling is applied on the damage samples, by cropping similar region with different geometric transformations. Meanwhile datasets of type (1)-(3) are down-sampled, with cropped samples from not all but part of the original images that were selected randomly, with their characteristic features.

3.2.6 Augmentation

In Keras, the augmentation functionality is already implemented so for all models used with Keras augmentation is applied. Typically chosen operations include rotation, shift, flip and zoom.

4 Experiment

4.1 Models

4.1.1 VGG16

In many transfer learning applications, VGG16 is a popular choice for its good general performance and simple structure. In this project we adopted a VGG16 model which has been pre-trained on the ImageNet data and implemented using the Keras open source API. VGG16 model has 41 layers in total and 16 layers with learnable weights: 13 convolutional layers, and 3 fully connected layers. The detailed structure of VGG16 is illustrated by Fig. 6. To preserve the pre-trained weights, in the training process only the fully connected layers were reset and re-trained. More specifically, we did four types of experiments on VGG16:

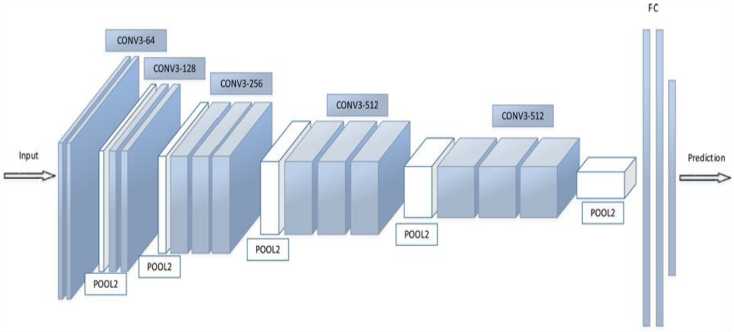


Fig. 6. The model structure of VGG16, with 5 blocks of convolutional layers and 3 fully connected layers[14]

- With all convolutional layers frozen, we only re-initialize the fully connected layers with various settings. Such operation can be viewed as using VGG16 as feature extractor and adding an normal shallow ANN as classifier. The new fully connected layers were trained with our datasets after pre-processing with augmentations.
- Besides retraining the fully-connected layers, adjusting the last convolutional layers block (last three gray blocks in Fig. 6) could possibly improve the results. Thus with a model trained with procedure above, we fine-tuned the last convolutional block with our dataset. We started by same procedure above, and only afterwards fine-tuned the weights of the previously frozen last convolutional block with fully connected layers. This is to avoid initial training of the newly added fully connected layer having large influence on the convolutional layer. Fine-tuning the last convolutional block can keep the general features and at the same time add more peculiar features about specific problem through training.
- As mentioned above, we could use VGG16 as feature extractor, then the final classifier could be SVM instead of normal ANN. Since the amount of our dataset is small and imbalanced, SVM which usually works fine with hundreds of training samples is expected to outperform the original fully connected layers structure.
- To deal with the imbalance problem of the dataset, we tested structure of VGG16 as feature extractor with one-class SVM [15], using RBF kernel. With one-class SVM there is no need for data from damaged class. We could train three one-class classifiers respectively for clean, anode and connection. By exclusion, suspected samples of damaged class can be detected.

4.1.2 Shallow Network

We manually designed a shallow 3-layer convolutional network with 2 fully connected layers for classification of the 4 different classes. The structure is shown in Fig. 7 with cross-entropy loss function. Notice that we do not expect good performance on shallow networks because given the dataset is so small, training from scratch would be difficult. Rather, we would prefer fine-tune previous complicated models so that the first several layers for general feature extraction could be reused. Note that given the dataset is small with only several different damages, we could anticipate overfitting problems. Hence even if the testing result is promising, we cannot trust it before further understanding about the network is obtained.

