Sea Lion Population Count: How Many Sea Lions Do You See?

Hector Anadon, Albert Bou, and Beatrice Ionascu

KTH Royal Institute of Technology

Abstract. This paper presents the adaptation of a convolutional neural network (CNN) cascade originally designed for face detection for a different visual object detection task involving sea lions. The cascade architecture built on CNNs is capable of combining increasingly more complex classifiers that operate on increasingly higher resolutions. This allows background regions of the image to be quickly rejected in the low resolution stage, while spending more computation on promising sea lion-like regions during higher resolution stages. The cascade can be viewed as a sea lion-focusing mechanism which ensures that discarded regions are unlikely to contain sea lions. The cascade consists of alternating detection and calibration networks, so that the output of each calibration stage is used to adjust the detection window position for input to the subsequent stage, which ultimately leads to improved localization effectiveness. The system attains competitive detection performance on the NOAA Sea Lions dataset.

Keywords: neural nets, object detection, graphics processing units (GPU), image recognition, image resolution, convolutional neural network cascade, discriminative capability, CNN-based calibration

1 Introduction/Problem formulation

Steller sea lions are an endangered population of sea lions in the western Aleutian Islands in the North Pacific. Due to their 94% decline in the last 30 years, they have become the focus of conservation efforts which require annual population counts [1]. Specially trained scientists at NOAA Fisheries Alaska Fisheries Science Center [2] conduct these surveys using airplanes and unoccupied aircraft systems to collect aerial images. Having accurate population estimates enables environmental scientists to better understand factors that may be contributing to the lack of recovery of Stellers in this area.

Currently, it takes biologists up to four months to count sea lions from the thousands of images NOAA Fisheries collects each year. Once individual counts are conducted, the tallies must be reconciled to confirm their reliability. The results of these counts are time-sensitive.

Algorithms which accurately count the number of sea lions in aerial photographs could tremendously benefit the researchers. Automating the annual population count would free up critical resources allowing NOAA Fisheries to

focus on what cannot be done by a computer: developing scientific technology and information to support the sustainable management, protection and conservation of the region's maritime habitat.

The sea lions population is not the only endangered species that can benefit from advancements in computer vision and deep learning. Applied to aerial population counts, these algorithms can greatly benefit other endangered species that require yearly accurate population estimates.

In essence, the problem of population counts from aerial images is an application of one of the main tasks in computer vision, namely visual object detection. Object detection is one of the more challenging problems in computer vision, not only because it combines classification and localization, but because it involves multiple categories and requires that all instances of these categories are correctly labeled and localized in an image.

In addition to the typical difficulties of a computer vision task, such as addressing changes in illumination, scale, appearance, etc., object detection from large aerial images also has the challenge of a large search space of possible object positions. The former category of challenges require accurate binary and multi-class classifiers, whereas the latter require time efficient algorithms.

It is thus essential to use a system that enables fast evaluation and quick early rejection of false positive detections as well as learning of features that can capture complex variations and effectively differentiate between different objects, or sea lions in our case.

CNNs are suitable for complex classification tasks [3], however their high computational expense prohibits the use of deep structures for scanning large images. As an alternative, we explore a cascade of shallow CNNs, an approach that is more fitted for practical purposes and that has been shown to give excellent results for face detection [4].

We present here an adaptation of the model proposed by [4] and our results on the NOAA Sea Lions dataset (NOAA) [5].

2 Background

2.1 Historical context

Detection is in fact an old problem in computer vision. We list here some of the most notable approaches that were successful before the onset of deep learning.

In 2001, Viola *et al.* [6] proposed the Viola-Jones detector consisting of a boosted cascade with Haar features. At the time, this was considered to be the most effective approach for face detection, however the simple nature of the features leads to inferior performance in uncontrolled environments. Improvements have been proposed to this method, mainly focusing on the use of more complex features, in order to improve classification accuracy [7].

In 2005, Dalal *et al.* [8] proposed a framework for pedestrian detection that used a feature representation called histogram oriented gradients. This was very successful at the time as it outperformed previous hand-crafted features, i.e. edge and gradient-based descriptors.

