Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Master's Thesis

Sonar Patch Matching via Deep Learning

Arka Mallick

Submitted to Hochschule Bonn-Rhein-Sieg,
Department of Computer Science
in partial fullfilment of the requirements for the degree
of Master of Science in Autonomous Systems

Supervised by

Prof. Dr. Paul G. Plöger
Prof. Dr. Gerhard K. Kraetzschmar
Dr. Matias Valdenegro-torro

January 2019

I, the undersigned below, declare that this work has not previously been submitted to this or any other university and that it is, unless otherwise stated, entirely my own work.

Date	Arka Mallick

Abstract

Accurately modelling features for sonar image matching has always been very challenging. Only recently in Valdenegro et al [13], instead of hand designed features, a Convolutional Neural Network(CNN) was encoded to learn the general similarity function to be able to predict a pair of sonar patches belong to the different instance of same object or a different one. The result has been a significant improvement over the state of the art methods. However the results need to be further improved. The error in it's prediction affects the overall performance of object recognition, tracking etc high level tasks which are based on patch matching. In this work more advanced CNNs are evaluated with the ambination of improving the state of the art results. Using Densenet it was possible to predict binary classification score of matching and non-matching cases with 0.955 average AUC of ten trials, where the network was trained from scratch. This was a clear improvement over the state of the art [13] results of AUC 0.894 on the same unseen test dataset. For Siamese network with Densenet branches the best performance was 0.921 average roc AUC. A Siamese network with VGG branches were also evaluated along with Contrastive loss. This network's best perfromance was 0.944 average roc AUC. To ensure the evaluation is effective thorough hyper parameter search has been performed for all three methods.

To my maternal uncle Subhas Chandra Ghosh,
without whose support my dream of pursuing higher studies would not
have been possible. May he rest in peace.

Acknowledgements

My deepest thanks goes to my supervisor Dr. Matias Valdenegro-Toro. For several reasons. From technical aspect, his guidance and suggestions were invaluable and always spot on. At the begining I spent several months training for deep learning and exploring other prerequisits for this work. He has been very supportive and motivating through out the period of this thesis and even much before. In spite of very busy schedule, he spent lot of hours helping me, many times even during weekends. It was a great experience working with him. Also, it was a long time aspiration to work in the domain of underwater robotics. A special thanks to him for giving me this opportunity.

Special thanks Prof. Dr. Paul G. Plöger for making this project possible through his supervision and invaluable inputs, also for the neural networks lesson, which was pivotal for deep learning training. I sincerely thank to Prof. Dr. Gerhard K. Kraetzschmar for kindly agreeing to supervise this project. Along with all the technical lessons, I also thank him for the wonderful opportunity of being able to take part in Robocup competitions, which were great experience for me to learn hands on programming and be in touch with great minds. In the context, my sincere thanks Santosh, Oscar and Shehzad who were great mentors and taught me many important lessons. I also thank Prof. Dr. Rudolf Berrendorf and the entire cluster team for providing such great computational environment for all of the students. Without cluster I could have never finished this work.

I must thank my best friends Swarit, Pranika, Meemansha, Arshia, Aaqib, Nour for making my stay in Bonn stress-free and enjoyable. I also thank Adhideb and Abhilash, without their help I would not dare dreaming about masters in Germany. Finally, I thank my family and my partner Arshia for being supportive and encouraging through out and just being there whenever I need them.