4.1.3 MobileNet V1

During our tests of shallow models, one observation was that our classification problem could potentially be solved by more lightweight models. Therefore, MobileNet[16], as a class of efficient models for mobile and embedded vision applications, was applied and tested. MobileNet is based on a streamlined architecture that uses depth-wise separable convolutions for better computational efficiency. Besides, two parameters, i.e. width multiplier and resolution multiplier, are used

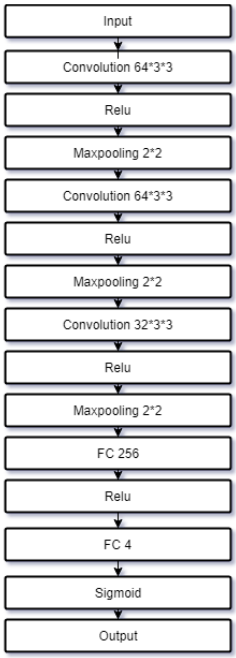


Fig. 7. The model structure of shallow model, with 3 convoluntional layers and 2 fully-connected layers

for further trade-off between efficiency and accuracy. Compared to VGG16 (138 million parameters), MobileNet contains much less parameters (4.2 million, in the model we used).

5 Analysis

5.1 Model Performance Comparison

5.1.1 VGG16

- **Small scale**
  With fine-tune of VGG16, we can reach positive accuracy of 97.81% and negative accuracy of 87.84% for binary classification test. We do this with positive as damage + anode + connection and negative as clean pipe. This is to show if we enrich the damaged data by similar images what can best positive accuracy be through direct transfer learning of VGG16. Notice that this can only be done with small cropped images because the positive data

only resemble locally. From this, we can see that indeed the distinction between clean and not clean groups are very obvious so that by some further techniques, we should be able to accomplish the task successfully.

- **Large scale**
  With fine-tune of VGG16 on large image, we checked the ability for classification of the four classes directly. The results show imbalanced accuracy: clean-87% , anode-80% , connection-84%, damage-60% .
- **SVM**
  With convolutional layers as feature extractor and SVM for binary classification, we compared the results for different datasets (all in small cropped images).
  1. Anode+Connection+Damage **VS** Clean
     With RBF kernel the accuracy goes extremely low. With a linear kernel, the overall accuracy is around 85%-90% but very imbalanced if not weighted properly, like one accuracy 99% and the other 50%. But how to weight the samples from imbalanced data is only based on trial and test without any systematic approach. With additive χ^2 kernel the accuracy is balanced with both around 90% to 95%.
  2. Different combinations
     Table 1 shows the accuracy and observations of different combinations of dataset using SVM. We can see that to differentiate the damage with other classes, the most severe problem is to deal with imbalanced results. We can hardly get a balanced and good accuracy.

Table 1. Accuracy for different combinations of dataset.

+	-	Result
Damage	Clean	Around 80% - 95%, but two classes not high accuracy simultaneously
Damage	Anode	Around 60% - 80% for both classes. When one class gets 90% the other gets worse than 50%. Augmentation helps to improve the weaker class acc. Normalization improves too.
Damage	Connection	Best result damage accuracy 79%, connection accuracy 89%. Usually highly imbalanced result.

- **One-class SVM** Another option to use VGG16 model as feature extractor, is to replace the previous fully-connected layers with an one-class SVM classifier (of RBF kernel in our tests). This is applicable in our problem because the main goal is to detect the images with damaged pipes, rather than achieving best performance on classification of all classes. Provided that MobileNet model turned out to be of better overall accuracy, the tests on one-class SVM were not thoroughly done. As table 2 shows, we did tests on one-class SVM for the three classes except clean pipe, considering that clean pipe class can be distinguished relatively well by our VGG16 model. In table 2, the first column lists the training dataset of the one-class SVM model. The one-class classifier only requires samples from the "positive" class, so there are in total three classifiers, one for each class. The three test datasets are independent from the training datasets. When classifiers of anode and connection were trained, training dataset of damage was utilized as validation to select a model that gave high accuracy on distinguishing damage samples. The one-class classifier using damage dataset gives a true positive (the rate of true damage samples being detected) of 0.879, while the intersections of negative predictions of classifiers trained on anode and connection datasets gives true positive of 0.907. In view of one-class SVM's less demand on training dataset as well as acceptable true positive rate on damage class, it can be a potential solution when training deep models on small dataset does not provide a good performance.