It also led to subsequent groundbreaking work, including Felzenszwalb *et al.*'s deformable parts model [9]. This model uses a sliding window approach where the classifier is run at evenly spaced locations over the entire image, but some argue that it has some resemblance to CNNs due to the structure of the classifier [10].

2.2 Deep learning-based

Indeed, the past decade has seen a shift in focus towards CNNs and, more generally, approaches that allow more complex and computationally demanding classifiers to be used in object detection. Every year one or more groundbreaking methods emerge, and we list here what is considered to be state-of-the art in terms of visual object recognition today based on performances achieved on VOC challenges [11].

One popular approach is using region proposals. Girshick *et al.*'s R-CNN [12] follows the region proposal paradigm to first generate category-independent bounding boxes and extract CNN features from the regions. Then it applies class-specific classifiers to recognize the object category of the proposals.

Although it was the winning approach at its time, R-CNN was followed by two attempts that improved its speed at test time: Fast R-CNN [13] and Faster R-CNN [14].

Finally, Redmon *et al.* [15] proposed a completely different method, YOLO, outperforming all variants of R-CNN and other previous approaches. Their approach looks at object detection as a regression problem, directly mapping image pixels to bounding box coordinates and class probabilities. In this extremely fast model, a single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation.

Our problem poses challenges different from the general object detection task, because of the small size of the interest objects with respect to the size of the images, and hence the methods presented above are not the most effective.

Zhang *et al.* [16] and Park *et al.* [17] address this by adopting a multi-resolution idea. While sharing a similar technique, Li *et al.* [4] propose an approach that utilizes CNNs as the classifiers and combines the multi-resolution and calibration ideas for face detection in a multi-resolution CNN cascade architecture that can be more discriminative than the single resolution CNN with only a fractional overhead. This approach introduces a CNN-based face bounding box calibration step in the cascade to help accelerate the CNN cascade and obtain high quality localization.

We adapt the approach in [4] to solve the problem of sea lion detection and classification.

3 Approach

3.1 Overall framework

The overall test pipeline of our sea lion detector and classifier consists of a set of seven different CNNs applied in cascade to an input image. The final output

is the number of pups, juveniles, subadult males, adult males and adult females that appear in an image. The full test pipeline end-to-end approach is shown in Figure 1.

The model works with three levels of image resolution; high, medium and low. Given a test image with low resolution level, *12-net* scans the whole image by means of a sliding window of size 12x12 and binary classifies the different windows as containing or not a sea lion (no matter which type). Working in low resolution allows the network to quickly reject an important percentage of the windows. The location of the remaining detection windows is then adjusted by the *12-calibration-net* to approach potential sea lions nearby. Next, Non-maximum suppression (NMS) is applied to discard highly overlapping detection windows.

At the end of the low resolution stage, all remaining windows potentially containing sea lions are resized to medium resolution (24x24 size) to be processed by *24-net*. Similar to the first stage, this second stage also consist of a binary classification network, a calibration network (*24-calibration-net*) and a NMS process. The remaining windows at the end of the medium resolution stage are resized to high resolution level (48x48) to be finally processed by *48-net* and *48-calibration-net*. These last two networks are a deeper version of the one we can find in the first two stages.

On top of the detection model, we have included a classification network. The outputs of this last networks permit to obtain a final count of the number of sea lions of each class in the image.

