ANALYSING MARINE ANIMALS CHARACTERISTICS USING CONVOLUTIONAL NEURAL NETWORKS

by

Parmeet Singh

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
August 2019

Table of Contents

List of Tables

List of Figures

Abstract

The thesis explores the efficacy of convolutional neural network(CNNs) to categorize lobster images for improving lobster grading and traceability. Traceability ensures that lobsters are traceable to a sustainable source. Lobsters kept in unsuitable conditions such as extremely low temperatures or densely packed crates have low chances of survival leading to a lower grade ultimately affecting prices. The CNNs were able to achieve high accuracies for assessment of lobster traits. Attention mechanisms that learn to extract discriminating features were explored to improve the performance of CNNs. The attention augmented CNNs had similar accuracies compared to the vanilla CNN but were less sensitive to choice of architecture and learning rate. The attention CNNs could map landmarks on lobster images(for sizing) with an acceptable error of about 2cm. Additionally, siamese networks, that were explored for a black box approach towards uniquely identifying lobsters, were able to achieve a top-3 accuracy of about 84%.

List of Abbreviations Used

AAM	Active Appearance Models
ADAM	Adaptive Moment Estimation
ANN	Artificial Neural Networks
CNN	Convolutional Neural Networks
LSTM	Long Short Term Memory
IoU	Intersection over Union
EFA	Elliptic Fourier Analysis
ML	Machine Learning
MSE	Mean Squared Error
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
ROI	Region of Interest
RNN	Recurrent Neural Networks
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
TP	True Positives
FP	False Positives
VGG	Visual Geometry Group

Acknowledgements

I would like to thank my parents and peers for constant support and motivation towards completion of my thesis. I would thank my supervisors Dr.Mae Seto and Dr.Thomas Trappenburg for their valuable guidance and mentoring.

The efforts of Lawrence Taylor (NovaSpectrum Analytics Inc.) and the Tangier Lobster Company Ltd. are acknowledged for providing the lobster images for this research. We would particularly like to recognize Phil Morgan from Tangier Lobster for his help with the data labelling.

We are especially grateful to the National Lobster Hatchery in the United Kingdom for lending us images of lobsters for our research.

Chapter 1

Introduction

This thesis reports on research and development to improve sustainability and traceability towards the marine stewardship that maintains the ocean's health and ensures the fish or lobster consumed is traceable to a sustainable source. There are ecological and economic sustainability reasons to assess and monitor stocks and stock populations. Traceability aims to uniquely identify each animal for consumer quality purposes. The methods developed here target the Canadian fishery due to the small number of species involved. The aim of the project is to determine the count and species of a catch as well as each species' length and color (measure of quality) for economic sustainability reasons. Another objective is to determine the gender and characteristic dimensions for ecological sustainability. For traceability, species imagery was collected to assess whether say the carapace on a lobster can be uniquely identified.

The Canadian Department of Fisheries and Oceans have expressed interest in such technologies for in-shore waters as they have the same above ecological objectives. The present use of trained human observers can only monitor 1.5-2.0% of the lobster fleet activity not the 20% necessary to manage non-endangered species (lobster) or the 50% for endangered species (cod, cusk, jonah crab). Lastly, value chain stakeholders have articulated their requirement for primary grading (size and quality) while at-sea to guarantee lobster quality and value when it gets to market.

Steps are presented to automate some of the processing that happens at various stages and to characterize lobster traits for lobster traceability. Chapter 3 presents an approach towards separating out fish species such as cod or kusk from a lobster catch. The approach presents a machine learning model that learns to perform species classification and feature localization in images of marine animals. The output of the model is a bounding box around the species in the image. Bounding boxes are typically used in image analysis to isolate targets or regions of interest in an image.

Two types of bounding boxes i.e axis-aligned (aligned with the horizontal and vertical axes of the image) and rotatable, are evaluated on how tightly they enclose the region of interest. If the bounding box is tight then its dimensions approximate the the length and width of target of interest (in pixels).

In Chapter 4, a method is proposed to scale the size of objects in images, in pixels, to their physical size. The method estimates the distance between two graduations of a ruler inserted in the background. This makes it possible to scale the length and width of bounding boxes from pixels to say millimeters for the marine animals in the boxes 3. This technique is generally useful to size targets in an image.

Chapter 5 documents efforts to uniquely identify lobsters by using the lobster images, extracted from images using the bounding boxes, above, as an input to a deep learning model. The model outputs an n-dimensional vector in the Euclidean space for every lobster. The model learns to minimize the Euclidean distance between the output vectors for images of a given lobster and maximize its distance for images of different lobsters. A lobster can then be uniquely identified by matching its image with that of a lobster with the smallest Euclidean distance in the database. This method is applied towards lobster traceability from point-of-catch to market.

In Chapter 6, various methods for mapping morphological landmarks onto the various keypoints on the lobster image are explored. This landmark mapping makes it possible to characterize lobster features such as the length from its eyes to the end of the carapace, claw length, widths, abdomen lengths, etc. The Department of Fisheries and Ocean stipulates a minimum carapace length for a lobster to be legally purchased, sold or possessed[20]. The lobster carapace length, determined using the methods from chapter 4, from the landmark mapping model could be used to detect lobsters that are illegal to catch. This size information is also useful towards monitoring lobster populations for traceability from point-of-catch to market.

In Chapter 7, lobster characteristics such as gender, side of the body the crusher claw is located, first abdominal bulge, posture and tail spread are explored. These attributes, determined from a visual inspection of the lobster, form part of a profile that characterizes a lobster and are helpful towards tracing its point of origin. These attributes are also useful towards monitoring lobster health and indirectly provide an indication of the conditions the lobster was stored in. An abnormally distended

lobster bulge at its first abdomen is an indication it was kept in cramped spaces and thus in poor health.

The methods to detect bounding boxes in Chapter 3 have been used in subsequent chapters to crop or extract the portion of an image that contains the lobster. The sizing method described in chapter 4 is applied to the work documented in chapter 5 to scale a lobster's morphological size in pixels to a standard unit of length. Chapters 5, 6 and 7 describe work to characterize lobster traits towards lobster traceability. Chapter 5 describes a deep learning method that could uniquely identify individual lobsters by assembling an n-dimensional vector, for every lobster, based on multiple attributes. Chapters 6 and 7 characterize qualitative traits such as gender, level of aggression in the posture, etc and quantitative traits such as claw size. All these traits form a lobster database that can uniquely identify individual lobsters and thus, contribute to lobster traceability.

The contributions of this thesis are as follows:

- A comparison of axis-aligned and rotatable bounding boxes has been made towards size estimation of marine animals. A CNN architecture using transfer learning that outperforms existing marine animal classification approaches has been proposed.

- A novel approach is proposed to measure the dimensions of objects in images using a reference object like a ruler. The approach crops out the ruler section using a rotatable bounding box and subsequently, finds pixel distance between ruler graduations.

- Chapter 5 is the validation that lobsters can be uniquely identified based on visible carapace markings and the position of their spines. This was achieved with a pair-wise comparison of different lobsters using a Siamese neural network architecture optimized over several objective functions.

- The contribution of Chapter 6 is two fold: the first is to consider, and prove viable, the use of deep learning towards automating the mapping of landmarks on lobster images. The second is to design and implement a convolutional neural network that uses attention mechanisms for geometric morphometry of lobsters.

- The contribution of the Chapter 7 is two fold: the first is to consider, and prove viable, the use of deep learning towards classification of lobster traits such as gender, abdominal bulge, aggressive posture, etc. The second is to evaluate the use of attention mechanisms on convolutionl neural networks towards improving their accuracies.

Chapter 2

Background and related work

In this section, a background is presented on the deep learning architectures specifically, convolution neural networks (CNN), that are used for classification and regression tasks in later chapters. As well, optimizers such as the adaptive moment estimation or ADAM[41] and stochastic gradient descent or SGD[7] used to learn the weights of the CNNs, will be described. Performance and classification evaluation evaluation used in the thesis are also defined here.

2.1 Convolutional Neural Network

A regular neural network consists of a series of hidden layers. The input to a network is a single vector which is sequentially modified by these hidden layers. Each hidden layer consists of neurons that provide an output value from applying a function to the input values from the receptive field in the prior layer. This function is in the form of a vector of weights and a bias. The hidden layers are also fully-connected layers because each neuron in the one layer is connected to every other neuron in the next layer. Learning is achieved through incremental changes to these weights and bias via back propagation. Convolutional neural networks [45] are a category of neural networks that modify input volumes with convolution layers, pooling layers and fully-connected layers. Convolution layers contain a set of learnable filters that slide across the width and height of the input volume during the forward pass. Pooling layers down sample the spatial dimension of the input volume through averaging or 'max pooling'.

2.1.1 Activation functions

Activation units are one of the building blocks of a neural network which decide if a neuron in the neural network should be activated. The activation function is a non-linear and gives the neural network the capability to learn complex mappings

from input features to network predictions[79]. Three types of activation functions have been used in the thesis.

- Sigmoid function: The sigmoid activation function is given by equation2.1 as

$$f(x) = \frac{1}{1 + exp(-x)}. \tag{2.1}$$

  The sigmoid function is continuous, differentiable and a real-valued function that ranges from 0 to 1. The gradient of the sigmoid function is zero, when the output is close to zero and one, where the weights of the network do not update or are very slowly updated. This problem is also called the vanishing gradients problem.[30] The output of the sigmoid function is not zero-centered because the gradient updates propagate in different directions leading to slow convergence.[61]

- Hyperbolic tan (*tanh*): The tanh function is given by eq. 2.2

$$tanh(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)}. \tag{2.2}$$

  The *tanh* function is differentiable, monotonic and real-valued (ranges from -1 to 1). The advantage of the *tanh* function, compared to the sigmoid function, is that it helps in faster convergence because it is zero-centered [61]. However, *tanh* activations suffer from the vanishing gradient problem, like the sigmoid function, since the gradients are zero for output values equal to zero and one.

- Rectified linear unit (*ReLU*): The *ReLU* activation function is given by eq. 2.3 as

$$ReLU(x) = \begin{cases} x & \text{if } x >= 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

  The *ReLU* is a monotonic function that forces the negative values to zero and avoids the vanishing gradient problem observed in *sigmoid* and *tanh*.[61]. The *ReLU* is computationally faster since it does not involve any calculation of exponentials or divisions.

2.2 Transfer learning and pre-trained CNN architectures

Transfer learning[87] is a machine learning method that takes a model, trained on a large dataset, and transfers its knowledge to a smaller dataset. Transfer learning is useful when there is insufficient data to train a machine learning model. Similarly, training an entire convolutional network from scratch can be difficult since it is unusual to find a sufficiently labelled data set for image classification. It is common to pre-train a convolution neural network on a very large dataset, such as ImageNet[15], which contains 1.2 million images across 1000 categories. The pre-trained model can be used as a *feature extractor* where there is no need to re-train the model. The extracted features are used as input to a new classifier which must be trained from scratch. Alternatively, the pre-trained CNN can be fine-tuned by adding new classifier layers re-training on a new dataset. Pre-trained CNN model architectures that will be considered are described next.

VGG16: This is a deep convolutional network trained by the Visual Geometry Group proposed by Simonyan and Zisserman [69]. The network uses 3×3 convolutional layers stacked on top of each other. The first step is a convolution of the image. Then, the image size is reduced through down sampling (max pooling). This alternates until the two layers become fully- connected. The '16' in VGG16 refers to the number of convolutional layers in the CNN network. Figure 2.1 shows the VGG16 architecture with the fully-connected layers at the end removed. Each convolution block consists of two convolutional layers and one pooling layer.

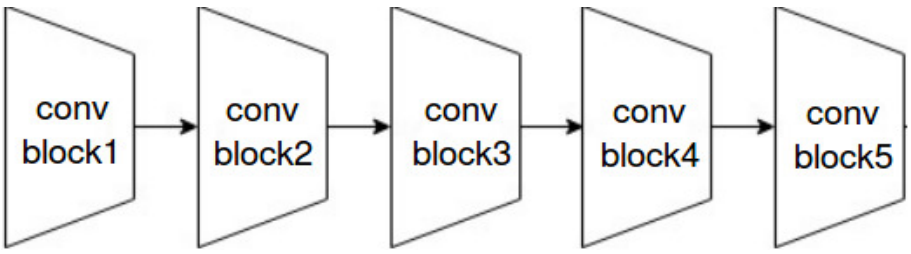


Figure 2.1: The VGG16 architecture with five convolution blocks. The fully-connected layers at the end have been removed. Transfer learning can be achieved by initializing the VGG16 with weights trained on the Imagenet [15] dataset and replacing the fully-connected layers at the end of a regular VGG with extra layers and subsequently, re-training the network with a custom dataset. Transfer learning is commonly used for training where the dataset is limited.

Residual Network: Resnet[28] allows the addition of hundreds of layers to a network and is still able to achieve good performance (in terms of computation effort) compared to VGG. Residual networks use residual mapping, or skip connections, towards a deeper version of the network. At each layer, Resnet is implemented as shown in Eq.2.4

$$y = f(x) + x \tag{2.4}$$

such that *f(x)* is the convolution, or batch normalization, layers and x is the skip connection that allows the gradient to pass backwards, directly. In principle, the gradient could skip over all the intermediate layers and reach the bottom one without being diminished. Therefore, Residual mappings assist in avoiding the vanishing gradient problem that occurs in deep CNNs [30]. Residual networks also use batch normalization layers which are intermediate normalization layers. These layers address the problem of vanishing and exploding gradients.[4][21].

MobileNet: MobileNet [32] uses a 3×3 depth-wise separable convolution which has less computations than standard convolutions with only a small reduction in accuracy. Depth-wise separable convolutions are made up of two layers: depth-wise convolutions and point-wise convolutions. In depth-wise convolutions, filters are applied to each input channel. Point-wise convolution is a 1×1 convolution used to create linear combinations of the output of the depth-wise layer. This two-step method reduces the computation effort and learning model size. The depth-wise convolutions filter the input channels but do not combine them to create new features whereas the point-wise convolutions generate new features.

2.3 Attention Mechanism

In this thesis, an attention mechanism proposed by Rodriguez et al.[67] was used to improve the classification accuracy of the vanilla VGG network. This is especially helpful if the training dataset is small – which it is, in this case. Regular CNN architectures do not deliberately extract detailed features from images. However, the attention mechanism learns to focus on regions of the images that can assist the CNN in learning discriminating features for image classification. This attention mechanism is independant since it can adapt to pre-trained architectures like VGG[69] or ResNet. The process consists of an attention module that can be added after each convolutional

layer *without* changing the underlying information pathways of the architecture. This is helpful since it augments architectures like VGG and ResNet with no additional supervision and can be inserted into any trained network to quickly perform transfer learning.

The attention mechanism is first used in Chapter 6 to improve the accuracy of mapping morphological landmarks onto the various keypoints on the lobster image. The attention mechanism is then used in Chapter 7 to improve the classification accuracy of lobster traits like gender, crusher claw location, etc.

As shown in Figure 2.2, the original CNN can be augmented with attention modules at arbitrary depths. Each attention module contains K attention heads that tap the feature activations at an arbitrary depth. The K attention heads make a prediction based on these feature activations. The original network $output_{net}$ is then corrected based on the output from the attention modules by means of the global attention gates to yield the final output.