Table 2. Accuracy for one-class SVM classifier

Training Data	Damage Test	Anode Test	Connection Test
Damage	**0.879**	0.906	0.917
Anode	0.921	**0.774**	0.946
Connection	0.968	0.813	**0.723**

5.1.2 MobileNet One of the crucial hyper parameters of MobileNet is width multiplier α, which is used for reducing the number of channels and computational cost proportionally.

In our experiments, with all layers frozen except the fully-connected layers, we tried 3 typical settings of α: 1, 0.75 and 0.5. The results of large scale images are shown in Table 3. The average classification accuracies of these models are similar. We can infer that it is enough to classify these pipes with the most lightweight MobileNet. As for small scale images, with fine-tune of MobileNet-V1, α=0.5, we can reach positive accuracy 94.16% and negative accuracy of 90.02% for binary classification test. Besides high accuracy, the number of parameters is reduced significantly [16]. For an application-oriented research, both accuracy and efficiency are significant. Thus MobileNet is the most suitable among tested models for our object.

5.1.3 Shallow Network The shallow network from scratch is also able to learn some patterns from the data, although the result is far worse than MobileNet. Notice that although the network is shallow, because we are training it from scratch, the number of trainable parameters is still no less than those for fine-tuning a deep network. Hence in training small datasets, overfitting problem still needs to be taken into consideration.

Table 3. Accuracy showing the effect of α of MobileNet

α	Accuracy				
	clean	anode	connection	damaged	average
1.0	0.95	0.95	0.92	0.91	0.93
0.75	0.94	0.98	0.93	0.85	0.91
0.5	0.95	0.98	0.87	0.87	0.92

In Table 4 some results for different networks and datasets are shown. Notice that for all the datasets we used the same images for testing. The only difference is that small portion of different images are taken out of the training data to form a validation dataset. Hence ideally this should not change much about the performance of the network. As can be seen in Table 4, using the same settings while only changing from dataset #1 to #2 changed the performance a lot. But the performance for MobileNet changed little compared to Table 3. We can see from Table 4 that shallow network can in general learn something from the training data but not enough for 4-class classification and the result is not stable, meaning changing something in the dataset changes the performance a lot. The imbalance in result can change to another dataset. This instability can be explained by overfitting because we are only using small dataset to train from scratch. Moreover using dropout as can be seen in Table 4 is good for improving the results. The conclusion is that shallow network is not able to accomplish the task but we can see the network is able to learn something from the dataset and at least give some inference about the classification.

Table 4. Shallow network accuracy

Network	Data Set #	Accuracy				
		clean	anode	connection	damaged	average
shallow	1	0.77	0.94	0.92	0.62	0.81
shallow (without dropout)	1	0.59	0.81	0.87	0.98	0.81
shallow	2	0.51	0.78	0.87	0.98	0.79
shallow	3	0.90	0.95	0.93	0.63	0.85
MobileNet	3	0.99	0.95	0.90	0.90	0.94
shallow (without dropout)	3	0.92	0.90	0.94	0.21	0.74