Contents

1 Introduction **1**

1.1 Motivation 2

1.2 Problem Statement 3

2 State of the Art **5**

2.1 brief overview of the sonar image existing techniques here 5

2.2 Deep neural network 5

2.3 Convolutional network 5

2.4 Siamese network 6

2.5 Valdenegro et al. 6

2.6 Densenet 7

2.7 Contrastive loss 8

2.8 Limitations of previous work 9

3 Methodology **11**

3.1 Data 11

3.1.1 Sonar image capture procedure 11

3.1.2 Data generation for deep learning 12

3.2 Experimental Design 14

3.2.1 Search for best hyperparameters 14

3.2.2 Grid search 15

4 Evaluation **17**

5 Results **19**

5.1 Use case 1 19

5.2 Use case 2 19

5.3 Use case 3 19

6 Conclusions **21**

6.1 Contributions 21

6.2 Lessons learned 21

6.3 Future work 21

Appendix A Design Details **23**

Appendix B Parameters **25**

References **27**

List of Figures

1.1 Use of convolutional network for learning general similarity function for image patches. The patches in the image are samples taken from the data used in this thesis. Inspired from Zagoruyko et al.[18] . . . 3

2.1 Two channel(left) and Siamese CNN architectures used in Valdenegro et al., though the figure and inspiration is from [18] 7
2.2 Densenet structure. 8

3.1 Algorithm for matching and non matching pair of patch generation from Valdenegro et al [13] 12
3.2 Data processing pipe line 12
3.3 Some sample of the matching and non-matching pairs from Valdenegro et al [13] 14
3.4 Hyper parameters word cloud. 15

List of Tables

1.1 Best result from Valdenegro et. al. 'Different' represents the underlying test objects were different from the train objects. The binary prediction scores are presented here from the original results in [13] 3

3.1 Custom grid search space example. 15

1

Introduction

"You can't cross the sea merely by standing and staring at the water."

- Rabindranath Tagore

More than two thirds of the earth surface is covered by ocean and other water bodies. For a human it is often impossible to be able to explore it extensively. For finding new source of energy, monitoring tsunami, global warming or may be just to learn about deep sea eco-system or may be to look for a lost ship or an airplane, the need for venturing into potentially dangerous underwater scenarios appears regularly, in fact getting more frequent. That's why more and more robots are being deployed in underwater scenarios and lot of research is going in this direction. Some of the exploration or monitoring tasks requires the robot to see underwater, in order to make intelligent decisions. But underwater environment is very difficult for optical cameras, as light is attenuated and absorbed by the particles in the water. And lot of real life monitoring and mapping tasks take place in cluttered and turbid underwater scenario. The limited visibility range of optical sensor is a big challenge. Hence sonar is more practical choice for underwater sensing as the sound can travel long distances with comparatively little attenuation. Though there are challenges with acoustic sensing as well, for example, it has low signal-to-noise ratio, lower resolution and unwanted reflections also cause problems.

Patch matching is one of the fundamental tasks in todays myriad computer vision and image processing applications. Patch matching can be used as low-level tasks like image stitching [2], deriving structure from motion [10] or high-level task as object instance recognition [8], object classification [17], multi-view reconstruction [11], image-retrieval etc. Patch matching for acoustic images are getting increasingly useful in underwater scenarios for data association in simultaneous localization and mapping (SLAM), object tracking, sonar image mosaicing [7] etc. applications. Typical challenges in patch matching tasks are different viewing points, variations in illumination of the scene, occlusions and different sensor settings. For patch matching in sonar images the typical issues with sonar images adds to the overall challenge, for example low signal to noise ratio, less of visibility causes the underlying object features to be not so prominent as a normal image. So it has been found that it is very challenging task to manually design features for the matching task is very challenging.

1.1 Motivation

In Valdenegro et al. [13] it was first displayed that deep learning methods can be directly applied to the sonar image patch classification task, without use of any hand designed features it is still possible to perform a patch matching task with high accuracy. In [13] the authors recorded forward looking sonar images of different objects in underwater environment and generated balanced dataset of image patches for the classification task. Two channel convolutional network and Siamese convolutional network were used in predicting the binary classification scores (table 1.1) on unseen test data. Using Two channel CNN the overall best result obtained was of roc area under curve of 0.894, with test accuracy of 82.9%. The reported results are better than both classic keypoint matching methods (AUC 0.61 to 0.68) and ML-based methods such as Support Vector Machine (SVM) and Random Forests (AUC 0.65 to 0.80). These findings were pivotal for the work presented in this thesis.