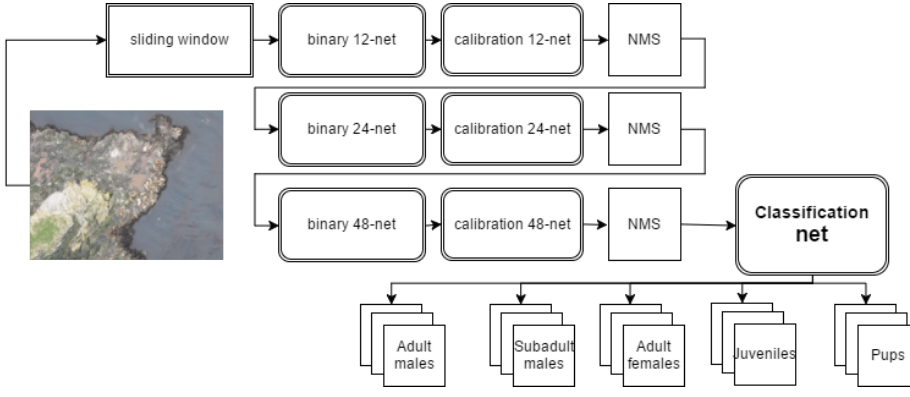

Fig. 1. Overall framework

3.2 Binary classification

12-net is the first net in the pipeline. It is a shallow convolutional network that scans the input image in low resolution and performs binary classification sea lion vs. no sea lion.

12-net takes a 12x12 image in RGB as input and first applies a set of 16 3x3 filters with stride 1. Following, a max-pooling layer with 3x3 kernels and stride 2 is performed. Finally, two consecutive fully connected layers with 16 and 2 outputs respectively complete the full architecture of the network. ReLU nonlinearity function is applied after the pooling layer and the first fully connected layer. Softmax function is applied after the second fully connected layer to interpret the final output as probabilities.

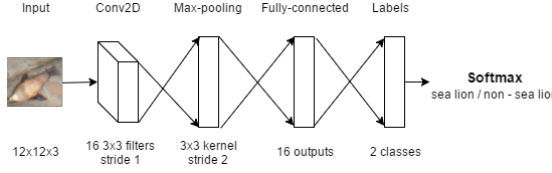

Fig. 2. Architecture for binary *12-net*

24-net refers to the second binary classification network in the pipeline, which processes images in medium resolution. The aim of this net is to further reduce the number of detection windows potentially containing a sea lion at a faster pace than if using full resolution images. Therefore, *24-net* aims at retaining the most promising windows, which will be then processed in the last stage.

24-net has a 2-branch structure. For the main branch, the input image has size 24x24 in each normalized RGB channel. Then, a set of 64 5x5 filters with stride 1 is applied. Following, a max-pooling layer with 3x3 kernels and stride 2 is performed as in *12-net*. At this point, the main branch continues with a fully connected layer with 128 output nodes. The secondary branch has the same structure as *12-net* up to the first fully connected layer.

At the end, the outputs of the two independent fully connected layers are concatenated and used as input to another fully connected layer with 2 output nodes. Again, ReLU nonlinearity function is applied after the pooling layers and the first fully connected layers of each branch and Softmax at the last layer.

48-net refers to the final binary classification network in the pipeline. In this final stage, it is feasible to apply a deeper network architecture to detect sea lions since the number of windows to be analyzed should be only a small portion of the initial windows.

Just as *24-net*, *48-net* has a 2-branch structure. For the main branch, the input image has size 48x48 for each normalized RGB channel. Initially, a set of 64 5x5 filters with stride 1 followed by a max-pooling layer with 3x3 kernels and stride 2 is applied. Then, a second combination of convolution layer and max-pooling layer is added with batch normalization layers before and after the convolutional layer. This branch ends with a fully connected layers of 256 output

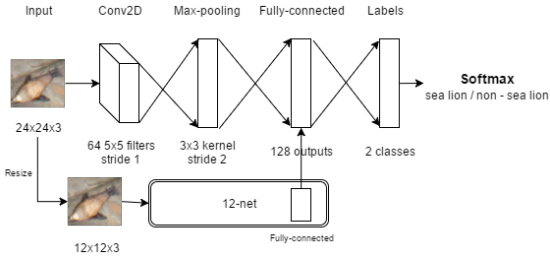

Fig. 3. Architecture for binary *24-net*

nodes. The secondary branch has the same structure as *24-net* up to the first fully connected layer.

Similarly to *24-net*, the outputs of the two independent fully connected layers is concatenated and used as input to another fully connected layer with 2 output nodes. Once again, ReLU and Softmax are used as activation layers.