5.2 Understanding Through Visualization

As exhibited above, the accuracies of MobileNet and VGG16 differ significantly, but meanwhile various existing benchmark tests do no show a salient advantage of MobileNet over VGG16 [17, 16, 18, 19]. Therefore, we used the approach proposed in [20] to further verify the classification result. For image to be classified, we occluded part of the image with a square patch and applied classifier as on normal images. However, applying pure gray patch as in [20] failed on our datasets as with occlusion the class predicted was always altered wherever it was applied. Most probable reasons should be that a small region with inconsistent color and sharp edges was similar to originally existing structure like connection, anode or crack. Thus instead, in our tests the region was occluded by a strongly smoothed version of the original patch, as Fig.8. Thus we minimized the extra information introduced by the occlusion while removing the structure information under the patch. Through the probability of correct class, we could examine if the classifier truly found the typical pattern of the correct class. For each image, the occlusion patch scanned through the whole image with stride 5. The selected examples include general case in result as well as particularly interesting individual.

In Figures 9 to 15, sub-figure (a) shows the image after pre-processing. Subfigure (b) shows the heat map of probability of correct class prediction, where cooler color of one cell indicates that the classifier is predicting correct class with less confidence, with the region around that cell occluded. For tests in which the occlusion alters the prediction result, an extra image shows the predicted class after occlusion.

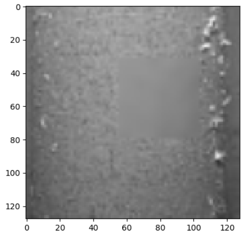
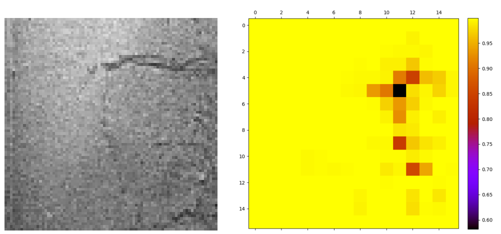


Fig. 8. Example of occlusion patch

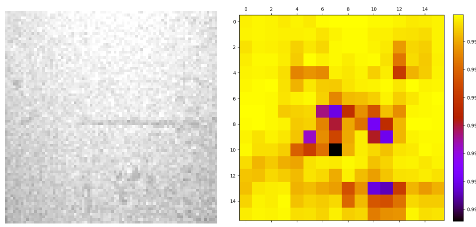
Above all, Fig.9, 11, Fig.12, Fig.15 respectively show the general cases in the class of damaged, anode and connection. From them a salient correspondence can be observed between low probability region in (b) sub-figures and typical structure of the class in the (a) sub-figures. Such correlation validates the ability of the classifier to appropriately base its judgment on the distinct pattern. Similar correspondence are obtained in general data samples in which structures are clear and irrelevant items are few, leading to a high accuracy as shown in Section 4.2. However, in samples with low image quality like Fig.10 and Fig.14, correspondence becomes vague, suggesting that classifier is identifying class by correct structure (or the structure we human use to distinguish the classes). Moreover, Fig.13 exhibits a wrong correspondence, between class anode and a small chunk of marine organism, instead of true anode on the right. A conclusion can be drawn on tests that the network has learned the distinguishing structure for classification, but not robust enough against change of image quality or rare irrelevant items. To a degree this is due to the lack of data and high homogeneity among samples of connection, anode and clean pipe. And by tests, we noted that this was not solved by adding stronger and more diverse augmentation on the training samples. Another interesting observation was that when part of anode or connection structure was occluded, the classifier always gave damaged as prediction. Provided that the training samples of damaged class are much fewer but more heterogeneous, the explanation could be that the anode and connection classes were overfit while the structure learned on damaged class was too broad because of the diversity of its training samples.