Network Type	Output	Test Objects	AUC	Mean Accuracy
2-Chan CNN	Score	Different	0.894	82.9%
Siamese CNN	Score	Different	0.826	77.0%

Table 1.1: Best result from Valdenegro et. al. 'Different' represents the underlying test objects were different from the train objects. The binary prediction scores are presented here from the original results in [13]

1.2 Problem Statement

However the results in [13] is far from perfect yet and the mis-classifications affect the object detection, recognition and tracking, which are dependent on the patch matching task. It is desired that the result is further improved. With this goal the same dataset has been obtained from the [13] for further evaluation. In this thesis more advanced architectures (such as Densenet) [6] will be evaluated on the data and the goal is to improve the state of the art on the dataset.

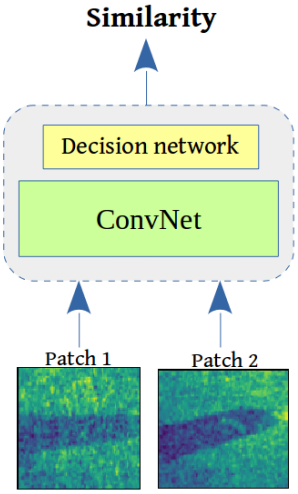


Figure 1.1: Use of convolutional network for learning general similarity function for image patches. The patches in the image are samples taken from the data used in this thesis. Inspired from Zagoruyko et al.[18]

For performing the patch matching task without using any hand designed features such as SIFT [9], we need to encode a network to learn the general similarity function (figure 1.1) for sonar image patches. Which is able to predict that two input sonar patches contain different views of a same object or not i.e matching or non-matching respectively.

With the goal of obtaining the best possible result in the evaluation, best hyperparameter search needs to be performed for each of the networks. The hyperparameters are the parameters of network or learning process whose values are already set before the training starts [16]. For example the starting learning rate or the optimizer can be fixed before the training process start.

Performing best hyper parameter search for each of the architectures is very important part of the evaluation. Another objective of this work is to do a comparative analysis of the performance of all the architectures.

2

State of the Art

2.1 brief overview of the sonar image existing techniques here

2.2 Deep neural network

From wikipedia [15] deep neural networks (DNN) are type of artificial neural networks which can have many layers between their input and output. The DNN can be used to estimate the underlying mathematical manipulation which maps the output to the input in, a manner which is more of less consistent across all the samples. The network calculate for all the layers the probability of each outputs and then the correct label can be displayed as output if the probability is above a certain threshold. Each mathematical manipulation can be considered a layer, a DNN can have many such operating layers, that is why it is named deep neural network. For example, in our scenario two sonar image patches are given as input to the DNN, it will calculate different features from both image patches and at the end it will decide if the derived features are similar then the input patches are similar.

2.3 Convolutional network

[14] Describe the parts of the conv net and the roles of the layers. Such as fully connected layers, max pooling, average pooling, activation functions such as Relu and Sigmoid. Initializers ?

Inspired by [13] following notation for CNN layers are used to describe details of the structure: Conv(Nf,Fw xFh) is a convolutional layers with Nf filters of width Fw and height Fh. A max-pooling layer is represented by MP(Pw, Ph) where PwxPh is the sub-sampling size of the layer, and FC(n) is a fully connected layers where n represents output size.

2.4 Siamese network

In 1993 the concept of a new artificial neural network called Siamese network was introduced by Bromley et al. [1]. Siamese network consists two identical sub-networks which are joined at the output. During training stage one input each is connected to each of the subnetworks. The main idea here is that the subnetworks (also called branches) are trained simultaneously and extract features from the inputs. While the shared neurons are capable of measuring the distance between the two extracted feature vectors by each branches. If the predicted distance between two feature vectors is lesser than a threshold then it can be considered that the inputs are similar. Authors used this concept to compare two signatures in [1]. One of the signature was previously obtained from the authentic owner (up to 6 signatures were recorded). This was then used to compare with a new signature to verify if the both persons are same or not, as precaution against forgery. Authors implemented the Siamese network to be able to extract and compare different features of the two signatures and if the output is within a threshold then they were considered matching. If not then it was most likely a forgery.