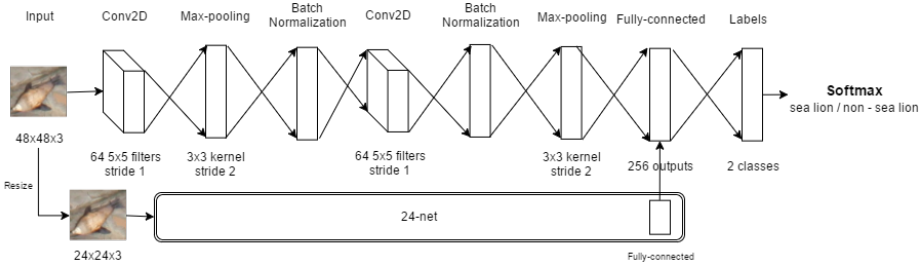

Fig. 4. Architecture for binary *48-net*

3.3 Bounding box calibration

12-calibration-net is a shallow CNN applied right after *12-net* with the objective of adjusting the location of the windows that may contain a sea lion so the lion ends at the center of the image. Given a detection window with the top left corner in (x, y), nine possible different shifts are defined as:

$$x_n \in \{-15, 0, 15\}$$
$$y_n \in \{-15, 0, 15\}$$

where each value refers to a the number of pixels the window should be shifted in the x or the y axis for the high resolution case (i.e. moving actually one quarter of pixels in this resolution scale).

12-calibration-net takes as input a 12x12 window in RGB format (normalized) and first applies a convolutional layer with 16 3x3 filter and stride 2. Following, a max-pooling layer of 3x3 kernel and stride 2 and 2 consecutive fully connected layers with 128 and 9 outputs respectively complete the architecture. Note that the final number of outputs is the same as the number of possible shifting patterns.

The output of the network is a vector with the probability of each pattern. Since calibration patterns are not orthogonal to each other, we take the averaged shift over all patterns that obtained a probability value over a previously specified threshold. If no pattern probability is over the threshold, no shift is applied.

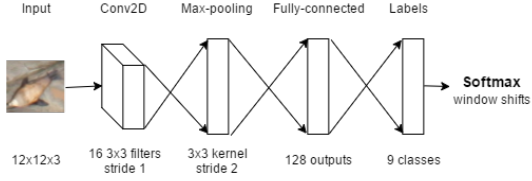

Fig. 5. Architecture for *12-calibration-net*

24-calibration-net is the second calibration net of the pipeline. The exact same shifting patters defined for *12-calibration-net* are defined for this network. Also, it has a very similar structure to *12-calibration-net*. Nonetheless, in this case the filter size of the convolutional layer is 5x5 and the number of outputs of the fully connected layer is 64. The number of final outputs, as in the previous case, is the number of possible shift we can apply to the image, nine.

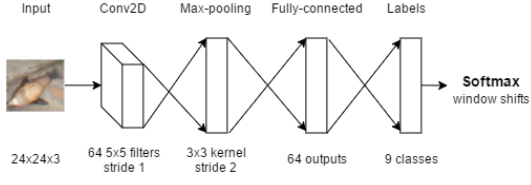

Fig. 6. Architecture for *24-calibration-net*

48-calibration-net is the last calibration net in the cascade. In order to achieve a more accurate calibration, this is a deeper network than the first two calibration networks. More specifically, this networks starts with a set of 64 5x5 filters with stride 1 followed by a max-pooling layer with 3x3 kernels and stride 2, and

follows with a second set of convolution layer and max-pooling layer is with batch normalization layers before and after the convolutional layer. The model ends with two fully connected layers, with 256 and the number of calibration patterns respectively. The output movements are the same as in previuos calibration networks.