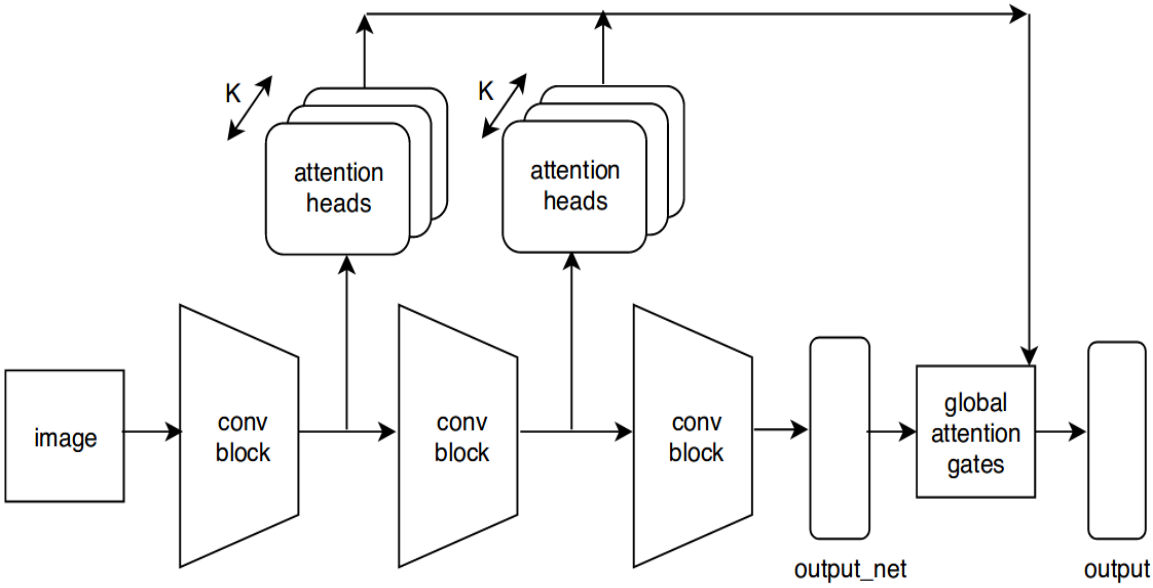


Figure 2.2: The attention mechanism used in this thesis [67]. The feature representations from intermediate VGG layers are passed as input to the attention module. The outputs from the attention heads are combined using the global attention gates.

Rodriguez et al. [67] proposed the attention mechanism for fine-grained classification. However, the attention mechanism has been modified in this thesis for landmark detection. The following equations describe the attention mechanism for

both fine-grained classification and landmark detection.

The features activations, Z_l, from layer l and of dimension $\mathbb{R}^{c\times h\times w}$, such that c is the number of channels of the feature activations, and h and w are the spatial dimensions of the feature activation, are fed to an attention module containing K attention heads. ${W_H}^l$ in Eq 2.5 is the convolution kernel with K filters. The softmax function changes an n-dimensional vector, z, of arbitrary real values to an n-dimensional vector $\sigma(z)$ where $\sum_{i=1}^{K}\sigma_i = 1$.

Z_l is convolved with ${W_H}^l$ and then followed by a spatial softmax. Spatial softmax is a channel-wise softmax operation performed to normalize attention scores across the K attention heads. H_l^k represents the attention score given by the kth attention head to the feature activations at layer l.

$$H_l = spatial\_softmax({W_H}^l \bullet Z_l). \tag{2.5}$$

Here, $\bullet$ refers to the inner product operator. ${W_{O_k}}^l$ in Eq 2.6 is a convolution kernel for an attention head K which is applied on the feature activation Z^l. ${W_{O_k}}^l$ has filters equal to the number of labels i.e the number of classes it needs to classify. Rodriguez et al.[67] uses ${W_{O_k}}^l$ to calculate class probability scores for the classification problem. In case of the landmark detection problem, since this is a regression problem, ${W_{O_k}}^l$ in the model would have the number of filters equal to twice the number of landmarks(x and y coordinate for each landmark). The output dimension of ${O_k}^l$ after convoluting Z^l by ${W_{O_k}}^l$ is $\mathbb{R}^{2\times number\ of\ landmarks\times h\times w}$.

The convolution operation by ${W_{O_k}}^l$ can be performed for all attention heads K in a single pass by setting the number of output filters to be $K\times 2\times number\ of\ landmarks$.

${O_k}^l$ represents the output vector from the feature activations extracted at layer l and the kth attention head.

$${O_k}^l = {W_{O_k}}^l * Z^l \tag{2.6}$$

The o_k^l Eq. 2.7 is obtained by an element-wise product between H_k^l and O_k^l. H_k^l has dimension $h\times w$ and O_k^l has dimension $\mathbb{R}^{number\ of\ labels\times h\times w}$. In case of landmark regression, O_k^l has dimension $\mathbb{R}^{2\times number\ of\ landmarks\times h\times w}$ Therefore, H_k^l is repeated, i.e. $2\times labels$ and for landmark regression H_k^l is repeated, i.e. $2\times number\ of\ landmarks$.

The output o_k^l from Eq. 2.7 is the predicted classification scores from the kth attention head weighted by the attention score H_k^l and spatially averaged over the x, y. In case of landmark detection, The output o_k^l from Eq. 2.7 is the predicted landmarks from the kth attention head weighted by the attention score H_k^l and spatially averaged over the x, y (Eq.2.7).

$$o_k^l = \sum H_k^l \circledast O_k^l \tag{2.7}$$

where $\circledast$ is the element-wise multiplication operator.

The output of the lth attention module o^l (Eq. 2.8) is the sum of outputs from the individual attention heads o_k^l (Eq. 2.7) weighted by $g_{H_k^l}$.

$$o^l = \sum_k g_{H_k^l} o_k^l \tag{2.8}$$

g_H in Eq. 2.9 is obtained by first convolving Z^l with W_g^l. The resulting dimension after convolution with W_g^l becomes $\mathbb{R}^{|H| \times h \times w}$ where $|H|$ is the number of attention modules. The resulting output is multiplied element-wise with H_l followed by a softmax operation.

$g_{H_k^l}$ represents the weight given to the output of each attention head K i.e o_k^l.The weight $g_{H_k^l}$ is a function of the feature activation Z^l and the attention scores H_l.

$$g_H^l = softmax(tanh(\sum_{x,y}(W_g^l \star Z^l) \circledast H_l)) \tag{2.9}$$

The output o^l from each attention module is weighted using candidate values c_l (Eq. 2.10). c_l is a function of the feature activation Z^l.

$$c_l = tanh(W_G Z^l) \tag{2.10}$$

where W_G is the weight given to feature activation Z^l.

The global attention gates, g, (Eq.2.11) are obtained by normalizing the set of candidate scores for all attention modules by means of a softmax function. g_{o^l} represents the weight given to the output predictions from the attention module l. i.e o^l

$$g_{o^l} = \frac{e^{c^l}}{\sum_{i=1}^{|G|} e^{c^i}} \tag{2.11}$$

The $final\_output$, Eq.2.12, of the network is the weighted sum of original output, $output_{net}$, and output from the L attention modules o^l.

$$final\_output = g_{net} * output_{net} + \sum_{l} g_l \cdot o^l \tag{2.12}$$

In the case of classification, the $final\_output$ is fed to a fully-connected layer containing units equal to the number of labels to be classified with softmax activations. In the case of landmark predictions, the $final\_output$ is fed to a fully connected layer containing units equal to the twice the number of landmarks (x and y coordinate).

2.4 Image Augmentation

The datasets used in this thesis are limited in size. However, deep learning frameworks require large amounts of data to train on[48]. Therefore, the datasets have been augmented using the *imgaug* [2] software image augmentation library. The following image augmentations were performed(Figure 2.3): randomly rotated horizontally and vertically; affine transformations like image translation from -10% to 10%; rotations from -45° to 45 °; images sharpened with pixel intensity multiplicative ratios from 0.75 to 1.5; image brightness changed for each RGB channel by adding pixel intensity from -10 to 10 and contrast normalization ratios ranging from 0.9 to 1.10. Pixel intensity values in the images were normalized to be in the range of 0 to 1. Figure 2.3 shows example images of image data augmentation. The augmentation parameters were chosen so they cover variations in target orientation, image brightness, contrast, etc. in the dataset.

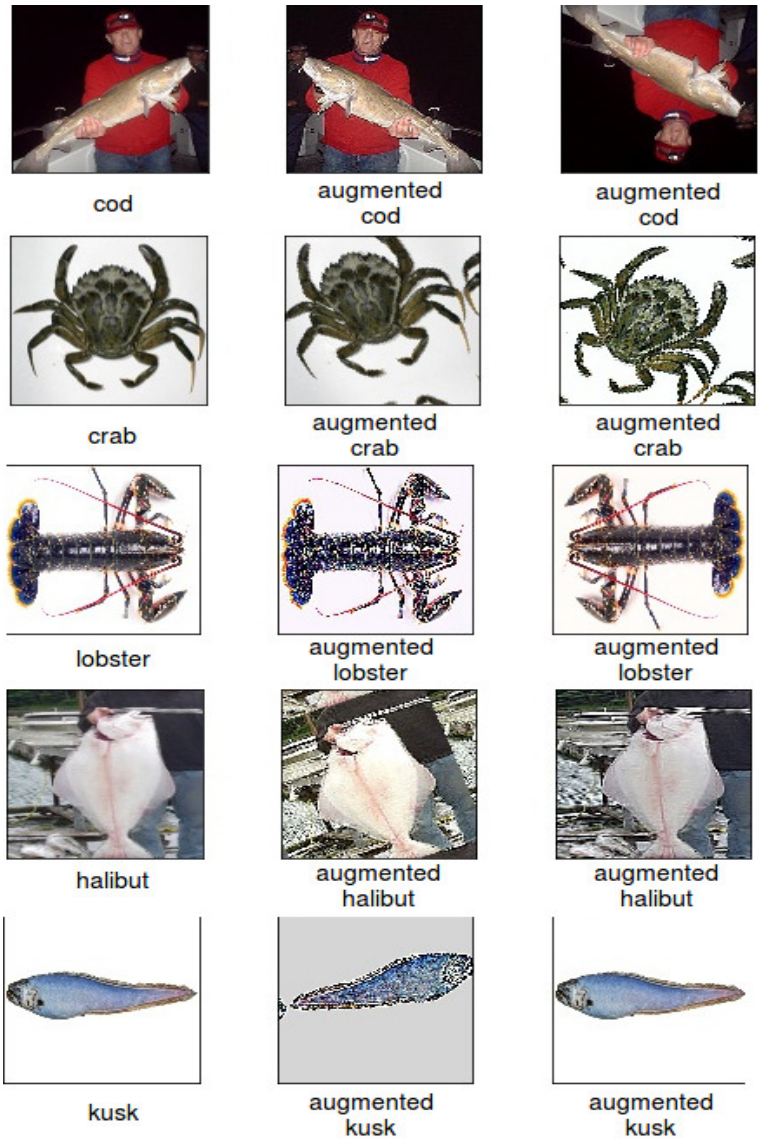


Figure 2.3: Example images generated for each of the marine animals from image augmentation.

2.5 Optimizers

Optimizers are used in deep learning to learn the weights of a model given a dataset. Gradient descent is an example. Gradient descent finds the parameters of a function that minimize the cost function by iteratively taking steps proportional to the negative of the function gradient at a given point. The change in value of the parameter is the gradient of the loss function multiplied by the **learning rate** or step size.

Learning rate: The learning rate is a configurable hyperparameter which controls the rate or speed at which the model learns. The learning rate does not change during the training process. A large learning rate makes the model learn faster but the model might learn sub-optimal weights because of oscillating loss values over epochs. However, a small learning rate might not converge and thus, get stuck on a sub-optimal solution. Therefore, the learning rate should not be too large or too small to be able to find the optimal solution. [22]

2.5.1 Stochastic gradient descent (SGD)

Stochastic gradient is a variant of gradient descent [7] which calculates the function gradient for a small subset of the whole dataset. Gradient descent calculates the gradient of the entire dataset which can be computionally expensive compared to SGD. SGD maintains a single learning rate for all weight updates.

2.5.2 Adaptive Moment Estimation (ADAM)

ADAM[41] is another variant of gradient descent which maintains a different learning rate for each network weight parameter. When using different learning rates for each network parameter, the parameters which get large gradient changes will have their effective learning rates reduced, while parameters which get smaller gradient updates will have their effective learning rates increased. This can possibly lead to a faster convergence to the global extremum The learning rate for each network parameter changes adaptively according to the mean and variance of the previous gradients allowing it to factor in the rate of change of the gradient. Subsequently, this algorithm can perform better on noisy data.

However, Keskar et al.[39] observed that the ADAM optimizer did not converge to the optimum in certain settings and suggested that ADAM had lower generalizability compared to SGD in terms of convergence to the local minimum.

2.6 Performance and Evaluation Metrics

2.6.1 Student t-test

A student t-test is a method of statistical inferencing where a hypothesis is proposed for the comparison of two datasets. First, a null hypothesis is formulated which states that there is no effective difference between the mean values of both datasets. Then, a p-value is calculated which is a measure that determines the probability of rejecting the null hypothesis. Typical selected threshold values are 1 % or 5% (which is the case here). If the p-value is greater than the threshold then there is strong evidence in favour of the null hypothesis. However, if the p-value is less than 0.05 [94] then the null hypothesis can be rejected.

2.6.2 Intersection over Union (IoU)

The intersection over union (IoU) was used to evaluate the accuracy of the axis-aligned bounding boxes. it is defined as the ratio of the intersection (overlap) area between two bounding boxes and the area of union of the two bounding boxes.

2.6.3 Normalized Mean Squared Error

The normalized mean squared error, NME, (Eq. 2.13) is a loss function that can be used as an evaluation metric to compare convolutional neural network model performance. It is defined as:

$$NME = \frac{\sum_{i=1}^{N} \sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}}{N} \tag{2.13}$$

such that N is the number of landmarks, $\hat{x}_i, \hat{y}_i$ are the predicted landmark coordinates and x_i, y_i are the ground truth landmark coordinates.

2.6.4 Binary Classification Evaluation

Binary classification models can be evaluated using accuracy as measured through the $f1$-score. The $f1$-score is especially useful when the classes are unbalanced. It is defined as follows:

$$f_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

such that precision and recall, respectively, are defined as follows: precision = $\frac{\text{TP}}{\text{TP+FP}}$ and recall = $\frac{\text{TP}}{\text{TP+FN}}$ (TP = true positives, FP = false positive and FN = false negatives). The $f1$ score is thus the harmonic mean of precision and recall as defined here.

2.6.5 Penalized Cross-Entropy Loss

Penalized classification imposes an extra cost on a CNN model for making classification errors on the minority class during training. This is applicable in binary classification cases where the training dataset is heavily weighted towards one class more so than the other.

The penalized cross-entropy loss (Eq. 2.14) is defined as:

$$ce(y, \hat{y}) = -\sum_{i=1}^{\text{n}} \alpha_i y_i \log(\hat{y}_i) \tag{2.14}$$

such that y_i is the image ground truth label, $\hat{y}_i$ is the prediction confidence score and α_i is the penalty given for predicting class i. α_i is inversely proportional to the number of examples of class i in the dataset.

Test Accuracy

The simple definition for test accuracy used here is the ratio of correct predictions normalized to total predictions:

$$test\ accuracy = \frac{correct\ predictions}{total\ predictions}$$

Chapter 3

Species identification and localization in marine animal images

3.1 Introduction

A catch from a fishing trawler can contain multiple animal species. They are usually manually sorted, by species, then shipped to different factories for further classification and processing. As part of this, there is a requirement to sort these animals based on their physical maturity and size.

Sorting and classifying marine catch (fish, in this example) on the basis of their dimensions and species using pattern recognition algorithms is proposed. Consequently, it is possible to automate the sort and classification steps of the fish processing with computer vision to improve traceability, profit margins and product quality. The collateral benefit is that the size distribution by species, of a catch, also has ecological significance towards characterizing the fish population and its evolution. There are earlier efforts in fish classification [66][47][62] however, there is much less on automated estimation of the fish dimensions. This chapter explores methods to simultaneously isolate (localize) the fish in a static image, from amongst other objects, through a bounding box that contains the fish. Then, it determines its dimensions and classifies the fish by species.