(a) (b)

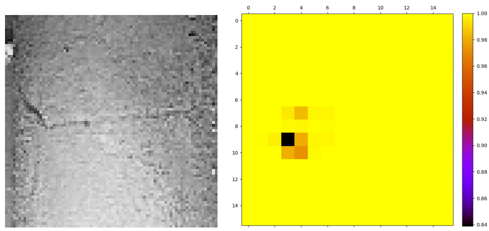
Fig. 9. Example 1 - Damaged

In addition, we also applied same visualization technique on the VGG16 classifier trained on our datasets (with only fully-connected layers retrained). For easy comparison, the same images were used for tests on VGG16 model. In Figure 16, same image as Figure 9, classifier focused on some region of damaged structure, but also much irrelevant part. Figure 17 shows a peculiar pattern widespread in tests, that occluding lower parts of the image leads to recognition of clean pipe class. This might indicate the characteristics of clean class was incorrectly learned, or possible problems remaining in the occlusion technique. Similar problems for anode and connection class are shown by Figures 18 and 19, where correct region of interest was covered but irrelevant part also impact the classification and unreasonable recognition of clean class was obvious. In all, based on visualization result, the lower accuracy of VGG16 should be partially



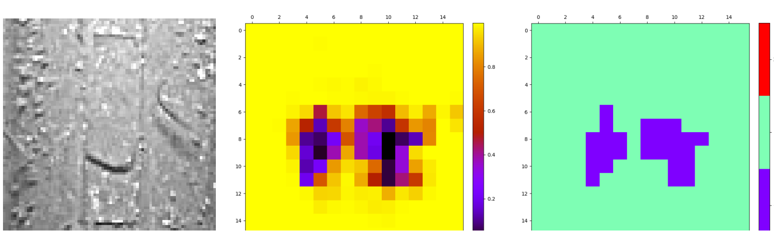
(a) (b)

Fig. 10. Example 2 - Damaged



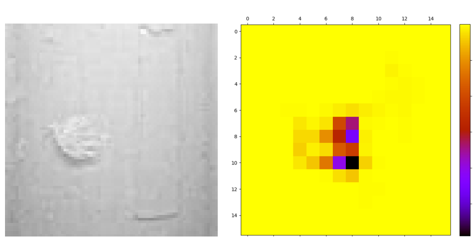
(a) (b)

Fig. 11. Example 3 - Damaged



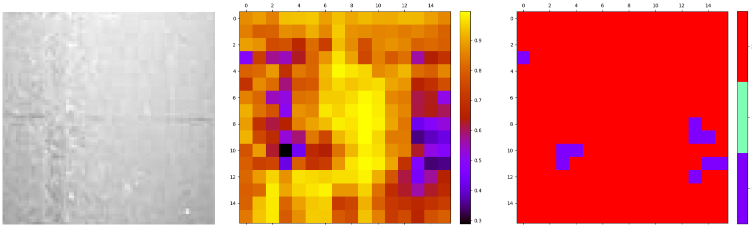
(a) (b)

Fig. 12. Example 4 - Anode



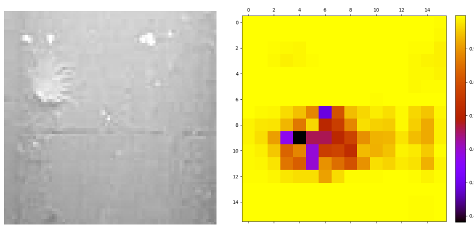
(a) (b)

Fig. 13. Example 5 - Anode



(a) (b)

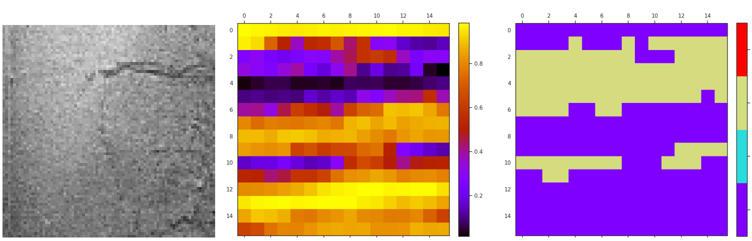
Fig. 14. Example 6 - Connection



(a) (b)

Fig. 15. Example 7 - Connection

related to its weaker focus on correct structure and more confusing information learned.