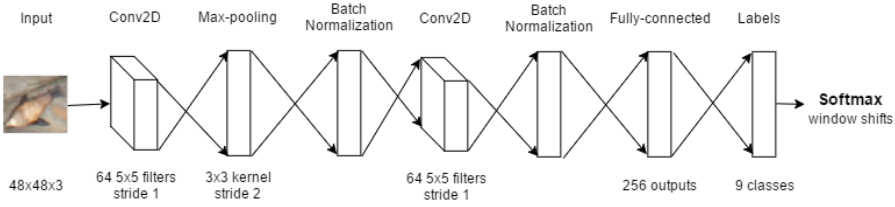

Fig. 7. Architecture for *48-calibration-net*

3.4 Multiclass-classification-net

The *multiclass-classification-net* is the last net in the cascade. As shown in Figure 8, the architecture starts with a set of 32 5x5 filters followed by a max-pooling layer with 2x2 kernel and follows with a second set of convolution layer with 64 filters instead. Finally, a fully connected layer of 512 outputs with dropout is added.

Giving a list of windows returned by the *48-net*, this network predicts the probability of belonging to the different sea lion classes.

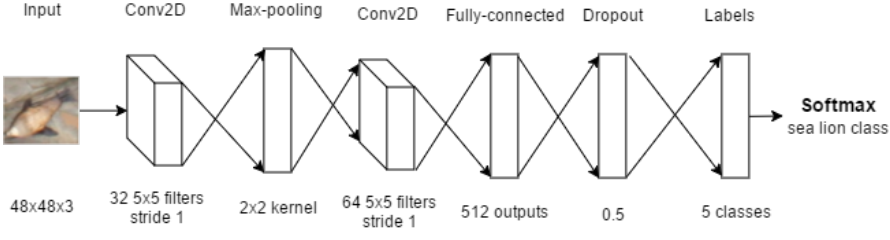

Fig. 8. Architecture for *multiclass-classification-net*

3.5 Training

To train all the CNNs in the cascade, seven different datasets were generated. That allowed us to parallelize the process and train each net independently.

For binary nets, images of all sea lions in the dataset (a total of 75,000) were cropped out and resized to low, medium and high resolution. Furthermore, 125,000 negative samples where randomly selected from background regions on the images. These images were also resized and added to each one of the different resolution datasets.

For calibration nets, images of sea lions were altered by using the inverse shifting patters defined as outputs. More specifically, for the n-th pattern (x_n,y_n), we shifted the corners of the images by $(-x_n,-y_n)$ and labeled the pattern n as the correct output. Overall, a total of 675,000 images compose each dataset in low, medium and high resolution respectively.

For *multiclass-classification-net*, the dataset used was composed by the same images used to train *48-net* (i.e. images of sea lions and background regions in high resolution). In this case, the 5 different classes of sea lions (pups, juveniles, adult males, subadult males, adult females) were defined as labels instead of the binary classification used in *48-net*.

3.6 Testing

Figure 1 shows the testing pipeline that has been followed to locate and then classify sea lions in their correspondent classes. Giving an image capture by a drone, sliding windows of size 48 by 48 are generated with 24 padding. Then, those windows are resized to 12 by 12 pixels and normalized as they will be the input for the *12-net*. The prediction of this network outputs the probability of each window containing a sea lion. In order to get better precision a threshold over 0.7 is required so false positive are avoided. In this prediction most of the windows are discarded. After that, calibration of the windows is predicted with the *12-calibration-net* giving the probabilities of every possible movement. The averaged shift of the patterns that obtained a probability value over a previously specified threshold is performed. If no pattern probability is over the threshold, no shift is applied. Once the shifting is applied, NMS (explained in Section 3.6) is perform reducing again the number of windows.

The remaining windows are resized to 24 by 24 from the original image and the actions previously explained are perform, but using the *24-nets* instead. The number of windows decreases again, and the remaining ones are resized to 48 by 48 from the original image. Again, the previously explained actions are performed over the *48-nets* obtaining windows where a sea lion appears.

Finally, the *multiclass-classification-net* is used to predict the class of each sea lion.

This pipeline returns the position and the class of every sea lion given a picture taken from a drone.

Non-maximum supression [18] is an algorithm used as an intermediate step in many comptuer vision models. In our pipeline permits to suppress overlapping windows, which avoids counting the same sea lion several times. Therefore, we have implemented a version of NMS that takes a set of window corner as input

and outputs a selection of them for which no overlapping over a certain threshold (0.3) exists.