The next section is a description of existing work related to fish species classification and sizing techniques towards this objective with intent to apply it to lobsters.

3.2 Related work

Rathi et. al. [66] performed classification of fish species. They perform pre-processing steps of Otsu's binarization [86], dilation and erosion to improve the quality of the image. They add the pre-processed image as the fourth channel to an already existing RGB image and use CNN for classification of fish images. Rathi et al. have used a

custom CNN architecture which they trained from scratch. The work in this thesis uses pre-trained CNN architectures which uses transfer learning to leverage the large amount of training time from a different project.

White et. al. [84] determines the orientation of the fish based on the moments of the polygon spanned by the fish silhouette. They determine the fish species based on colour and shape. Then, they determine the length of the fish by mapping eight points on the detected outline of the fish. Contrary to White et. al., this thesis plans to determine the width and height of the fish by creating a bounding box around the fish. The bounding box technique is robust to various types of fish. This technique will be applied in this thesis to lobsters.

Larsen et. al. [47] perform classification of fish species based on shape and texture features. They estimate the parameters of an active appearance model using geometric and texture-based features. Subsequently, they apply principal component analysis (PCA) to these model parameters which yield features that they apply linear discriminant analysis [38] to for an accuracy of 76%. Their method is dependant on building a separate model apart from the learned one for each species of fish. The CNN architecture from this thesis creates an intrinsic model of the species for classification and does not need a separate model.

Hsieh et al. [33] propose a technique to measure a tuna fish's snork to fork length using a Hough transform. The longest line measured by Hough transform in the image is a measure of the fish length. They transform every point in image space to Hough space. The co-linear points in the image space are presented in the Hough space. Therefore, the weight of the largest peak must be the fish length. In other words, this technique measures the longest length of an object in the image. However, this will not work well if there are objects longer than the fish in the image whereas the technique developed for this thesis do not have that limitation.

Costa et. al. [11] were able to estimate the size of the fish using external shape analysis. First,they used the Canny operator in MATLAB [8]to create a binary image. The Canny operator smooths the image through Gaussian convolution and applies a two-dimensional first derivative operator to highlight regions with edges. The next step was to create 200 equally spaced outline points along the perimeter of the animal. The shape of each fish was then analyzed by elliptic Fourier analysis (EFA) on the

coordinates of each of the 200 outline points. EFA is based on Fourier decompositions of the incremental changes in each of the x and y coordinates [12]. Their sizing technique aligns with the thesis objective of localization of the fish in the image but the limitation of this method is that the binary image is specific to a species and sensitive to variations within a species.

Ogunlana et. al. [62] extracted fish sizes like the body length and width and the five fin lengths; namely anal, caudal, dorsal, pelvic and pectoral. Then, they used support vector machines (SVM) for species classification with a 78.59% accuracy, which was significantly higher than what was obtained with artificial neural networks, k-nearest neighbours and k-means clustering-based algorithms on the same dataset. Their species classification method aligns with the thesis objective of marine species classification but this approach does not take into account the color and texture features of the marine animal which would further help classify it.

Hasija et. al. [27] use image sets to classify fish species using graph embedding discriminant analysis unlike state-of-the-art methods which operate on single images. Multiple views of the fish, as in this thesis' approach, might achieve a better classification of the fish's species. Their classification method aligns with the thesis objective of marine animal species classification. However, their algorithm is not immune to distortion caused by noisy images which have a classification accuracy of 76 %.

Spampinato et. al. [71] perform fish species classification by extracting texture and shape features. The texture features were extracted using gray-level histogram, spatial Gabor filtering and properties of the co-occurrence matrix. The shape features were extracted by using a histogram of Fourier descriptors of boundaries and curvature scale space transformations. The species classification method by Spampinato et. al. aligns with our goal of fish species classification. However, they do not utilize the color features of the fish unlike the method developed in this thesis. The sizing method used by Costa et al. also used Fourier descriptors which is similar by Spampinato et. al. for fish species classification.

Benson et. al. [5] make use of Haar classifiers to count and classify the Scythe Butterfly fish captured in underwater video images. The Haar classifiers were trained to detect only a single species of fish. The proposed method trains on multiple fish species.

3.2.1 Overall comparison with existing approaches

The methods described above, except Rathi et al[66] do not use convolution neural networks for classification and localization of marine animals. CNNs learns what features to extract compared to using classical computer vision methods which use hand-crafted features like Haar features as used by Benson et al[5], hand engineered features like body length and width as used by Ogunlana et. al. or shape and texture features by Larsen et al[47]. The features learnt by CNN are generic and independent of any specific classification task and can be learned directly from observations of the input images[48]. Rathi et al[66] use CNNs for fish classification but they train the CNN from scratch on limited data (not too different from the approach in this thesis). White et al.[47] determine the length of the fish by mapping eight points on the detected outline of the fish. Similarly, the method proposed by Larsen et al.[47], Costa et. al.[12] and Benson et. al.[5] is dependant on building a separate model for each species of fish. The CNN architecture for this thesis is able to determine length and width for any number of species of marine animals. Hsieh et al.[33] et al. determine fish length by finding the length of the longest object in the image using Hough transform. The CNN architecture for this thesis identifies the species in the image and then calculates the species length independant of whether there is a longer object in the image. The approach for fish classification proposed by Ogunlana et. al.[62] does not use colour features unlike the CNN architecture proposed in this thesis where features learnt are generic and independent of any specific classification task. As shown in further section 3.4, the CNN architecture proposed in this thesis is shown to be better in terms of test accuracy compared to the SVM technique used by Ogunlana et. al.[62].

In the next section, an approach that uses pre-trained CNN for simultaneous classification and localization has been described. The VGG16 was trained on the ImageNet[15] dataset. Using the pre-trained CNN for feature extraction is a form of transfer learning as described in section 2.2. The experimental methodology is described next including the dataset details, feature extraction techniques and model architectures used.

3.3 Experimental Methodology

The CNN developed is trained to predict the species of the animal, in an input image, as well as its location in the image. To predict the location of the object of interest in the image, axis-aligned and rotatable bounding boxes have been compared for tightness of the bounding box around the object. The **axis-aligned** bounding box is aligned to the axes of the image coordinate system as shown in Figure 3.1. The axis-aligned bounding box is parameterized by the coordinates of the lower left and upper right corner of the rectangle. The size and aspect ratio of the bounding box does not tightly span the real shape of the target when it is not aligned to the coordinate axes of the image. For example, when the fish lies along the diagonal of the image, the axis-aligned bounding box does not approximate well, the size of the object. The **rotatable bounding box** is parameterized by the centroid of the rectangle and its length, width and body axis orientation. The size of the rotatable bounding box is a better indicator of the object size compared to axis-aligned boxes since the freedom to rotate enables the box to span the outline of the target object more tightly. The rotatable bounding boxes in Figure 3.2 are a tighter fit for the object of interest.

Figure 3.1: Axis-aligned bounding boxes cannot tightly span the object of interest if it is not aligned to the image's horizontal or vertical axes. Consequently, the box dimensions are not the best measure of the object's.

Figure 3.2: Rotatable bounding boxes are a tighter fit for the object of interest and thus a better measure of its dimensions.

3.3.1 Dataset

Static images of marine animals for five species namely Jonah crab, lobster, halibut, cod and cusk were culled from the internet (Table 3.1). The first stage in the dataset processing was to manually draw bounding boxes around the object (its region of interest or ROI) with the *labelImg* software annotation tool[81] within each image (Figure 3.1). Then, the second stage was to manually draw rotatable-bounding boxes[54] around the object's ROI with the *roLabelImg* software annotation tool.[9] (Figure 3.2). Since the dataset is limited in size, image augmentation techniques as mentioned in Section 2.4 were used to increase the training dataset.

Table 3.1: Distribution of collected marine animal images

label	cod	crab	halibut	cusk	lobster
count	724	503	913	459	631

3.3.2 Feature Extraction

For feature extraction, pre-trained networks of VGG16 [69], VGG19 [69], Resnet [28] and MobileNet [32] were used. The weights used for the pre-trained networks are those from the ImageNet dataset [15]. The deeper layers of the pre-trained networks were made trainable to improve the accuracy. The pre-trained networks branch out into a regression head and a classification head.

3.3.3 Size Estimation

The width and height of the bounding box is a measure of the marine animal's size in pixels but it still needs to be scaled to a standard unit of length. To achieve this, a ruler is inserted in the image field-of-view. The ruler is detected in the image (Figure 3.3)[44] background and determines the length of a pixel and scales the width and height of the bounding box to this.

Figure 3.3: Automated size estimation using localization and ruler detection. Ruler detection determines the physical length that a pixel represents.

3.3.4 Model Implementation

Figure 3.4 shows the architecture of the developed learning model. The classification head contains a fully-connected layer containing 4 units followed by another fully-connected layer of 5 units. For the first fully-connected layer, the test accuracies were similar for neural network units greater than or equal to 4. Fig 3.5. The output of the classification head are neural network units equal in number to the number of distinct animal species considered (5, here) where each neural network unit gives the probability that the image belongs to that species.

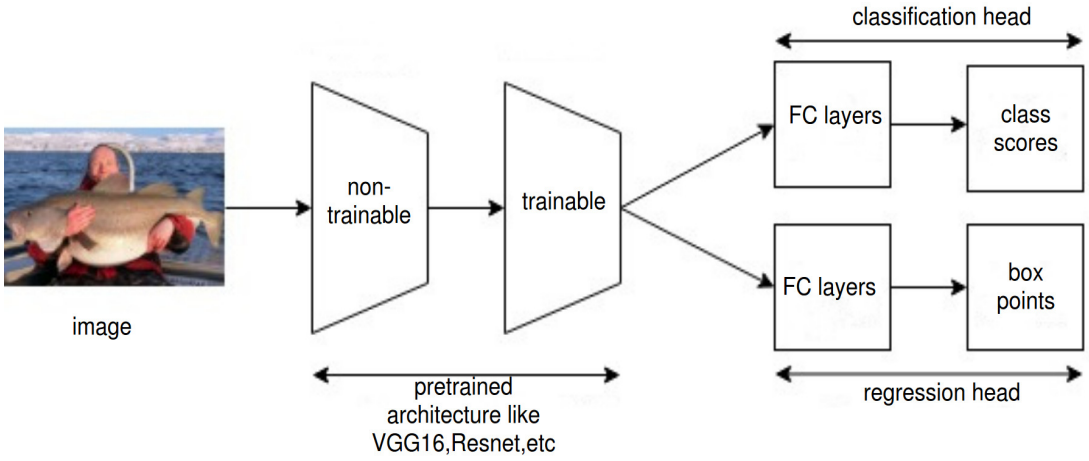


Figure 3.4: Convolutional layered architecture with the regression and classification heads for simultaneous localization and classification of objects in images.

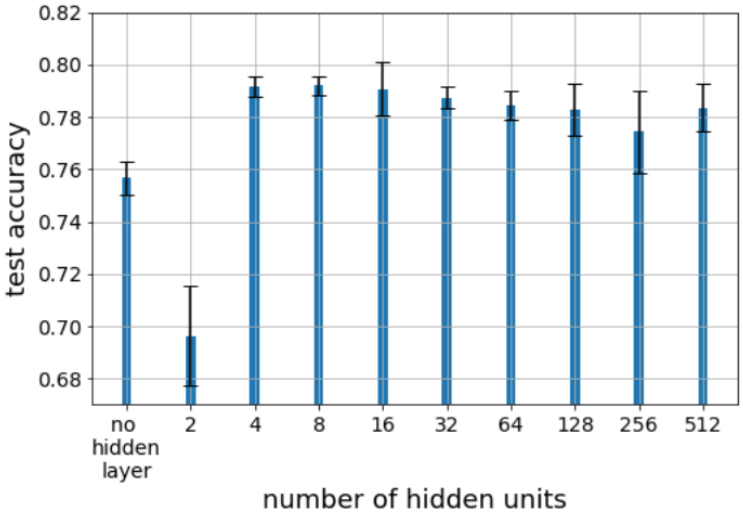


Figure 3.5: The test classification accuracy remains similar for number of hidden units greater than or equal to 4. The accuracy with no hidden layer is less than test accuracies with units greater than or equal to 4.

The regression head contains a convolution layer with 256 3×3 filters followed by a 4×4 max pooling layer. The IoU metric was observed to increase up to 256 filters Fig 3.6. The final layer of the regression head is a 1×1 convolution layer. The output of the regression head are localization coordinates for the object. In the case of axis-aligned bounding boxes, the coordinates are (x_1, y_1, x_2, y_2) where (x_1, y_1) is the lower left corner and (x_2, y_2) the upper right corner of the bounding box.

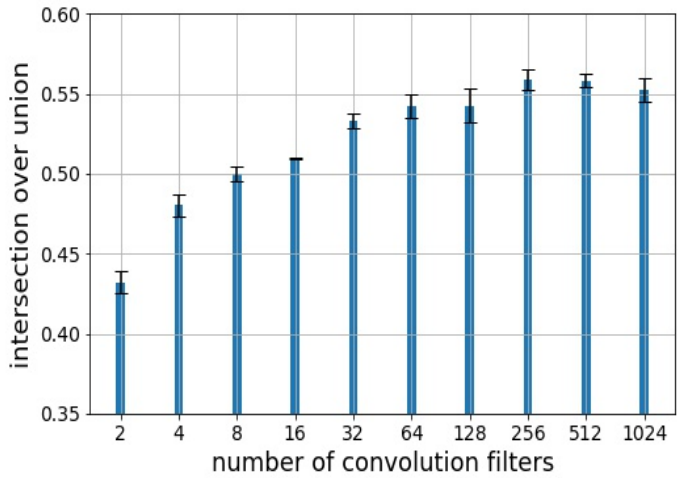


Figure 3.6: The IoU metric increases with increasing number of convolution filters until it saturates at 256.

The size of the animal can also be estimated by creating a rotatable bounding box around it. A rotatable bounding box fits an object oriented at an arbitrary angle to the horizontal, better, than an axis-aligned one. The regression coordinates for a rotatable bounding box are (x_c, y_c, h, w, θ) where (x_c, y_c) are the coordinates for the centroid of the bounding box, (h, w) are its height and width, respectively, and θ is the box's orientation relative to the horizontal axis.

3.3.5 Ensemble Architecture

Ensemble learning uses multiple models to attain better predictive performance than that obtained by any one model alone.[93]. Classification performance can be increased by combining the predictions of multiple weak models instead of training a single strong one.

Different ensemble architectures were used to process the outputs from the regression and classification heads.

The results from the *classification heads* of each of the CNN architectures (VGG16, VGG19, Mobilenet and Resnet) were concatenated. Then, the resulting 20-dimensional vector from concatenation of 5-dimensional (number of classes) vectors of four architectures each, was sent to the ensemble CNN for classification. The ensemble CNN has a fully-connected layer of 5 units (equal to the number of species).

Similarly, the output from the image *localization heads* of each of the CNN architectures (VGG16, VGG19, Mobilenet and Resnet) were concatenated. The resulting vector (20-dimensional for the axis-aligned and 25-dimensional for the rotatable boxes) was sent to the ensemble CNN for object localization. The ensemble CNN has a fully-connected layer containing neural network units equal to the number of localization parameters (4 for axis-aligned and 5 for rotatable bounding boxes).