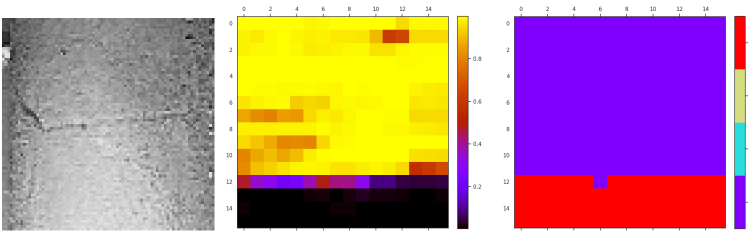


(a) (b)

Fig. 16. Example 8 - Damaged - VGG16

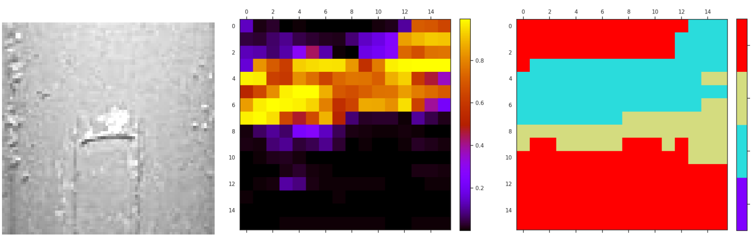
6 Demo

To illustrate the performance and feasibility of our project, we designed a demo program. The demo program classifies both large image of whole pipe(to check



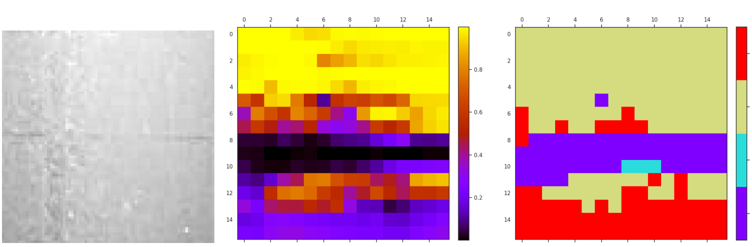
(a) (b)

Fig. 17. Example 9 - Damaged - VGG16



(a) (b)

Fig. 18. Example 10 - Anode - VGG16



(a) (b)

Fig. 19. Example 11 - Connection - VGG16

if it is a damaged) and small image of crops(to localize the damage), using two classifiers trained with MobileNet model for large and small scales. Segmentation together with image enhancement is done on all the images to remove seabed, preserve pipes and increase the contrast. The first classifier is trained on all these large images to classify the four classes, i.e., clean, anode, connection and damaged. Then the images that are suspected to be damaged are cropped into smaller parts. Connection and anode images are combined with damaged images with the hope to enrich the class dataset. Since those three classes look very similar locally, we simplified the problem to be just dividing clean and not clean crops. With the two classifiers trained, we are ready to run the demo program.

As Fig.20 shows, the demo starts with pre-processing including segmentation and sharpening on the original image. Then we classify which class this image belongs to, clean, anode, connection. If the image is no damage to four classed with the large scale classifier. If the result is damaged, then crop the original image to small scale images and pass each of them to the small scale classifier to locate the damaged position. Some example results are shown in Fig.21, in which the damaged images were classified and the damaged regions are further spotted.