4 Experiments

4.1 Data

The NOAA Sea Lions dataset consists of 892 large aerial photographs that contain sea lions from one or more of the following categories: adult males (also known as bulls), subadult males, adult females, juveniles, and pups. The labels are provided as a list of ground-truth counts for each image as well as markings showing where each animal is on copies of the training images. An example image from the dataset, with and without labels, is reproduced in the Appendix.

It is important to note here that the data is far from ideal. First of all, not all photographs were taken from the same altitude. The size of the adult males, for instance, ranges from 60 pixels to over 210 pixels in some of the images.

4.2 Data Processing

Most images have regions that have been blacked out by human labelers to avoid double-counting. These regions were ignored during training and testing. In addition, a number of images with mismatched labels were removed.

Upon an inspection of the data and the sizes of the sea lions, we decided to use a window size of 48 pixels that would be then resized to 24 and 12 pixels, respectively, as described in Section 3. This same value motivates the size of the sliding window used during testing.

4.3 Evaluation

The data was normalized and after 50 images were randomly selected for testing, the remaining 842 images were used to create 200,000 windows used for training (80%) and validation (20%). The datasets were zero centered by subtracting the per feature mean of the training data. The results achieved after training the networks described in Section 3 are presented below. For all networks in the pipeline, only the best models obtained during training were saved. For multiclass classification we tried different network configurations (including using a pre-trained VGG19 [19]), but only included in this report the one with the best results.

5 Results

In this section the different results are presented, then, discussed in Section 6.

In the following set of figures, we can observe the evolution of the loss function and the accuracy of the training and validation set (dotted line) during certain number of epochs. Figure 9 and 10 correspond to the binary nets, Figure 11 and

12 correspond to the calibration nets and Figure 13 and 14 correspond to the classification net.

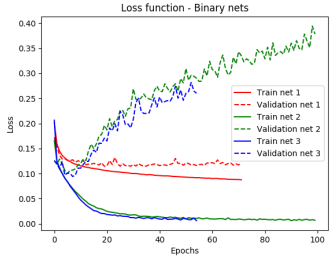

Fig. 9. Loss binary nets

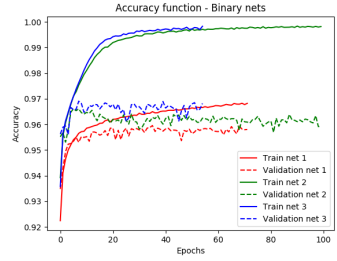

Fig. 10. Accuracy binary nets

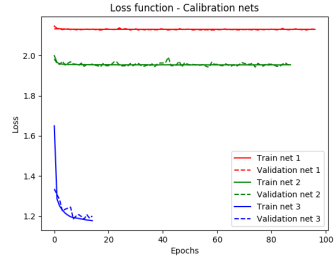

Fig. 11. Loss calibration nets

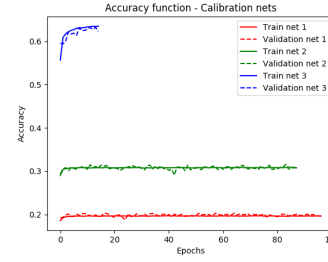

Fig. 12. Accuracy calibration nets

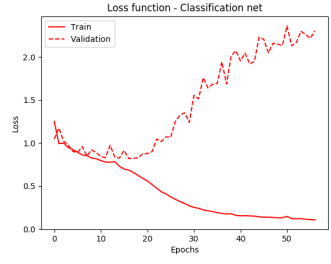

Fig. 13. Loss classification net

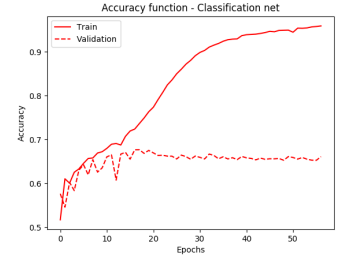

Fig. 14. Accuracy classification net

Table 1 shows the binary classification performance on the NOAA dataset, averaged over the test images. It includes number of detection windows, recall and precision rates computed after each pipeline stage.