3.3.6 Training

Losses

Loss is used in the training to obtain the best weights for a model. Loss is optimized (minimized) in the training through adjusting the CNN weights. The cross-entropy loss (Eq.3.1) was used for the classification head and the mean squared error loss

(Eq.3.2, 3.3) for the regression head. With the cross-entropy, ce, loss:

$$ce(y, \hat{y}) = -\sum_{i=1}^{\mathrm{n}} y_i \log(\hat{y}_i) \tag{3.1}$$

y_i is the ground truth label of the image and $\hat{y}_i$ is the predicted species score.

For the mean squared error loss for the axis-aligned bounding boxes:

$$\begin{aligned} mse_{ax} = -\sum_{i=1}^{\mathrm{n}}((\hat{x}_1^i - x_1^i)^2 + (\hat{x}_2^i - x_2^i)^2 + \\ (\hat{y}_1^i - y_1^i)^2 + (\hat{y}_2^i - y_2^i)^2) \end{aligned} \tag{3.2}$$

(x_1^i, y_1^i) and (x_2^i, y_2^i) are the ground truth coordinates of the lower left and upper right corners of the ith axis-aligned box and $(\hat{x}_1^i, \hat{y}_1^i)$ and $(\hat{x}_2^i, \hat{y}_2^i)$ are the predicted coordinates of the lower left and upper right corners of ith axis-aligned bounding box.

Eq. 3.3 is the mean squared error loss for the rotatable bounding box:

$$\begin{aligned} mse_r = -\sum_{i=1}^{\mathrm{n}}((\hat{x}_c^i - x^i{}_c)^2 + (\hat{y}_c^i - x^i{}_c)^2 + \\ (\hat{h}^i - h^i)^2 + (\hat{w}^i - w^i)^2 + (\hat{\theta}^i - \hat{\theta}^i)^2) \end{aligned} \tag{3.3}$$

such that (x_c, y_c) are the coordinates of the centroid of the ith box. (h,w,θ) are the height, width and angle (relative to the horizontal) of the box. $(\hat{x}_c, \hat{y}_c)$ are the predicted coordinates of the center of the ith box. $(\hat{h},\hat{w},\hat{\theta})$ are the predicted height, width and angle of the box relative to the horizontal.

The loss function used for training is the sum of the mean squared error from the regression head and the cross-entropy loss error from the classification head. The models were trained with a patience factor of 50 since it was observed that the test accuracies did not improve beyond 50 epochs. The evaluation of the resulting model is discussed in the next section.

Figure 3.7 shows the change in IoU value with increasing epochs while training. This trend is indicative of a good model.

Five fold cross validation [43] was used for model evaluation. The original training dataset was divided into five folds. At each iteration, four folds were used for training and the fifth for testing and evaluation. The data augmentation described earlier was performed on the training set but not on the test and evaluation set.

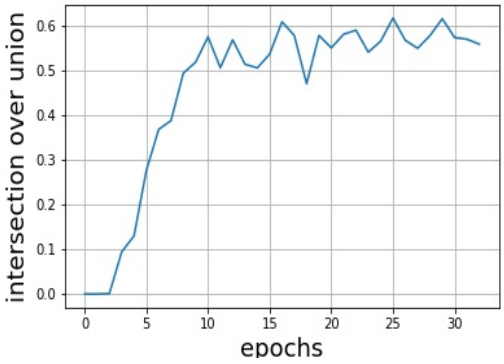


Figure 3.7: IoC metric change for Resnet architecture. IoC is intersection over union. The IoC increases during training time and converges to 0.6. A higher IoC score indicates better localization accuracy.

3.4 Results

Figure 3.8 shows a comparison of test classification accuracy between support vector machines[37] and ensemble CNN described in section 3.3.5. Both the support vector machines and ensemble CNN were trained on the dataset described in section 3.3.1. The ensemble CNN showed better classification accuracy than the support vector machines.

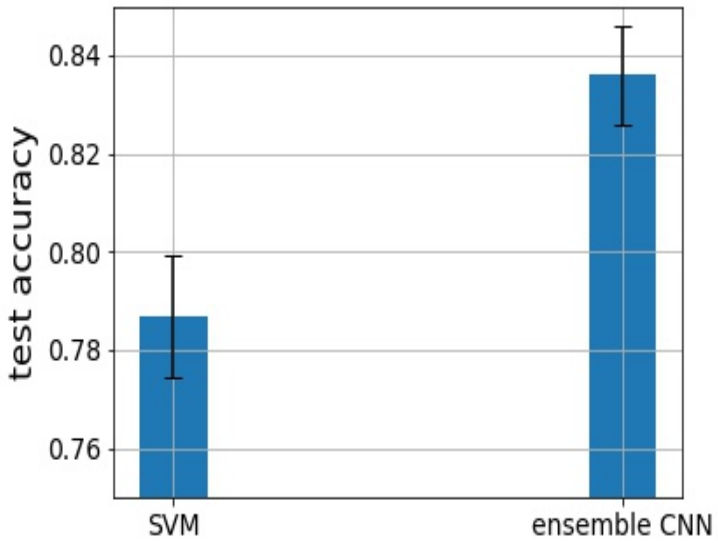


Figure 3.8: The test classification accuracy using ensemble CNN is better than Support vector machines using the dataset described in section. 3.3.1

The height and width of the object in pixels were determined in the images. To compare the performance between axis-aligned (Table 3.2) and rotatable (Table 3.3) bounding boxes, the mean absolute error in pixels of the height and width and the predicted height and width was calculated (Table 3.4)

Table 3.2: Performance accuracy of CNN architectures using *axis-aligned* bounding boxes. The classification accuracy of the ensemble architecture is 81% which is better than classification accuracies of the individual CNNs. The ensemble CNN gives a test IoU score of 0.57 which is only marginally better than the individual CNN IoU scores.

CNN model	localization		IoU		classification	
	train	test	train	test	train	test
VGG16	0.92	0.78	0.79	0.56	0.98	0.76
VGG19	0.93	0.79	0.79	0.56	0.98	0.75
Resnet	0.80	0.77	0.58	0.54	0.73	0.76
MobileNet	0.85	0.80	0.65	0.57	0.59	0.59
ensemble	0.93	0.81	0.78	0.57	0.99	0.81

Table 3.3: Performance accuracy for CNN architectures using *rotatable* bounding boxes. The classification accuracy of the ensemble architecture is 83% which is better than classification accuracy of the individual CNNs. The ensemble CNN gives a localization accuracy of 0.80 which is only marginally better than the individual CNN localization accuracy scores.

CNN model	localization		classification	
	train	test	train	test
VGG16	0.93	0.76	0.94	0.76
VGG19	0.94	0.79	0.90	0.73
Resnet	0.67	0.6	0.70	0.73
MobileNet	0.87	0.76	0.58	0.56
ensemble	0.95	0.80	0.99	0.836

Table 3.4: The mean absolute error (in pixels) for width and height using axis-aligned and rotatable bounding boxes. The rotatable bounding boxes show lower mean absolute error in the predicted height and width which was not unexpected.

CNN model	axis-aligned		rotatable boxes	
	height	width	height	width
VGG16	39.72	59.72	20.99	16.01
VGG19	39.98	61.04	18.94	16.69
Resnet	40.04	65.61	27.96	22.03
MobileNet	40.58	50.58	20.64	16.78
ensemble	40.57	56.95	17.19	13.00

Table 3.4 compares mean absolute error in predicted height and width (in pixels) between the two types of bounding boxes. The rotatable bounding boxes have notably lower mean absolute errors than the axis-aligned ones. This suggests rotatable bounding boxes are a better measure of the target height and width which is validation.

Figure 3.9 shows the confusion matrix for species classification. The classifier misinterprets some cusk images as cod. However, that is not unexpected as cusk are a type of cod.

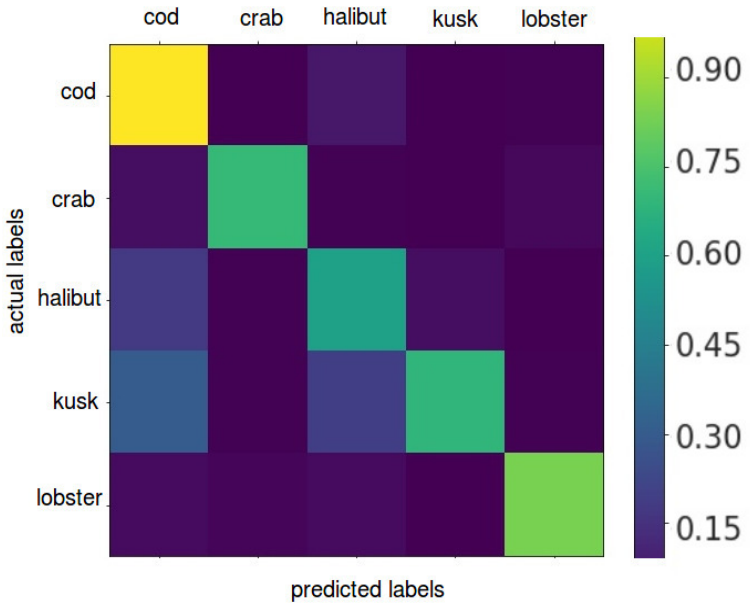


Figure 3.9: Confusion matrix for the Resnet classification. The diagonal cells show larger values indicating better classification accuracy for each label.

Figure 3.10 shows the confusion matrix for species classification using the support vector machine classifier. The predictions using support vector machines have lower percentage number of true positives compared to the CNN ensemble classifier. The cod is predicted with a test accuracy of 76% compared to ensemble classifier(which predicts cod with accuracy of 95%). In the worst case, the test accuracy for halibut and kusk is at least 71% and 68%, respectively. Generally, the ensemble CNN gives better classification between species for all labels.

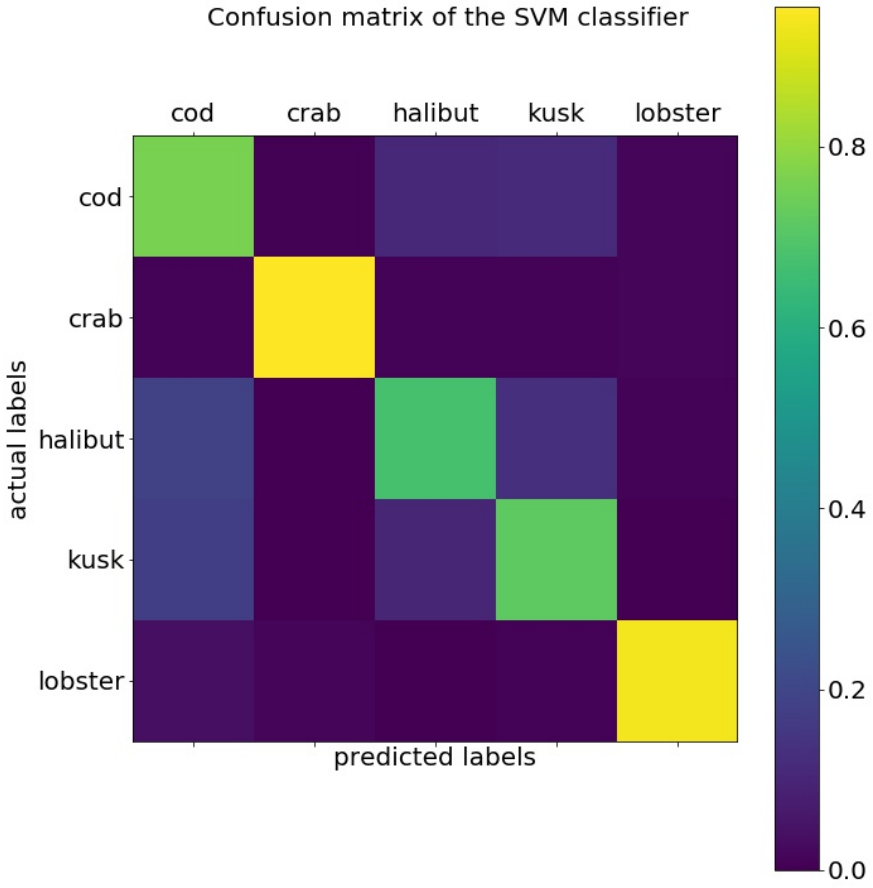
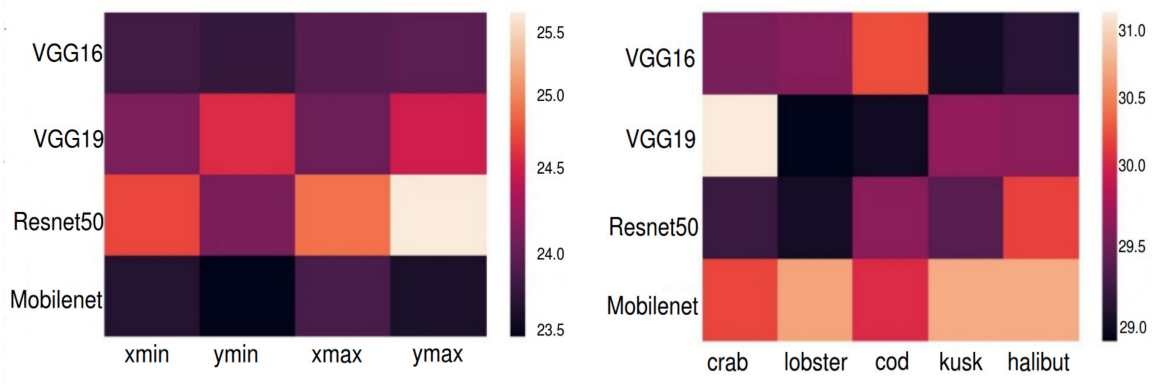


Figure 3.10: Confusion matrix for the Resnet classification using Support vector machines. The diagonal cells show larger values indicating better classification accuracy for each label.

Figure 3.11a shows the weights learned by the ensemble architectures to localize marine animals in an image. Lighter colors indicate a greater weight. The ensemble architectures favour Resnet50 because of better localization accuracy compared to VGG19 and Mobilenet.

Figure 3.11b shows the weights learned by the ensemble architectures for classification of fish species. Again, the lighter colors indicate greater weights. Note, the confidence in prediction of kusk and halibut is high for the Mobilenet ensembling architecture prediction. Kusk and halibut are most easily distinguished fish types from the other marine animals. Halibut is easily distinguished with its flat and long body. Kusk is recognized from its unique top fin.



(a) Weights assigned to CNN architectures to predict the bounding box corners that localize a species in an image. (xmin, ymin) are the left bottom coordinates and (xmax, ymax) are top right coordinates, respectively.

(b) Weights assigned to the CNN architectures for prediction of labels in the species classification

Figure 3.11: Weights assigned to ensemble architectures for prediction of: (3.11a) bounding boxes and (3.11b) species classification.

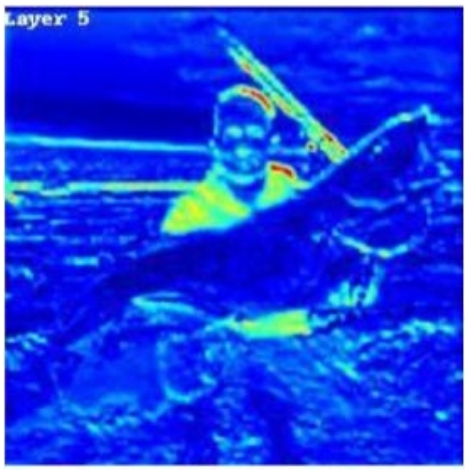
The ensemble classification accuracies and localization metrics are better than individual CNN architectures for both axis-aligned and rotatable bounding boxes. The ensemble classification had some value.