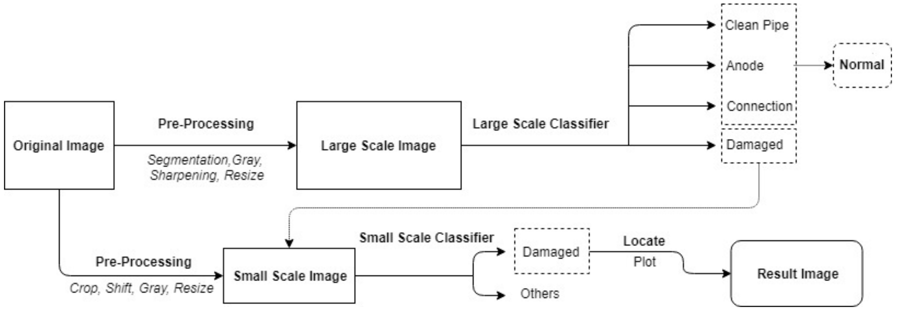


Fig. 20. The flowchart of the demo procedure

7 Discussion

7.1 MobileNet vs. VGG16

Both the test accuracy and visualization experiments above show that MobileNet is superior to VGG16 in our detection problem. We have not come to a concrete conclusion about the explanation for this. One hypothesis is that with our limited datasets, a smaller model bears less risk of overfitting. Although the difference of entire model parameter is significant between MobileNet and VGG16, yet in our application using pre-trained model, the main divide lies on that VGG16 comes with 2 fully-connected layers and 1 output layer while in MobileNet the

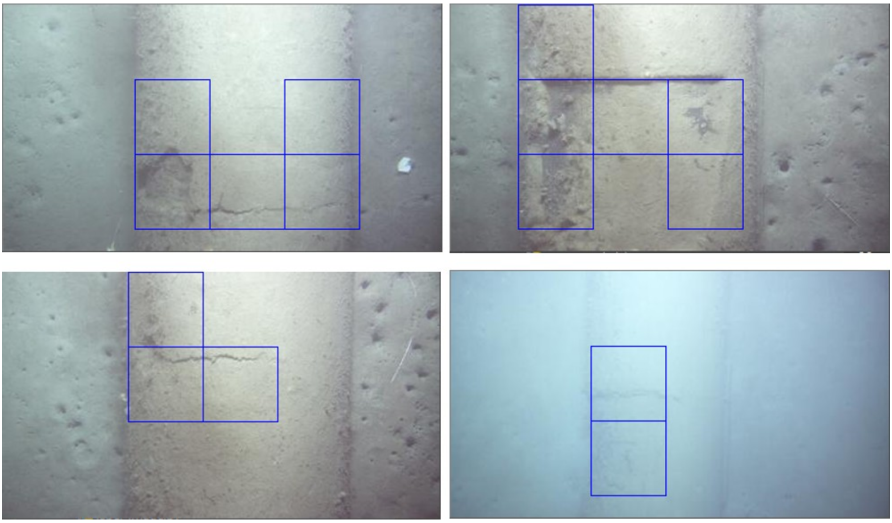


Fig. 21. The Results of the demo, the damaged positions are marked with blue box

setting is 1 fully-connected layer and 1 output layer. This means the number of parameters differ by three order of magnitude, since we typically used 1024 as neuron number in each fully-connected layer.

7.2 Trade-off on Cropping

At the beginning of this project, we planned to counter the problem of imbalance among classes and lack of data, through cropping. The purpose is to acquire more samples for the smallest class damaged. With small cropped images and additional augmentation, we created much larger dataset of distinct samples. This helped to reduce the problem of class imbalance. Clean crops could be distinguished from other crops with high accuracy. However, through cropping much large-scale information was lost, thus the difference among anode, connection and damaged classes vanished. It was a trade-off between data amount and information completeness, but later with MobileNet model classifier was successfully trained on small datasets of large images, making it less a dilemma. Therefore, in the final demo procedure, the main classification of four classes is conducted by a classifier on large scale and the classifier for crops is only used for spotting suspected damaged region.

7.3 Imbalance Problem

At a first glance the imbalance problem is severe. As is introduced in Section 3.1, we have only 83 samples of damaged pipe. However, through proper preprocessing, including image transformations to up-sample the damaged pipes to around 800 and down-sample the other classes by taking random samples, we have achieved reasonable results without too much overfitting problem. Hence, we can see in this problem the characters between different classes are relatively easy to be distinguished, without the need of a very large database of images.