Table 1. Binary classification performance

Network	Number of windows	Recall	Precision
Initial	29867.14	-	-
12-net	812.64	1.00	0.19
12-calibration-net	812.64	1.00	0.19
NMS	513.36	1.00	0.27
24-net	267.44	0.94	0.39
24-calibration-net	267.44	0.91	0.39
NMS	260.08	0.89	0.40
48-net	177.7	0.81	0.50
48-calibration-net	177.7	0.78	0.47
NMS	153.7	0.72	0.51

In Figure 15 we observe the evolution of the number of windows through the pipeline. The seven images are, in order: the original picture, windows after binary prediction of 12-net, after 12-net NMS, after binary prediction of net 2, after net 2 NMS, after binary prediction of net 3 and after net 3 NMS that is the final result that is introduced to the classification net.

Fig. 15. Evolution of the number of windows through the pipeline, from left to right: original image, windows after binary prediction of net 1, after net 1 NMS, after binary prediction of net 2, after net 2 NMS, after binary prediction of net 3 and after net 3 NMS (final result)

Table 2 displays the classification performance for the multi-class classification task on the NOAA dataset. It includes the average recall and the average precision rates for each class over the test images.

Table 2. Multi-class classification performance

Class	Recall	Precision
Pups	0.15	0.16
Juveniles	0.18	0.28
Subadult Males	0.01	0.08
Adult Males	0.12	0.15
Adult Females	0.41	0.26

As another measure of the performance of the pipeline at the end of the classification step, we computed the Root Mean Squared Error (RMSE) as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(c_i - \hat{c}_i)^2}$$

Where $N = 5$ (number of sea lion types), $\hat{c}_i$ refers to the number of predicted sea lions of a given class and c_i is the ground truth for that same class. The RMSE obtained is 20.65.

6 Discussion and further work

In this section the results are analyzed and further work is suggested.

As we can see in the loss function plots in Figures 9, 11 and 13, for the *12-net* and *24-net* there is clear overfitting. In the *12-calibration-net* and *24-calibration-net* we observe that the accuracy is too low and the loss function is not reduced over the time while *48-calibration-net* requires more training time. We would expect that during testing these nets would improve recall/precision; however, as shown in Table 1, they do not. Finally the *multiclass-classification-net* is overfitted.

Overall, the architecture requires further parameter tunning by means of cross validation method and further research on deeper architectures.

Even though the precision and recall rates shown in Tables 1 and 2 are not very high, the RMSE score, at the present day, would be included within the top 20 in the Kaggle competition leaderboard, showing that this is indeed a difficult problem to solve.

It is important to also note here that the recall and precision values are approximate since it was difficult to count exactly how many sea lions were correctly identified due to the close distance between sea lions in many of the images.

One other observation is that having bounding box labels instead of colored markings would have allowed us to add a resizing component to the calibration stages and ultimately improve the prediction of the system.

We have made several changes from the original architecture, where relevant, however, we believe that many of the architecture, parameter and hyperparameter choices can be improved with further testing. Some of the most important improvements to be made are listed below.

Within an image, the amount of space covered by sea lions is, by far, smaller that the background space. Therefore, it is reasonable to think that more negative samples should be included in the binary datasets. Furthermore, networks in advanced stages of the pipeline should be specialized in discarding images misclassified as sea lions in earlier nets. To that end, false positive in initial stages should be included as negative samples at more advanced stages of the pipeline.

Pups are generally of a smaller size compared to other classes of sea lion. That makes them hard to classify given a fixed size window of 48x48 in high resolution. To solve this problem, we have considered defining 2 independent branches in the pipeline, including a specific model with a smaller window size specialized in classifying pups.

When using NMS, the returned windows are not necessarily the ones that obtained the higher confidence value among each set of overlapping windows. Choosing the window with the highest confidence value would significantly improve localization, and as a result, the multi-class classification.