Visualization

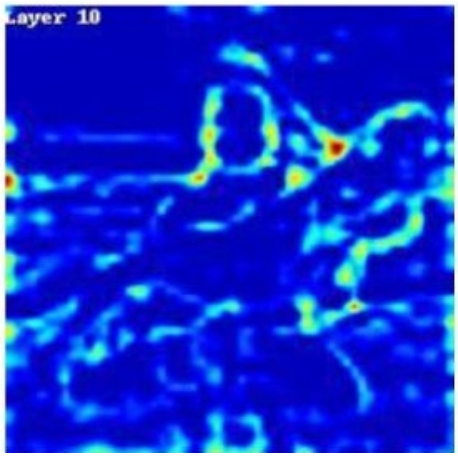
Figure 3.12 shows heat map images that portray the activation of the convolutional layers. This visualization is useful in understanding the image areas that the CNN focusses on to make a prediction. As shown in Figure 3.12d, the CNN focuses more on the fish area. Note, the final layer shows higher 'temperatures' around the kusk belly area and fin indicating the regions i.e the tail fin that discriminate the kusk from the other fish. This is useful to ensure the CNN is learning generic fish features

and not overfitting to a unique identifier like a tag or person in the background.

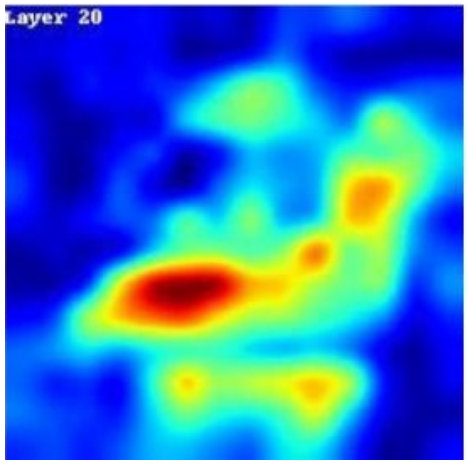
(a) Image with kusk fish in the centre.



(b) Feature activations from layer 5 focus on image edges and corners.



(c) Feature activations from layer 10 focus on image edges and corners.



(d) Feature activations from layer 20 focus on the kusk fish. The heat map focuses most on the tail (red area)
.

Figure 3.12: Visualization of convolutional layer feature activations when the CNN is fed the input image (3.12a). Layers (3.12b) 5 and (3.12c) 10 focus on the image edges and corners and (3.12d) layer 20, the decision-making layer, focuses on the fish body.

3.5 Concluding Remarks

This chapter reports on work that considers, and proves, the viability of using pattern recognition towards automating species classification and sizing of marine animals.

The pre-trained weights used in the convolution neural network were based on those used in ImageNet which contained lobster, crab, and several types of fish though not the cod, cusk and halibut that were of interest. The ensemble CNN architecture outperforms existing approaches in fish classification such as SVM as proposed by Ogunlana et. al. as shown in Figure 3.5.

The work evaluates axis-aligned (to the horizontal and vertical) and rotatable bounding boxes for marine animal localization in static images as well as classification based on size estimates. From an analysis of the mean absolute error in bounding box heights and widths, it was clear that rotatable bounding boxes perform notably better. Therefore, rotatable bounding boxes yield a better estimate of the marine animal height and width. They will be used in the subsequent analysis on lobsters for this thesis.

Future work builds on the achievements to date to collect and prepare more training data for the specific species of interest to increase the model accuracy. With the methods described in the current chapter, it is possible to determine the size of a marine animal in an image in terms of the height and width of its bounding box (in pixels). However, in order to determine the actual size of the marine animal, it is necessary to scale these pixel displacements to a standard measure of length (e.g. inches or centimeters). In the next chapter, a technique to do just that was developed and presented.

Chapter 4

Calculating Object Size in Images

A common way to determine the size of objects in images is to place a reference object of known size, like a coin, in the image. Then, it is possible to determine the number of pixels per unit length by scaling to the reference object in pixels against the known length of the reference object. This ratio helps determine the object sizes in the image. However,the known object such as a coin might be partially occluded by other objects in the image. In this case, the height and width of a bounding box might not be a good measure for the dimensions of the known object since the bounding box might detect the object partially and there will be mismatch between the known size and the detected size. However, if a ruler is used as the known object, then it is sufficient to calculate the distance between graduated markings to determine the scale of the image. In this chapter, a method is proposed to detect ruler graduations in the image and then subsequently, determine the distance between consecutive graduations. The next section explores existing work related to detecting ruler graduations in image backgrounds.

4.1 Background and Related Work

Ueda et. al. [82] created a method to detect graduations on a ruler by transferring the image to the frequency domain using the digital Fourier transform. However, the algorithm was susceptable to noisy images. Zambanini et. al. [89] developed an automatic method to determine the diameter of historical coins. They performed a Fourier analyis on the pattern produced by a ruler positioned next to the coin. However, since this method relies on finding frequencies of repeating patterns in the image, it is susceptible to other repeating patterns in the image such as floor tiles. Bhalerao et. al. [6] used discrete Fourier transforms to find the orientation of the ruler in the image. Subsequently, they fit a sine curve to the determine the ruler spacings in the image. Konovalov et. al.[44] were able to detect the absolute size of

barramundi fish in images by detecting a ruler in the image background. Konovalov et. al. used fast Fourier transforms and achieved a precision of 98% on a dataset containing images of 445 images of barramundi fishes. The methods by Bhalerao et al[6] and Konovalov et. al.[44] are also susceptible to confusion with other spatially repeating patterns in the image.

In this chapter, another technique is proposed and developed where the ruler section is first cropped from the image using a CNN. Then, this cropped image is overlaid with a grid and each box in the grid is processed separately to detect the ruler graduations. Unlike the existing work of Bhalerao et al[6], Konovalov et. al.[44] and Zambanini et. al. [89], this method is more robust against (less likely to be confused with) other spatially repeating patterns in the image. The next section describes this method.

4.2 Methodology

4.2.1 Locating Ruler Graduations

The ruler can be localized within the image using a CNN that can find the ruler portion with an axis-aligned or a rotatable bounding box around the detected ruler. A rotatable bounding box is a better fit for a ruler placed at an arbitrary angle within the image. In this section, a comparison is performed between the error resulting from using axis-aligned and rotatable bounding boxes. This includes details of the CNN trained to detect ruler graduations, the dataset, the model architecture and the post-processing techniques used on the cropped image from the model to find the inter-graduation distances.

Dataset

A dataset containing 350 images of lobsters along with a ruler was annotated with axis-aligned 4.1 and rotatable bounding boxes 4.2 using a *labelImg* tool[59]. Since the dataset is limited in size, image augmentation techniques as mentioned in Section 2.4 were used to increase the training dataset.

Figure 4.1: Detecting axis-aligned bounding boxes around the ruler in the image. Axis-aligned bounding boxes do not tightly fit the ruler when it is at an angle to the horizontal in the image.

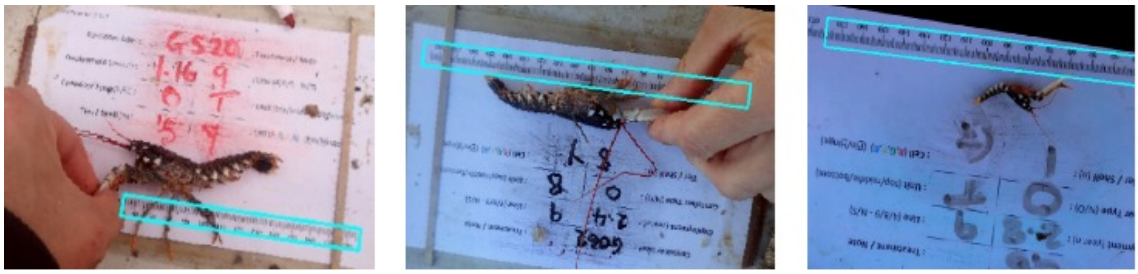


Figure 4.2: Detecting rotated bounding boxes around the ruler in the image. Rotated bounding boxes tightly fit the ruler better when the ruler is at an angle to the horizontal in the image.

Model architecture

Transfer learning using the pre-trained VGG16 convolution network (Figure 2.1, Section 2.2) was applied to the detection of the ruler portion in the image. As shown in Figure 4.3, a convolution layer with 64 3×3 filters followed by a 4×4 max pooling layer was appended to the pre-trained VGG. This was followed by another convolution layer with 64 3×3 filters followed by a 4×4 max pooling layer. The intersection over union was observed to increase until up to 64 filters and did not improve much beyond that (Figure 4.4). The intersection over union was better with two convolution layers, than one, and did not improve notably with three (Figure 4.5). The final layer is a 1×1 convolution layer containing output channels equal to the four corners of an axis-aligned bounding box around the object as defined by the lower left and upper right coordinates. The input image is then cropped using the predicted bounding box coordinates of the network. The mean squared error loss 3.2 and 3.3 for axis-aligned and rotatable bounding boxes, respectively, were used.

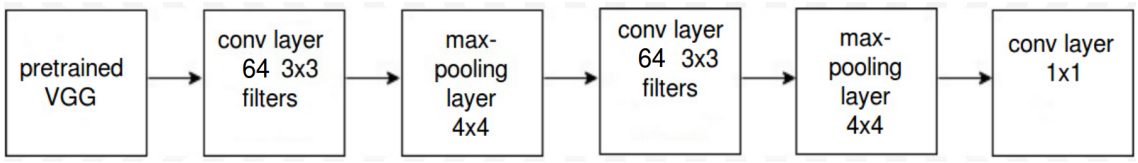


Figure 4.3: Transfer learning using pre-trained VGG16 network: The pre-trained VGG-16 network (Figure 2.1) is appended with two convolution layers and two max-pooling layers followed by a convolutional layer with 1×1 filter (4 filters for axis-aligned and 5 filters for rotatable bounding boxes).

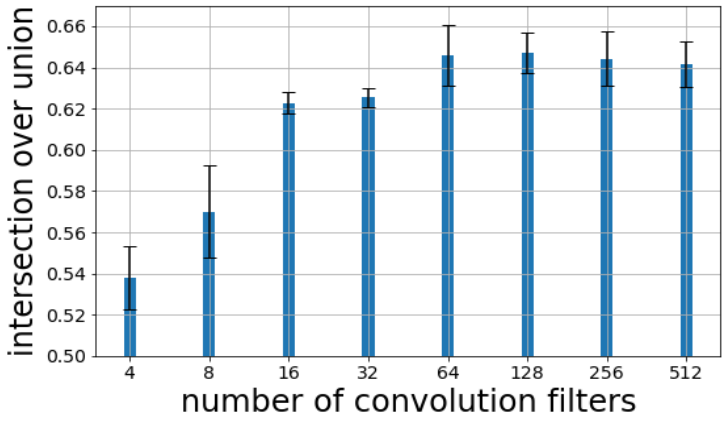


Figure 4.4: The IoU metric increases with increasing number of convolution filters until it saturates around 64.

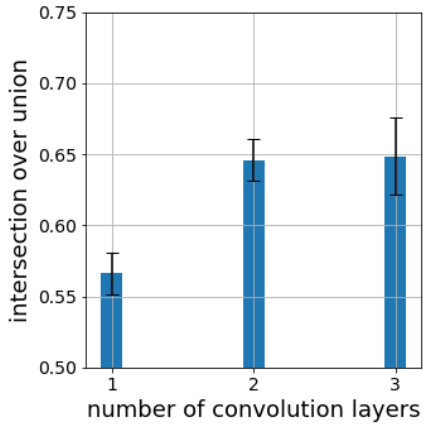


Figure 4.5: The IoU metric increases with number of convolution layers and saturate at around 2. This justifies using two convolution layers.

4.2.2 Detecting Ruler Graduations in Images

The cropped RGB image is first converted to a grayscale image. This is followed by conversion to a binary image. Different thresholding techniques for binarization such as adaptive Gaussian, Otsu and mean value were explored. The cropped image is then divided into blocks of 100 × 100 pixels. The block was summed across the vertical axis. The resulting 1-D vector is normalized to a mean of 0 Fig 4.6. The number of zero-crossings were calculated as they indicate the number of ruler markings (×2). In binary image thresholding there are only two values. Pixel values are reduced to one of the two depending on which side of the theshold they are on. In mean thresholding, the threshold value is the mean of the pixel intensity. In adaptive Gaussian thresholding, the threshold value is a weighted sum of all values considered. The weights are defined as Gaussian functions. Otsu's thresholding[86] assumes each pixel either belongs to the foreground or the background class. It proceeds to find the optimum threshold to separate these two classes such that the intra-class variance is minimum and inter-class variance is maximum.

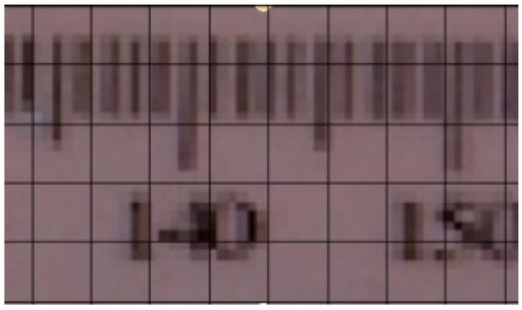
4.3 Results

Figure 4.7 shows the error in width and height using axis-aligned and rotatable bounding boxes. The comparison between axis-aligned and rotatable bounding boxes for localizing the ruler portion in the image shows that rotatable bounding boxes are better to determine the mean squared error in height and width. The next step was to determine the pixel length of ruler graduations in the image by dividing the cropped section into a grid boxes and binarizing all the boxes through applying adaptive Gaussian, Otsu and mean. Subsequently, the pixels were summed across the vertical axis, normalized the intensity values around a zero mean and counted the number of zero-crossings. The number of zero-crossings is a measure of the number of ruler graduations in that box (×2). Since, every box in the grid may give a different number of zero-crossings, the majority count from all boxes is considered. The adaptive Gaussian thresholding gave the least error in ruler length measurements (Figure 4.8). The adaptive Gaussian calculates the threshold for each image thus it can adapt to different ambient lighting conditions and brightnesses to give a better binarized image.

(a) Lobster image with a ruler in the background

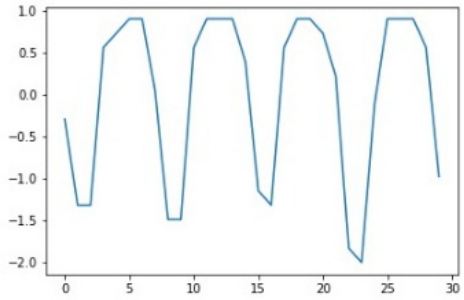
(b) The ruler portion is localized in the image.

(c) The detected ruler portion is cropped out of the image.



(d) The cropped ruler portion is overlaid with a grid.

(e) Each 100×100 pixel box is converted to a binary image.



(f) The binarized image is summed in the vertical direction. The resulting sequence shows a spike in value where the ruler graduations are present.

Figure 4.6: Steps in the methodology to determine the distance between ruler graduations. The cropped ruler section is overlaid with a grid system and every grid box is binarized and their intensity values are summed in the vertical direction. The resulting sequence yields peaks in the values for the case of the ruler.

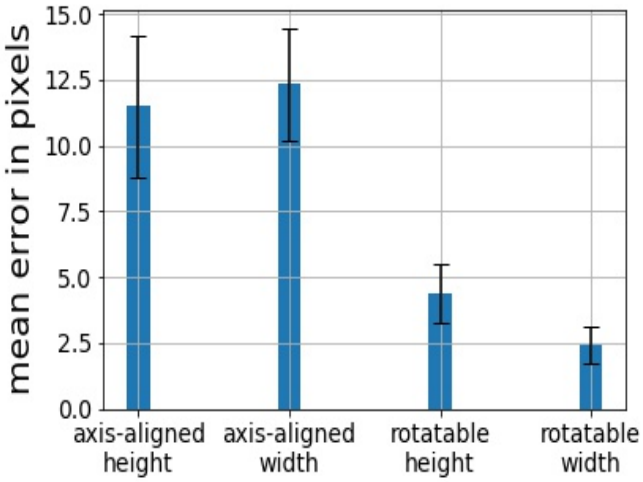


Figure 4.7: The rotatable bounding boxes have a lower mean squared error in height and width.