2.5 Valdenegro et al.

Zagoruyko et al. have demonstrated that CNNs can be directly deployed to learn the underlying similarity function to be able to determine that two input images/patches are different instances of same object or not, without any help from hand designed features. The scarcity of accurate hand designed features for sonar images motivated Valdenegro et al. to evaluate similar approach as Zagoruyko et al. and encode a CNN for sonar image comparison.

Valdenegro et al. [13] evaluated two architectures, a two-channel network and a Siamese network. The architectures 2.1 were based on the work from zagoruyko et al. [18]. A grid search was used over pre-defined set of parameters which define the

network structure. The final two channel network structure presented in the paper for predicting binary classification score was as follows, Conv(16, 5 x 5)-MP(2, 2)-Conv(32, 5 x 5)-MP(2, 2)-Conv(32, 5 x 5)-MP(2, 2)- Conv(16, 5 x 5)-MP(2, 2)-FC(64)-FC(32)-FC(1). Similarly grid search was also conducted for Siamese network structure for classification score. The structure for each of the branches or sub-network is as follows, Conv(16, 5 x 5)-MP(2, 2)-Conv(32, 5 x 5)- MP(2, 2)-Conv(64, 5 x 5)-MP(2, 2)-Conv(32, 5 x 5)-MP(2, 2)-FC(96)-FC(96). The output features from the branches were then concatenated to form 192 element vector. This was then passed through a decision network with FC(64)-FC(1). For both two channel and Siamese network the final FC(1) layer had Sigmoid [4] activation function and binary cross entropy loss[3] function.

Not very complex network structures were used. In general sense, adding more layers in the network does add more non-linearity to it and in some cases, that improves the performance of the network. So in theory, more advanced networks or loss functions which help extracting more discriminative and patch invariant features, should improve the results even further.

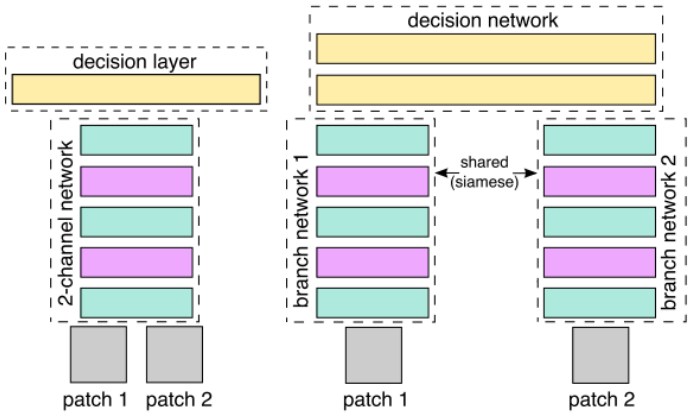


Figure 2.1: Two channel(left) and Siamese CNN architectures used in Valdenegro et al., though the figure and inspiration is from [18]

2.6 Densenet

In Densenet each layer connects to every layer in a feed-forward fashion. With the basic idea to enhance the feature propagation, each layer of Densenet blocks

takes the feature-maps of the previous stages as input.

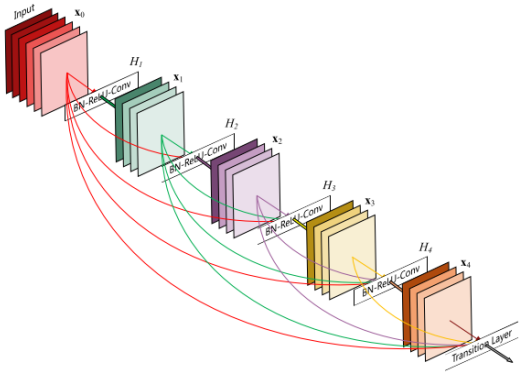


Figure 2.2: Densenet structure.