7 Conclusions

In this project we adapted a face recognition algorithm to the problem of locating and classifying see lions. The cascade architecture used is capable of combining increasingly more complex classifiers that operate on increasingly higher resolutions. This allows background regions of the image to be quickly rejected in the low resolution stage, while spending more time and resources during higher resolution stages on areas that sea lions are likely to be in. In addition, the calibration steps lead to having the sea lions centered in the windows, thus making the classification step easier.

The use of the cascade architecture allows us to parallelize the computations over the different networks used, leading to faster training and also testing.

Overall, this approach seems to work but there is also space for improvement. The system attains competitive detection performance on the NOAA Sea Lions dataset, as our RMSE value is comparable to those of the best approaches in the leaderboard of the Kaggle competition.

Acknowledgements

The computations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at the PDC Center for High Performance Computing at KTH.

References

1. Sweeney, K.L., Helker, V.T., Perryman, W.L., LeRoi, D.J., Fritz, L.W., Gelatt, T.S., Angliss, R.P.: Flying beneath the clouds at the edge of the world: using a hexacopter to supplement abundance surveys of steller sea lions (eumetopias jubatus) in alaska 1. Journal of Unmanned Vehicle Systems **4**(1) (2015) 70–81
2. : Alaska fisheries science center homepage - noaa. https://www.afsc.noaa.gov/ Accessed: 2017-05-21.
3. LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks **3361**(10) (1995) 1995
4. Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G.: A convolutional neural network cascade for face detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 5325–5334
5. : Noaa fisheries steller sea lion population count. https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count/data Accessed: 2017-05-21.
6. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. Volume 1., IEEE (2001) I–I
7. Zhang, C., Zhang, Z.: A survey of recent advances in face detection. Technical report (June 2010)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: In CVPR. (2005) 886–893
9. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9) (2010) 1627–1645
10. Girshick, R., Iandola, F., Darrell, T., Malik, J.: Deformable part models are convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 437–446
11. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2) (June 2010) 303–338
12. Girshick, R.B., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. CoRR **abs/1311.2524** (2013)
13. Girshick, R.B.: Fast R-CNN. CoRR **abs/1504.08083** (2015)
14. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. CoRR **abs/1506.01497** (2015)
15. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. CoRR **abs/1506.02640** (2015)
16. Zhang, W., Zelinsky, G., Samaras, D.: Real-time accurate object detection using multiple resolutions. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE (2007) 1–8
17. Park, D., Ramanan, D., Fowlkes, C.: Multiresolution models for object detection. Computer Vision–ECCV 2010 (2010) 241–254
18. Neubeck, A., Van Gool, L.: Efficient non-maximum suppression. In: Pattern Recognition, 2006. ICPR 2006. 18th International Conference on. Volume 3., IEEE (2006) 850–855
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

Appendix

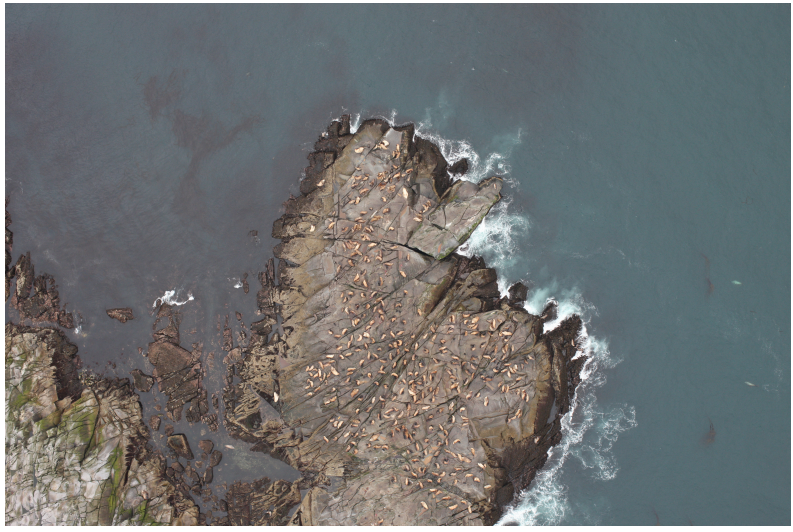

Fig. 16. Training image

Fig. 17. Labeled training image