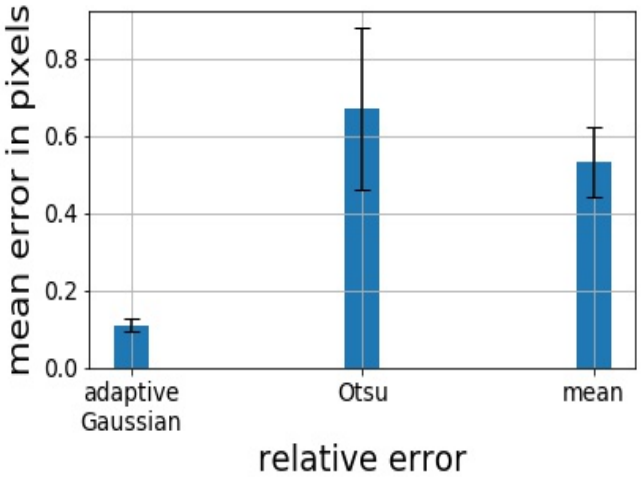


Figure 4.8: Error in ruler graduation spacings relative to height and width across different thresholding techniques. The adaptive Gaussian thresholding yields the lowest relative and absolure error compared to other thresholding techniques.

4.4 Concluding Remarks

So far, methods to detect bounding boxes around marine animals, have been described in Chapter 3. These have been used in subsequent chapters for cropping out portions containing lobsters from images. The scaling method described in this chapter can be

used to calculate the actual height and width against a standard measure of length for detected marine animals in images (with rulers). In the next chapter, the method to map morphologial landmarks on to lobters images is described. The scaling technique described in this chapter is applicable for scaling lobster morphological sizes in pixels to physical sizes.

Chapter 5

Visual Fingerprinting for Lobsters

Visual recognition of marine animal species, like lobsters, is used in research to monitor their growth and population demographics. This is usually achieved by physically tagging individual animals to identify and track them. However, tagging can sometimes cause physical harm or mobility restriction for the lobster, or other animal, with hard exoskeletons and thus interferes with the purpose for their tagging to begin with. Lobsters will periodically shed their exoskeletons which makes tracking them through tags difficult. This chapter aims to identify individual lobsters based on the patterns on their exoskeletons. Convolutional neural network architectures have been identified as an effective tool towards that [73] [76].

5.1 Background

MacDiarmid et al.[55] performed a study on identification of individual spiny lobsters. They observed that lobster body patterns are preserved at ecdysis (or moulting, when the exoskeleton is shed) which enables persistent identification of individual spiny lobsters throughout their lives. They marked 15 male and female Jasus edwardsii lobsters with color coded tags and obtained high quality images of them before and after moulting and observed that lobster body patterns are mostly preserved with only slight modifications. They also observed that such body patterns vary considerably among individual lobsters. This is true among other hard shell invertebrates as well. The research by MacDiarmid et al.[55] confirms that lobster carapace patterns are preserved through moulting. This means the lobster carapace patterns could act as a visual fingerprint for a lobster throughout its life.

Gosselin[23] et al. examined 332 female crabs and observed that each crab retained the pattern on its carapace after moulting. Three inexperienced observers were used to match the cast-off shell against photographs of crabs based on the pattern on their carapace. Their results indicate it is possible to identify individual lobsters based

upon the carapace pattern since both crabs and lobsters belong to the crustacean family.

This knowledge will be exploited to uniquely identify lobsters using vision-based techniques like CNN architectures towards lobster traceability.

Hillman et al.[29] developed a method to identify individial marine animals (e.g. dolphins) to compare new images with a group of previously identified images in their database. The matching algorithm was based on the pattern of nicks and notches on the dolphin's dorsal fin. They extracted the shape of the fin and used curve fitting methods to compare its shape against others in their database. They were able to achieve 75% accuracy. Their work indicates the possibility of uniquely identifying an animal within a species based on visual images - this concept is exploited in this thesis.

All this work suggests that there are characteristic markings and shapes that uniquely distinguish one member in a class of marine animals from another (Figure 5.1). Then, the next step is to find ways to automate this for lobsters. A similar problem that has achieved some automation and has had more attention than marine animal identification, is face recognition where the classes are individual people (where gender and age are varied).

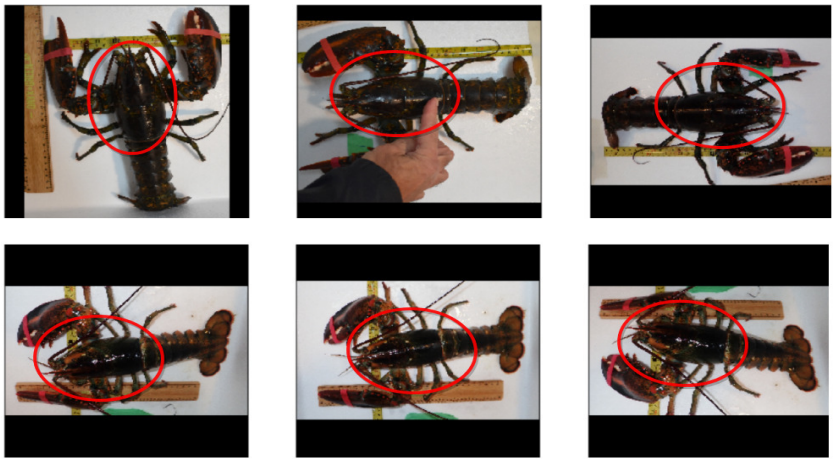


Figure 5.1: Variations in lobster carapace patterns (circled in red): the top row contains images of the same lobster from different views. Similarly, the bottom row contains images of the same lobster in different views. The top and bottom lobster can be visually discriminated by the pattern on the carapace.

Facial recognition can be challenging due to human behaviours like poses and

expressions as well as external conditions like ambient lighting which contributes to image variation in the same face. Early facial recognition methods such as Bayesian face[58] and LDA[3] attempted to minimize intra-personal face variations and maximize inter-personal ones. Guillaumin et. al.[24] used metric learning to map facial images to vector spaces such that images from the same person are 'closer' (in the Euclidean sense) than those that are from other people. However, deep learning approaches[73] [76] provided better tools to capture the complex nature of the intra- and inter- person variability due to its greater learning capacity.

The dataset used in the thesis has approximately five images per lobster. However, deep learning approaches require large datasets for the learning / training. Given a limited dataset, one-shot or few-shot classification techniques are relevant for the classifier to train on. Koch [42] et. al. used Siamese networks for one-shot image classification. A Siamese network consists of two convolutional sub-neural networks with shared weights. Pairs of images from the same class, and also from different classes, are created in equal proportion for a single batch of training. The image pairs are presented to the CNNs in the Siamese network. The structure of the CNN is a series of convolution and pooling layers. The feature representations from the layers are flattened to a one-dimensional vector. This flattened vector represents the projection of the image onto a continuous vector space. The distance between the two vectors is calculated using an Euclidean distance [14]. This distance is fed into a fully-connected layer and then finally, optimized using the cross-entropy loss function. Their results show the network correctly image differentiates pairs from the same class and those that are from different classes.

Taigman et al. [78] developed a nine-layer convolution neural network that trained on a dataset of four million images for 4000 distinct faces (or classes). They also developed a face alignment system based upon 3-dimensional models of faces. They achieved an accuracy rate of 97.35% on the LFW[34] dataset.

Sun et al.[73] developed their probabilistic classification model, DeepID2, to reduce intra-class and increase inter-class variations using face-identification and verification as supervision. The training inputs were pairs of images presented to a two-convolution neural network with shared weights. DeepID2 is optimized using two loss functions. The first addressed *identification*, which classifies the output of each

CNN to n different identities (people). The identification loss, captured through the cross-entropy (Eq. 5.1), requires knowledge of the identities' distributions (number of images / person), and is defined as:

$$CE = -1 * \sum_{i=1}^{n} c_i * log(\hat{c}_i) \tag{5.1}$$

such that c_i is zero for all i except the target class t (a particular person) which is 1 and $\hat{c}_i$ is that target's probability distribution. The second loss function, the face verification loss, discriminates between *feature representations* for the two images in a pair. The verification loss is a measure of *contrast* between images. This contrastive loss was defined [25] as :

$$Constrastive\ Loss = \begin{cases} \frac{1}{2}||f_i - f_j||_2^{\ 2} & \text{if } y_{ij} = 1 \\ \frac{1}{2} * max(0, m - ||f_i - f_j||_2^{\ 2}) & if y_{ij} = 0. \end{cases} \tag{5.2}$$

Here, f_i and f_j are representations of images i and j. y_{ij} is a Boolean variable which is 1 when both images are from the same class and 0 when they are from different classes. Eq. 5.2 requires the distance be larger than a margin, m.

Sun et al [74] developed DeepID3 by modifying two deep neural network architectures VGG[69] and Inception[77] for face recogntion. Similar to [73], they used the face identification (Eq. 5.1) and verification (Eq. 5.2) supervisory loss functions during training. Zhou et al. [92] trained a 10-layer CNN on a much larger data set of faces than DeepID [73] without joint verification and identification or 3-D alignment. They achieved 99.50 percent accuracy with the LFW[34] data set (state-of-the-art for the time) using the aforementioned data set which had five million labeled faces of 20,000 individuals. Schroff et al. [63] presented a CNN architecture, FaceNet, where triplet loss was used. The Siamese network was modified to take as input, a set of three images at a time. The triplet images consist of an anchor, positive and negative image. The anchor and positive images have the same identity whereas the negative one is different from both the anchor and positive images. This triplet loss aims to minimize the distance between the anchor and positive images and maximize the difference between the negative and anchor images. However, it was not straightforward to choose meaningful positive and negative pairs. Naive sampling can lead to choosing easy positive and negative pairs in the batch which causes a collapse in the

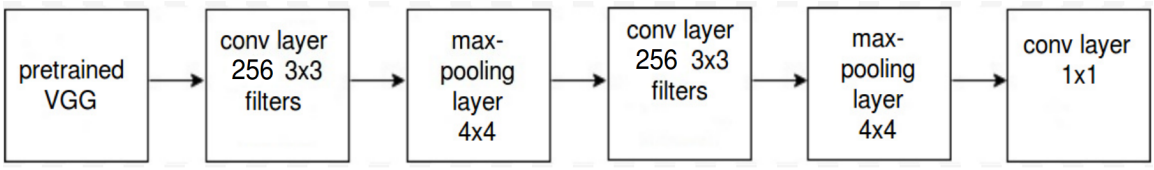


Figure 5.2: The pre-trained VGG-16 network is appended with two convolutional layers containing 256 filters and two max-pooling layers with 4×4 filters.

training. Many of the aforementioned CNN architectures like DeepID3 were trained with large data sets. The thesis' approach was a Siamese network-based architecture with pre-trained VGG (Fig 5.2) to address the the lobster recognition problem given the smaller datasets used.

5.2 Methodology

The thesis' approach to lobster recognition using deep learning networks is described in this section. The available dataset is described first followed by the model architecture.

5.2.1 Dataset

The dataset used contains 560 images of 114 lobsters which is approximately 5 images per lobster. Since the dataset is limited, image augmentation techniques as mentioned in Section 2.4 were used to increase the training dataset.

5.2.2 Model Architectures

Softmax-based classifier

The pre-trained VGG16 network was appended with two convolution layers with 256 3×3 filters and two max-pooling layers (Figure 5.2). The test accuracy was observed to increase notably until 256 filters and remained similar beyond that (Figure 5.3).

Additionally,the test accuracy was observed to increase notably until 2 convolution layers(keeping 256 filters) and remained similar beyond that(Figure5.4). The final layer is a dense one with units equal to the number of class identities and softmax activation. The softmax function changes an n-dimensional vector z of arbitrary real

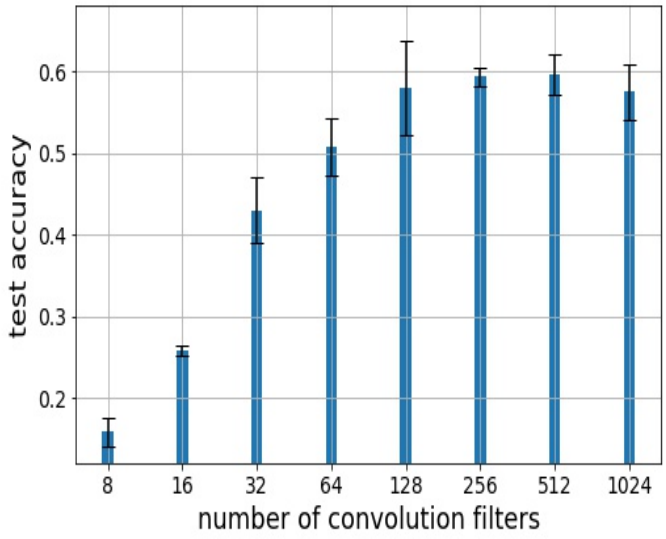


Figure 5.3: The test accuracy of the VGG-16 network was observed to increase until up to 256 filters and remained similar after that. This justifies using 256 convolution filters

values to an n-dimensional vector $\sigma(z)$ where $\sum_{i=1}^{K} \sigma_i = 1$ and $\sigma(z_j) = \frac{e^(z_j)}{\sum_{k=1}^{K} e^(z_k)}$. The Siamese network that this feeds into is described next.

Siamese Network

The pre-trained VGG16 network was appended with two convolution layers with 256 3×3 filters and two max-pooling layers Fig. 5.2. Subsequently, a dense layer containing 512 units was added. The weights used for the VGG16 are those from the training on the ImageNet dataset.

In each batch, pairs of images were created so that half of the pairs were from the same lobster (different camera aspect or type) and the other half were images from different lobsters altogether. Both images in the pair were input to the aforementioned CNN (Fig. 5.5). The CNN network outputs a 512-dimensional representation of each image. The Euclidean distance between the two vectors was calculated. The objective was to minimize the Euclidean distance if the image pairs were from the same class and to maximize the distance if they are from different classes.

The loss functions used to optimize the distance between image pairs were the binary cross-entropy (Eq. 3.1) and the contrastive loss functions (Eq. 5.2).