2.7 Contrastive loss

Using contrastive loss higher dimensional input data (e.g. pair of images) can be mapped in the much lower dimensional output manifold, where similar pairs are placed closer to each other and the dissimilar pairs have bigger distances between them depending on their dissimilarity. So using this loss function the distance between two input patches projected in the output manifold can be predicted and if the distance is closer to 0 then the input pairs are matching, otherwise its dissimilar (above threshold). The formula for the loss is shown below.

$$D_w(\overrightarrow{X_1}, \overrightarrow{X_2}) = \|G_w(\overrightarrow{X_1}) - G_w(\overrightarrow{X_2})\|_2$$

$$L(W, Y, \overrightarrow{X_1}, \overrightarrow{X_2}) = (1 - Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}\{max(0, m - D_w)\}^2$$

Here L is the loss term, the formula presented here is the most generalized form of the loss function, suitable for batch training. $\vec{X}_1$, $\vec{X}_2$ represents a pair of input image vectors. Y are the labels, 0 for similar pair and 1 for dissimilar pair. D_w is the parameterized distance function to be learned by the algorithm. m is the margin and m is always greater than zero and it defines a radius around G_w. The dissimilar pairs only contribute to the loss function if their distance is within the radius. One of the idea for evaluating this loss function is to use it with a Siamese

network, as the loss function takes a pair of images as input. So its very relevant to the problem statement of this thesis.

2.8 Limitations of previous work

Why the sonar ones does not work so well and why the state of the art can be improved.

3

Methodology

3.1 Data

Just like any other machine learning project the data is the most important part of the evaluation. Hence for the proper qualitative evaluation a deeper look at the data and how it was generated is needed. The data set for training and testing are both obtained from Valdenegro et al [13]. Following is a brief description of the overall pipeline to obtain the data for training.

3.1.1 Sonar image capture procedure

In Valdenegro et al [13] the authors have explained in details how the raw sonar images were captured. In a water tank in their facility, the images were taken using ARIS Explorer 3000 and mounted forward looking sonar(FLS). The objects featured are household garbage items and common marine debris. There are total 9 different types, such as metal cans, bottles, drink carton, metal chain, propeller, tire, hook, valve. In [13], in controlled underwater environment total of 2072 images; about 2500 instances of the aforementioned objects were captured and labeled with 9 classes accordingly.

$L_m \leftarrow \emptyset, L_{nm} \leftarrow \emptyset$
for img $\in I$ **do**
  for object $o \in$ img **do**
    $OC \leftarrow$ crop B_o from img.
    for $i = 0$ to S_p **do**
      $MC \leftarrow$ sample random object p of class C_o and make an image crop.
      Append (OC, MC) to L_m
    end for
    for $i = 0$ to S_n **do**
      $NMC \leftarrow$ sample random object p of class $C_p \neq C_o$, and make an image crop.
      Append (OC, NMC) to L_{nm}
    end for
    for $i = 0$ to S_n **do**
      $BC \leftarrow$ sample random background patch and make an image crop.
      Append (OC, BC) to L_{nm}
    end for
  end for
end for

Figure 3.1: Algorithm for matching and non matching pair of patch generation from Valdenegro et al [13]

3.1.2 Data generation for deep learning

As part of the same work [13], matching and non-matching pairs of sonar image patches were generated using a patch generation algorithm displayed in figure 3.1 where each patch, an instance of one of the 9 classes, were obtained from meaningful crops of the original sonar images. Authors have figured that 96x96 pixels is the best size of the crops.