8 Conclusion

In this project, we successfully accomplished the task to identify damage of underwater pipes using limited and imbalanced dataset. We used image preprocessing and enhancement to firstly create approximately balanced data and using that to test different network architectures. The best result is obtained using MobileNet and the lesson we learned is for doing the job we don't require a very deep network. Furthermore, we used visualization to see whether the network is learning properly, that is, if it is learning the distinctions we human beings see for classification. Finally, we established a demo program that illustrates the damage detection process which can greatly reduce the burden of manually inspection.

Acknowledgement

Thanks to Professor Josephine Sullivan for guidance in and after class.
Thanks to teaching assistant Taras-Svitozar Kucherenko and Yue Liu for helpful suggestions.
Thanks to Professor John Folkesson for the datasets and general advices.
Our VGG16 reference code is:
https://github.com/Arsey/keras-transfer-learning-for-oxford102
The MobileNet reproduce reference is:
https://www.tensorflow.org/hub/tutorials/image_retraining

References

1. Alvaro Fuentes, Sook Yoon, S.C.K., Park, D.S.: A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. (2017)
2. Arun Mohan, S.P.: Crack detection using image processing: A critical review and analysis. Alexandria Engineering Journal (2016)
3. Debejyo Chakraborty, Narayan Kovvali, B.C.A.P.S.A.: Structural damage detection with insufficient data using transfer learning techniques. (2011)
4. Fujita, Y., Hamamoto, Y.: A robust automatic crack detection method from noisy concrete surfaces. Machine Vision and Applications **22**(2) (2011) 245–254
5. Salman, M., Mathavan, S., Kamal, K., Rahman, M.: Pavement crack detection using the gabor filter. In: Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on, IEEE (2013) 2039–2044
6. Sinha, S.K., Fieguth, P.W.: Automated detection of cracks in buried concrete pipe images. Automation in Construction **15**(1) (2006) 58–72
7. Talab, A.M.A., Huang, Z., Xi, F., HaiMing, L.: Detection crack in image using otsu method and multiple filtering in image processing techniques. Optik-International Journal for Light and Electron Optics **127**(3) (2016) 1030–1033
8. Iyer, S., Sinha, S.K.: A robust approach for automatic detection and segmentation of cracks in underground pipeline images. Image and Vision Computing **23**(10) (2005) 921–933
9. Nguyen, H.N., Kam, T.Y., Cheng, P.Y.: An automatic approach for accurate edge detection of concrete crack utilizing 2d geometric features of crack. Journal of Signal Processing Systems **77**(3) (2014) 221–240
10. Zhang, L., Yang, F., Zhang, Y.D., Zhu, Y.J.: Road crack detection using deep convolutional neural network. In: Image Processing (ICIP), 2016 IEEE International Conference on, IEEE (2016) 3708–3712
11. Shi, P., Fan, X., Ni, J., Wang, G.: A detection and classification approach for underwater dam cracks. Structural Health Monitoring **15**(5) (2016) 541–554
12. Zhang, Z., Fan, X., Xie, Y., Xu, H.: An edge detection method based artificial bee colony for underwater dam crack image. In: Biomedical Imaging and Sensing Conference. Volume 10711., International Society for Optics and Photonics (2018) 107111R
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
14. Hachim El Khiyari, H.W.: Face recognition across time lapse using convolutional neural networks. Journal of Information Security (2016)
15. Manevitz, L.M., Yousef, M.: One-class svms for document classification. Journal of machine Learning research **2**(Dec) (2001) 139–154
16. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
17. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al.: Speed/accuracy trade-offs for modern convolutional object detectors. In: IEEE CVPR. Volume 4. (2017)
18. Yadav, N., Binay, U.: Comparative study of object detection algorithms. (2017)
19. Marchwica, P., Jamieson, M., Siva, P.: An evaluation of deep cnn baselines for scene-independent person re-identification. arXiv preprint arXiv:1805.06086 (2018)
20. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision, Springer (2014) 818–833