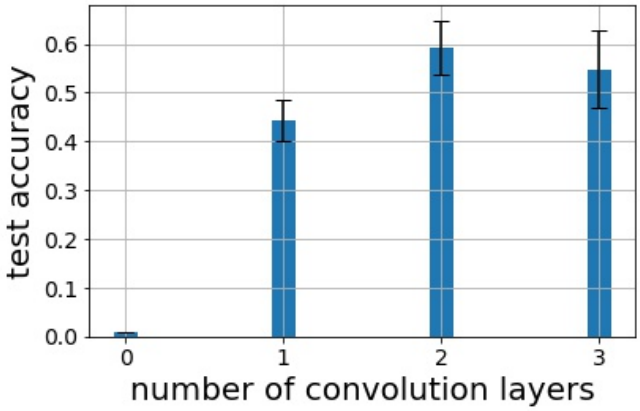


Figure 5.4: The test accuracy of the VGG-16 network was observed to increase until up to 2 convolution layers (keeping 256 filters) and was similar after that. This justifies using 2 convolution layers

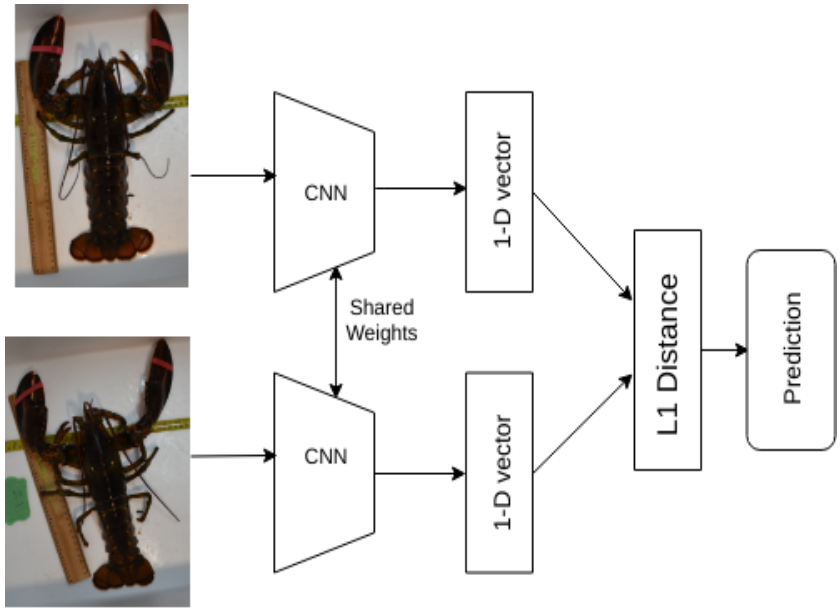


Figure 5.5: Siamese networks: pairs of images are input into the network. The network has two convolution neural sub-networks which are fed two separate images. The network learns to differentiate between pairs from the same class and those from different classes.

5.2.3 Inference

The test image was classified based on the training class which had the shortest Euclidean distance to it. The algorithm for test image classification is given in 1

Data:
M(x) = embedding for image x
i = test image
Result: Find the class id of the test image
support_set $\leftarrow$ *unique labels in the training set*
$em_i \leftarrow M(i)$
for *k in support_set* **do**
    $sum_k \leftarrow 0$
    $count_k \leftarrow 0$
    for *image j with label as k* **do**
        $em_j \leftarrow M(j)$
        $d_{ij} \leftarrow || em_i - em_j ||$
        $sum_k \leftarrow sum_k + d_{ij}$
        $count_k \leftarrow count_k + 1$
    end
    $average_k \leftarrow sum_k / count_k$
end
predicted_label $\leftarrow argmin(average_k)$

Algorithm 1: Pseudo-code for the algorithm to determine the closest label to the given test image

The models were trained with a patience number of 50, i.e if the validation loss (Eq. 5.1 and 5.2) does not decrease for a 50-epoch run, the training process is terminated. The models were trained with a patience factor of 50 since it was observed that the test accuracies did not improve much beyond that. The available data set was split so that the training was applied to 75% of the images and the test accuracy was reported on the other 25%.

Table 5.1 reports the top-k accuracy of softmax-based classifier and metric-based classification (i.e Eq. 5.2). The top-k accuracy is the percentage of instances where the true predicted labels lie in the top-k probabable predictions for that instance.

Table 5.1: Comparison of top-k accuracies using Siamese networks with contrastive loss for one-shot versus a softmax-based classifier. The contrastive loss based classifier had better top-3, top-5 and top-7 accuracies compared to the softmax and cross-entropy based classifiers as evident in the p-test values (< 0.05 threshold)

model	softmax-based classifier		contrastive loss		
	mean	std dev	mean	std dev	p-value
$k = 1$	0.56	0.06	0.57	0.025	0.97
$k = 3$	0.72	0.03	0.79	0.04	0.03
$k = 5$	0.81	0.03	0.87	0.02	0.002
$k = 7$	0.85	0.02	0.89	0.01	0.002
$k = 9$	0.88	0.03	0.90	0.01	0.22

Table 5.1 shows that the top-1 accuracy of cross-entropy and contrastive loss based methods were marginally better than the softmax-based classifier. The top-3, top-5 and top-7 accuracy of contrastive loss-based methods were better than the softmax and cross-entropy based classifiers. This means that with contrastive loss methods, it is possible to identify unique lobsters with at least equal or greater accuracy than softmax-based classifiers.

Figure 5.6 shows the variation in test accuracy with k-value for given margin parameters, m, of the contrastive loss function (Eq. 5.2). The top-k accuracies increase with increasing margin parameter to a certain point. Setting a very low value to the margin parameter desensitizes the loss function to the image pairs i.e the second part of Eq.5.2. However, setting a high value gives disproprotionate weight to images of different lobsters in a pair at the cost of pairs with images of the same lobster (favours inter- over intra- lobsters).

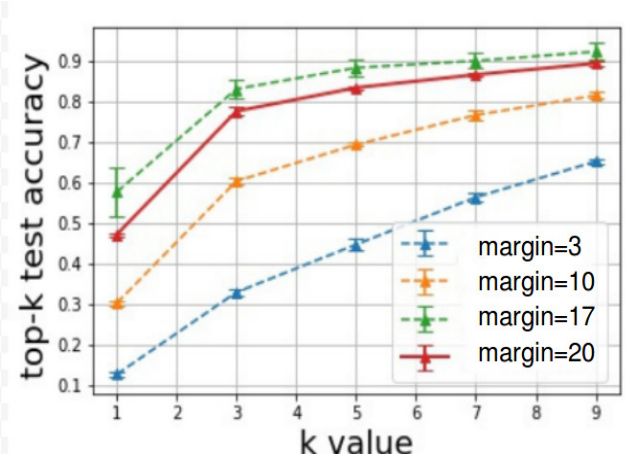


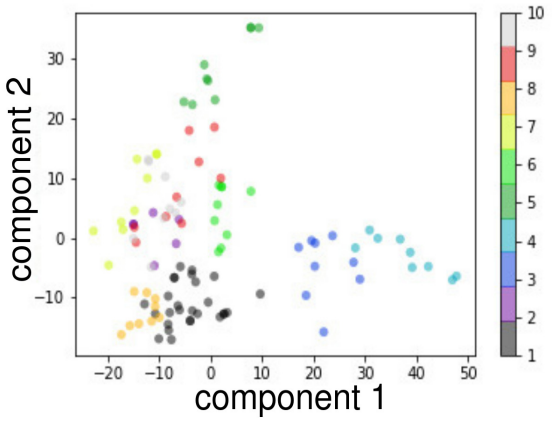
Figure 5.6: Variation in test accuracy with k-value for given margin hyperparameters (m in Eq. 5.2). Increasing the margin value increases the test accuracy to a point.

Figure 5.7 shows the Euclidean space of features after applying principal component analysis (PCA) on the feature space of the dataset embeddings learned by the classifiers (embeddings are a low-dimensional space into which high-dimensional vectors can be projected). Principal componant analysis reduces the dimensionality of a dataset that contains lots of features to a dataset that contains most of the information in the large data set. Siamese networks using constrastive loss show greater inter-cluster distance and reduced intra-cluster distance compared to softmax based features. This is the desired outcome.

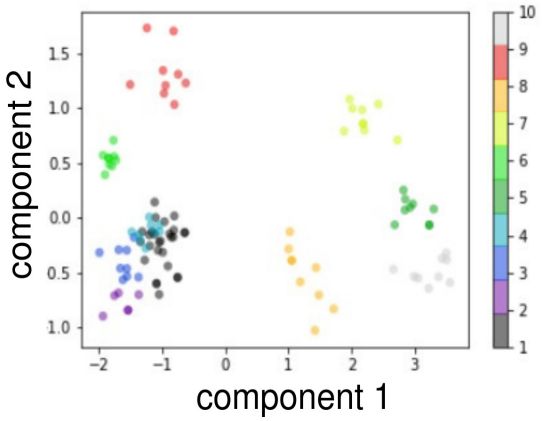
Figure 5.8 shows wrongly identified examples by the siamese network. The examples are for images where the lobster is partially occluded such as a hand as shown in figure 5.8a or extreme changes in lighting such as a camera flash as shown in figure 5.8b.

5.3 Concluding Remarks

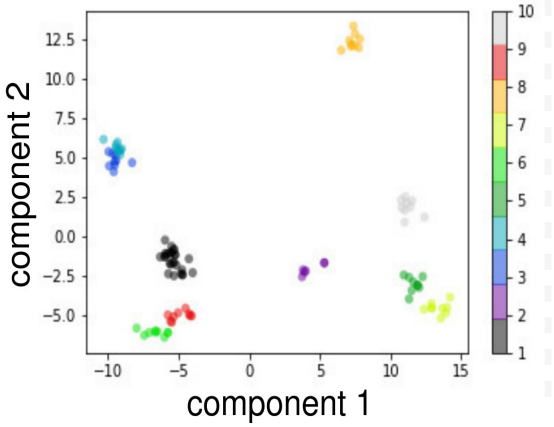
The chapter demonstrates the use of convolutional neural networks to identify individual lobsters. Horiguchi et al. [31] state that metric-based features perform well when there is less data, as in the case here. An example is one-shot or few shot classifications. This is further validated in the thesis results. Since there were only five images per lobsters (i.e. less data), the metric-based features i.e contrastive loss performed better than the softmax-based ones.



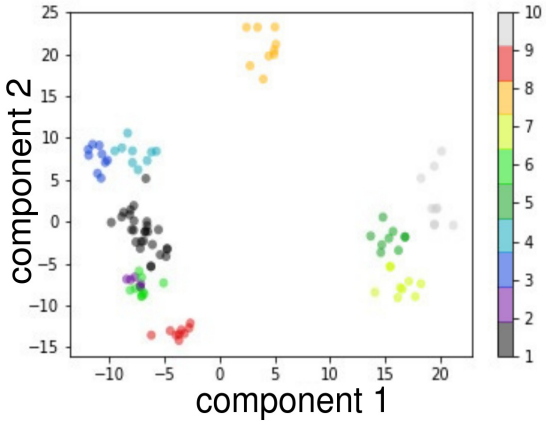
(a) PCA of the lobster Euclidean space from the softmax classifier. Inter-class differences < intra-class differences (not desirable).



(b) Contrastive loss for $m = 3$ (eq. 5.2). Inter-class differences < intra-class (not desirable).



(c) Contrastive loss for $m = 17$ (eq. 5.2). Note the desirably smaller intra-class, and larger inter-class, differences compared to (a) and (b), above. This is the best case.



(d) Contrastive loss for $m = 20$ (eq. 5.2) performs worse than (c), which has the desirable pronounced differences, but better than (b) and (a).

Figure 5.7: Visualization of Euclidean space using principal component analysis of the embedding space learned by using different models. Figures 5.7a and5.7b have low inter-class, and greater intra-class, differences (undesirable) compared with Figures 5.7c which has smaller intra-class, and larger inter-class, differences (desirable).

(a) The lobster is wrongly identified – possibly due to occlusions in the image such as a hand.

(b) The lobster is wrongly identified possibly due to extreme change in lighting such as a camera flash in in this case.

Figure 5.8: Examples of lobsters wrongly identified mainly due to the occlusions or extreme change in lighting such as camera flash

As well, Siamese networks were used to create an n-dimensional vector for every lobster. The method is essentially a black box technique to characterize lobsters. In the next chapter, lobsters are characterized quantitatively using various methods for mapping morphological landmarks on the lobster images. The landmark mapping enables the characterization of quantitative lobster features such as the length of lobster from the eyes to the end of the carapace, claw length and widths, abdomen lengths,etc.

Chapter 6

Lobster Landmark Localization

6.1 Introduction

Biologists use morphometry for quantitive analysis on the size and shape of organisms. Traditional morphometry consists of physical measurements like length, width, weight and area. Landmark-based morphometry uses a set of anatomical positions to analyse the shape [57] of a population. These anatomical positions are biologically meaningful [68] points that are consistent across similar species of a population. Morphometry is thus used to distinguish between species of similar shape. For example, Sontigun et al.[70] used wing morphometrics to identify 12 blow fly species of Thailand. Truss network measurements[72] are a set of distance measurements between landmarks on the surface of an organism. The truss network system uses geometric morphometry for stock identification. Stock identification can be used to cluster a population into groups with different growth rates. As mentioned earlier, there are ecological and economic sustainability reasons to assess and monitor stocks and their sizes.

This chapter reports on approaches to learn mapped landmarks on lobsters from training images and ultimately to extract them. The approaches consider deep learning architectures that include vanilla CNN, cascaded CNN, and CNN augmented with attention mechanisms. The next section outlines work related to landmark detection on human facial features and the use of attention mechanism to enhance the test accuracies of convolutional neural networks.

6.2 Background and Related Work

State-of-the-art research in deep learning using facial landmark detection algorithms could be applied to feature recognition on marine animals [65]. Facial landmark detection techniques map the coordinates of key facial features like eyes, nose and mouth from images and videos. Earlier work towards this includes Active Appearance

Models (AAM) [10] that create a statistical model using principal component analysis (PCA). PCA learns the orthogonal bases that capture the variations of the face. Then, the AAM finds the coefficients of the statistical model that best fits the test image.

Dollar et. al.[17] calculates object (landmark) pose in images with a cascade of random fern regressors [51]. Each regressor takes as input the image patch around landmarks predicted by the previous regression model. Xiong et al. [85] uses scale-invariant feature transform[53] of image patches as input to multiple cascade regressors to calculate facial landmarks at each stage. Sun et. al. [75] propose a three-level cascaded regression method with a CNN classifier at each level. Their first CNN provides a robust initial estimate of the facial landmarks with the following two CNNs refining the initial prediction towards higher accuracies. The work of Zhang et. al. [90] uses a cascade of coarse-to-fine autoencoders. The first autoencoder calculates preliminary landmarks from a low resolution version of the face. Then, subsequent autoencoders refine the landmarks by taking the local features extracted around the current landmarks. However, cascaded regression networks have shortcomings. For example, the learning process is independant of other stages. To address that, Trigeorgis et al.[80] proposed mnemonic descent which uses long short-term memory networks to model the dependancies between iterations in the cascade learning.

Attention mechanisms in deep learning are inspired by the human ability to focus on a subset of the environment [46]. The human visual attention mechanism can selectively focus on part of an image in high resolution while ignoring the rest. Similarly, an algorithm that maps landmarks on lobsters could focus on the lobster body in an image compared to its background. A CNN augmented with an attention mechanism could potentially map lobster landmarks more accurately by learning to focus on the lobster's body.

Yue et al. [88] propose a fully end-to-end convolutional regression network, as shown in Fig: 6.1, that yields facial landmarks in a single iteration. They introduce an attention mechanism where the network learns to "pay attention" to regions around landmarks *without* using image patches. The network generates spatial attention maps from the outputs of its multiple convolutional layers. The network explores image features at different scales with an attention mechanism that uses intermediate supervision to learn features that are relevant to facial landmarks. The total network

loss is the sum of the regression loss at each level of the network. However, this network needs to be trained from scratch. Therefore, this attention mechanism cannot augment pre-trained architectures like VGG or Resnet which are used in this thesis.