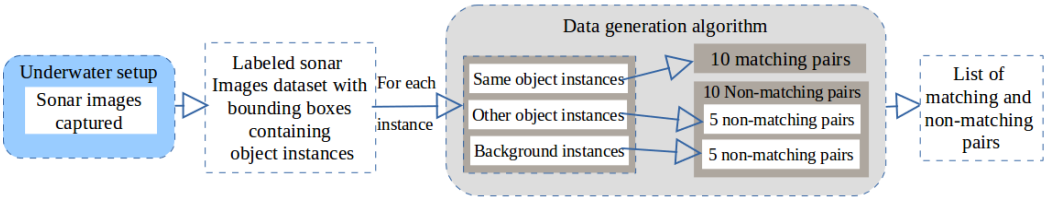
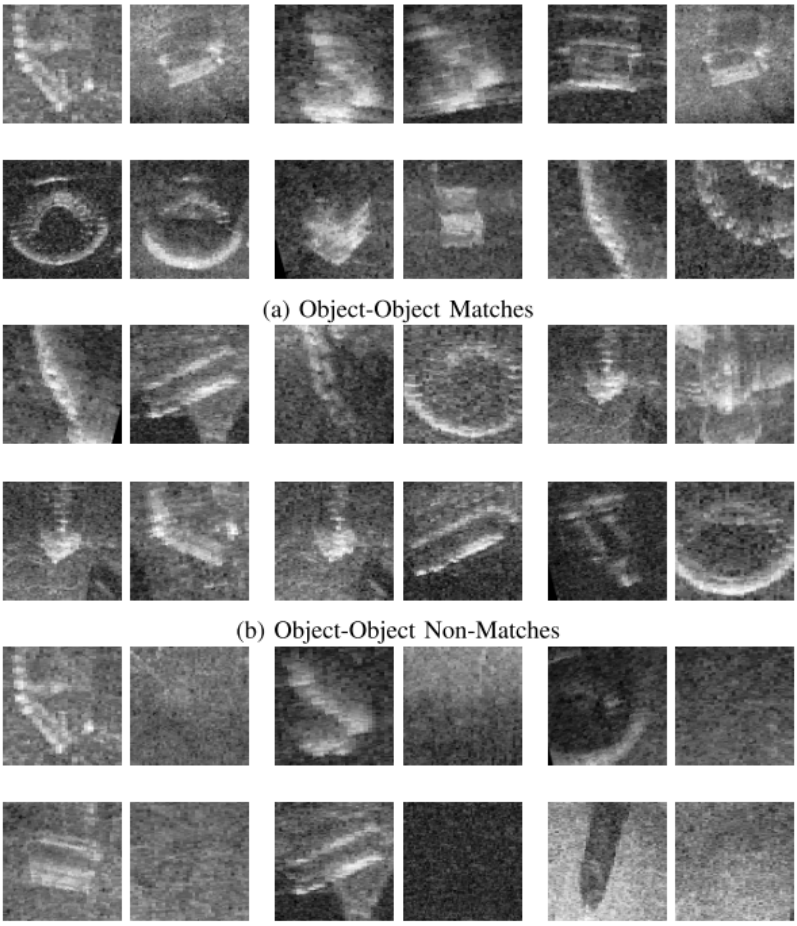


Figure 3.2: Data processing pipe line

To create matching pair, two patches that belong to the same object-type but different perspective and insonification level were used. For generating non-

matching pair, two instances of different object-types were used. Also, one instance of an object and a background patch, which contains no object instance, were used to generate additional non-matching patches. According to the authors of [13], for balanced and effective learning, the ratio of matching and non-matching pairs in both train and test data is maintained at 1:1. That is, for every ten matching pair five object-object non-matching and five object-background non-matching pairs were generated. In figure 3.2 the overall pipeline for the data generation is displayed in simple block diagram. In figure 3.3 some sample patches from all type of pairs are displayed. Using the patch generation algorithm total of 39840 pairs were generated from the instances of 6 distinct object type, which were used as the train data. While another 7440 pairs were generated from the instances of remaining object types, for testing purpose. This test dataset does not contain any common object with the training dataset, it should be a good test for the generalization of the approaches. The labels for the data are 0 and 1 representing non-matching and matching pairs respectively. For this thesis work, this dataset is obtained in HDF5 files [5]. It contains patches in form of tensors [12], each of shape (2,96,96). Here the pair of patches (size 96x96) are placed in a way that each channel contains one patch.



(a) Object-Object Matches

(b) Object-Object Non-Matches

(c) Object-Background Non-Matches

Figure 3.3: Some sample of the matching and non-matching pairs from Valdenegro et al [13]

3.2 Experimental Design

3.2.1 Search for best hyperparameters

What are the hyperparameters. Effect of setting best hyper parameters.

Figure 3.4: Hyper parameters word cloud.

Epochs	Optimizer	Batch size	Dropout	Use batch norm
5	adam	64	0.4	True
5	adam	64	0.4	False

Table 3.1: Custom grid search space example.

3.2.2 Grid search

Currently the search space for the evaluation is hand designed and supplied to the evaluation script externally. Example

In this example it is displayed that how the search space is hand designed to evaluate different cases, here the effect of using batch normalization can be investigated by comparing the results of two test cases. All the values set in the grid passed as input to the actual code and defines the structure or different parameters from inside the code. In 3.1 the flag use batch norm is set to 'True', which gets propagated in the actual block of code where it enables the use of batch normalization layer, where applicable.

For the evaluation the epochs are set after multiple testings in order to ensure, to some extend, the networks does not get too overtrained and the generalization gets poorer. Also if the network is undertrained, then it will not generalize well. The setting the epochs were one of the big challenge of this work, because in some cases the networks gets overtrained after 4-5 epochs.

4

Evaluation

Implementation and measurements.

- Compare based on AUC
- What is roc AUC
- AUC vs Accuracy, why AUC is better than Accuracy or other point based methods.
- Implementation details of all three structures here with explanation
- Implementation from where why chose this one
- modifications made and why
- diagrams and hyper parameter lists

5

Results

5.1 Use case 1

Describe results and analyse them

5.2 Use case 2

5.3 Use case 3

6

Conclusions

6.1 Contributions

6.2 Lessons learned

6.3 Future work

A

Design Details

Your first appendix

B

Parameters

Your second chapter appendix

References

[1] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[2] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73, 2007.

[3] François Chollet et al. Binary ross entropy loss keras, December 2018. URL `https://keras.io/losses/#binary_crossentropy`.

[4] François Chollet et al. Sigmoid keras, December 2018. URL `https://keras.io/activations/#sigmoid`.

[5] The HDF Group. What is hdf5, December 2018. URL `https://support.hdfgroup.org/HDF5/whatishdf5.html`.

[6] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[7] Natalia Hurtós, Yvan Petillot, Joaquim Salvi, et al. Fourier-based registrations for two-dimensional forward-looking sonar image mosaicing. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5298–5305. Ieee, 2012.

[8] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[9] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[10] Nicholas Molton, Andrew J Davison, and Ian D Reid. Locally planar patch features for real-time structure from motion. In *Bmvc*, pages 1–10. Citeseer, 2004.

[11] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *null*, pages 519–528. IEEE, 2006.

[12] Tensorflow. Tensors, December 2018. URL `https://www.tensorflow.org/programmers_guide/tensors`.

[13] Matias Valdenegro-Toro. Improving sonar image patch matching via deep learning. In *Mobile Robots (ECMR), 2017 European Conference on*, pages 1–6. IEEE, 2017.

[14] Wikipedia. Convolutional neural networks, November 2018. URL `https://en.wikipedia.org/wiki/Convolutional_neural_network`.

[15] Wikipedia. Deep neural networks, November 2018. URL `https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks`.

[16] Wikipedia. Hyperparameter machine learning, November 2018. URL `https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)`.

[17] Bangpeng Yao, Gary Bradski, and Li Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3466–3473. IEEE, 2012.

[18] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2015.