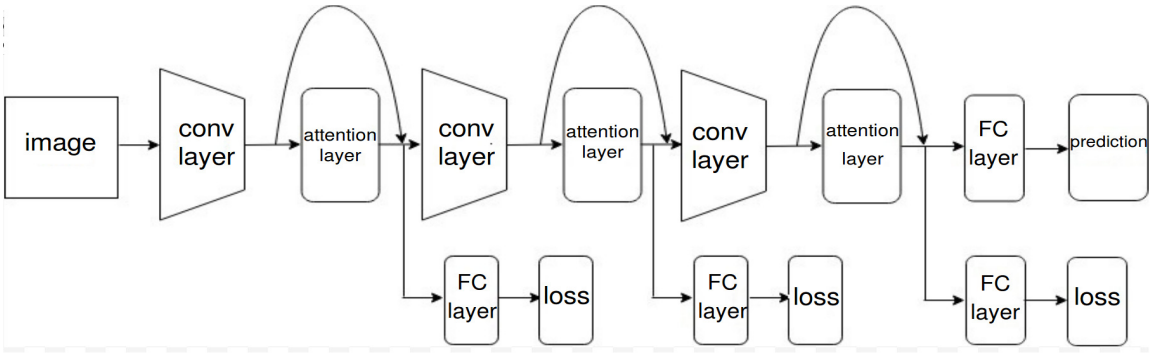


Figure 6.1: Alignment attention mechanism as mentioned in Yue et. al.[88]. The attention layers are placed in between the layers of the CNN modifying the original structure of the CNN.

With that inspiration, the next section outlines the experimental methodology for landmark localization of lobsters including dataset description, image augmentation techniques and details of the CNN model architectures considered.

6.3 Experimental Methodology

6.3.1 Dataset

This training dataset was 350 lobster images from National Lobster hatchery in the United Kingdom. [1]. Figure 6.2 shows a representative image from which 11 landmarks were mapped: two on the claws, one on the eyes, one at the end of the carapace, six between the segments of the abdomen and two on the tail. Bounding boxes span the lobster in an image. Since the dataset is limited, image augmentation techniques as mentioned in Section 2.4 were used to increase the training dataset.

6.3.2 Model Architectures

Vanilla CNN

For feature extraction, a pre-trained VGG16 CNN was used. The weights used for the VGG16 are those from the ImageNet dataset. The deeper layers of the pre-trained

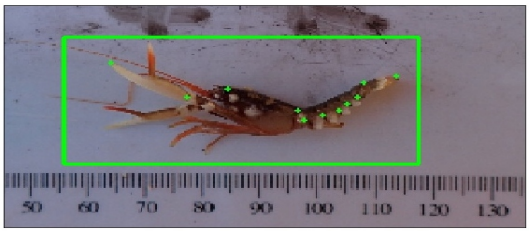


Figure 6.2: A representative training image that shows 11 landmarks on the lobster body. The manually drawn green bounding box shows the lobster extracted from the background. This was required for training the cascaded CNN. The image dataset was provided by National Lobster Hatchery in United Kingdom.[1]

networks were made trainable to fine-tune the accuracy [87]. The pre-trained VGG16 CNN is appended with a convolution layer with 128 3×3 filters followed by a 4×4 max pooling layer. This was followed by another convolution layer with 128 3×3 filters followed by a 4×4 max pooling layer (Figure 6.3). The mean squared error in pixels was observed to decrease with increase in convolution filters until up to 128 and remained similar beyond that (Figure 6.4). The mean squared error in pixels was observed to decrease with increase in convolution layers until up to 2 and remained similar beyond that. (Figure 6.5). The convolutional layer along with the pooling layers were added to down sample the output feature. The final layer is a 1×1 convolution layer. The number of output channels for the 1×1 convolution layer is twice the number of landmarks on the lobster (x and y coordinate needed for each landmark).

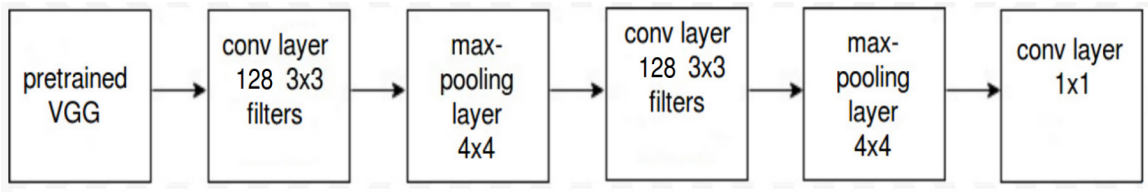


Figure 6.3: Landmark regression using pre-trained VGG16 architecture. The pre-trained VGG16 is appended with two convolutional layers containing 128 3×3 filters and two max pooling layers. The final layer is a 1x1 convolution layer with number of filters equal to twice the number of landmarks i.e 22 in this case.

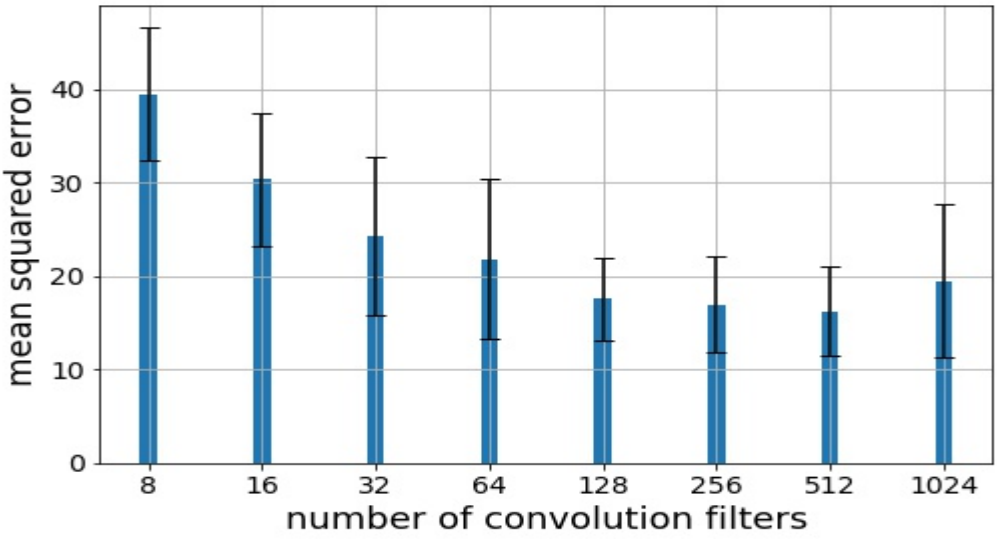


Figure 6.4: The mean square error, in pixels, decreases with increasing number of convolutional filters up to 128 and remained similar beyond that. This justifies using 128 convolution filters.

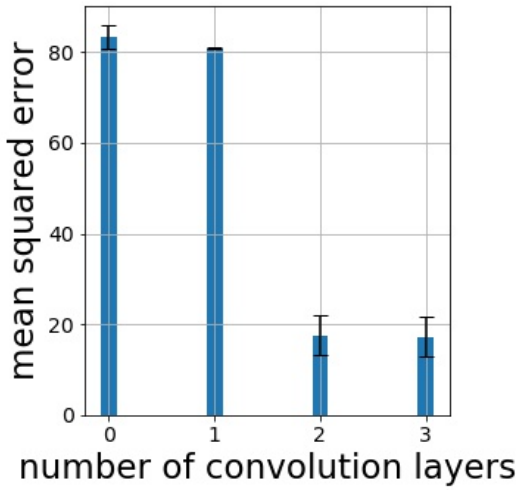


Figure 6.5: The mean square error (pixels) decreases with increasing number of convolutional layers up to 2 and remains similar beyond that. This justifies using 2 convolution layers.

Coarse-to-Fine CNN Cascade

Inspired by [75] and [56], a cascade of CNN regressors that progressively refines the output at each stage was created. First, an axis-aligned bounding box is detected

around the target (the box was manually placed there in an earlier step). The pre-trained VGG16 network (Figure 2.1) was initalized with ImageNet weights[15] and appended with additional convolution layers as shown in Figure 6.3. The last layer of the modified VGG network is a 1×1 convolution layer containing output channels equal to the four corners of an axis-aligned bounding box around the object as defined by the lower left and upper right coordinates. The input image is then cropped using the predicted bounding box coordinates of the network. Secondly, another modified VGG16 network, as shown in Figure 6.3, was used to detect an initial estimate of the landmark points on the cropped image. The final layer is a 1×1 convolution layer containing output channels equal to twice the number of landmarks on the lobster(x and y for each landmark). The cropped images were re-sized to 448×448 pixels with borders padded if necessary to preserve the original image aspect ratio. In the final step, separate modified VGG16 networks, as in Figure 6.3, were trained for each landmark point. The final layer of each network is a 1×1 convolution layer with two channels (x and y coordinate for each landmark). The input to each of these networks is a 90×90 pixel patch around the predicted landmark from the cropped image. This patch size was chosen since the maximum deviation of the true landmark and predicted landmarks was 90 pixels. All patches are re-sized to 224×224 again to preserve the aspect ratio. The reason for choosing a patch size of 224 was that a smaller size would lead to loss in image information and a large size would have unnecessarily more computational overhead. These networks are trained to refine the initial predictions with the input to the networks being small regions around the landmarks.

Attention-Based Landmark Detection

The modified VGG16, shown in Figure 6.3, is augmented with the attention mechanism described in Section 2.3 with two attention modules (not shown) as follows. The feature activations from ConvBlock3 and ConvBlock4 (Figure 2.1) are fed into two separate attention modules with attention width $K = 10$. The last layer of the modified VGG16 network is a 1×1 convolution layer with the number of output filters equal to twice the number of landmarks (x and y coordinate) which is 22 in this case (11 landmarks were regressed). The VGG16 network prediction is combined

with the predictions from the two attention modules.

6.3.3 Wing Loss

The mean squared error function (also called the $L2$ loss) penalizes large errors moreso than smaller ones (square of larger numbers is greater). The mean absolute error function (also called the $L1$ loss) is more sensitive to outliers since it does not square the errors but takes their absolute value. Feng et. al.[19] proposed a new loss function, the wing loss as defined in Eq. 6.1, which magnifies errors in the range of $(-\omega, \omega)$ compared to the mean squared and mean absolute error. The value of ω can be adjusted to define the range where loss needs to be amplified. The wing loss function switches from $L1$ loss to logarithmic loss when the error is in the range of $(-\omega, \omega)$ and hence puts more emphasis on errors in the range of $(-\omega, \omega)$.

$$wing(x) = \begin{cases} \omega \ln(1 + \frac{|x|}{\epsilon}) & \text{if } |x| < w, \\ |x| - C & otherwise \end{cases} \tag{6.1}$$

such that $C = \omega \ln(1 + \frac{|x|}{\epsilon})$. With this method, parameters of modified VGG16 network augmented with the attention mechanism were optimized as described in Rodriguez et. al. [67] using the wing loss [19] instead of the mean squared error.

The training and evaluation methodology was consistent for all the convolution models. Five fold cross validation [43] was used for model evaluation. At each iteration, four folds were used for training and the fifth for testing and evaluation. The data augmentation (Section 2.4) was performed on the training set but not on the test and evaluation set. In each iteration, the training was performed for 50 epochs.

In the next section, the results and conclusions from these experiments are described.

6.4 Results

The normalized mean squared error is summarized for the models tested in Table 6.1. The ADAM optimizer was used. The results in Table 6.1 and Figure6.6 indicate that the cascaded CNN approach and the attention augmented approaches have lower average normalized mean squared error compared to the vanilla VGG16

network. However, a t-test between the vanilla VGG and attention with wing loss yields a p-value of 0.059 (> 0.05 threshold). This means that there is no significant statistical difference between the two distributions. However the p-value is marginally close to the 0.05 threshold so the attention-based methods might have some effect on improving the vanilla VGG performance.

Table 6.1: Comparison of normalized mean squared error (NME) for CNN models considered for landmark regression performance across five folds.

CNN model	normalized mean squared error					mean	std
VGG16	18.44	16.93	33.77	12.73	16.61	19.70	7.28
cascade	12.50	11.11	11.75	15.96	11.88	12.64	1.71
attention	14.72	10.47	12.96	11.78	10.43	12.07	1.62
attention wing loss	14.32	10.30	12.66	10.93	9.10	11.45	1.83

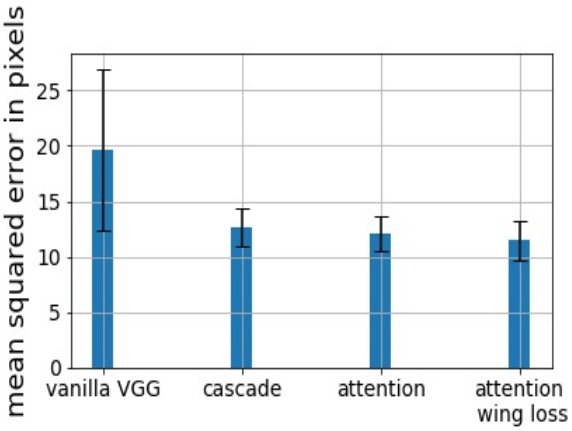
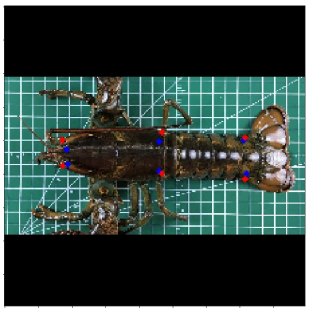
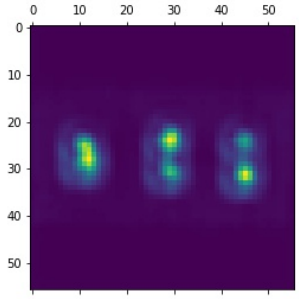


Figure 6.6: Root-mean-squared error for landmark localization on different model architectures. The cascade, attention and attention with wing loss do not appear to have a significant difference. A t-test between the vanilla VGG and attention with wing loss gives a p-value of 0.059 (> 0.05 threshold) suggesting no statistical difference between the two distributions.

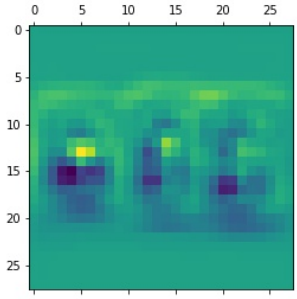
Figure 6.7 show the landmarks predictions for lobsters along with heat maps visually highlighting the attention heads' feature activations. The heat map temperature is greater around the landmark locations indicating that the attention maps are learning to focus on these specific lobster regions – as desired.



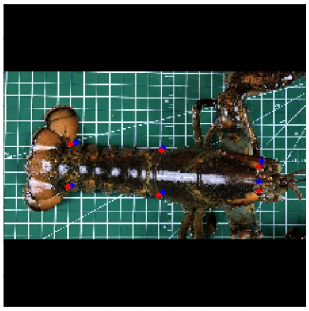
(a) Good e.g. 1: Ground truth and predicted landmark location comparisons.



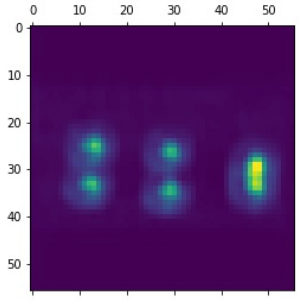
(b) 1st attention head heat map (good e.g. 1).



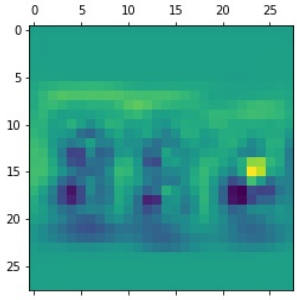
(c) 2nd attention head heat map (good e.g. 1).



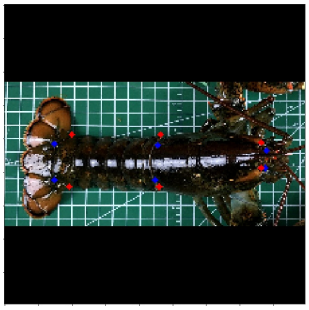
(d) Good e.g. 2: Ground truth and predicted landmark location comparisons.



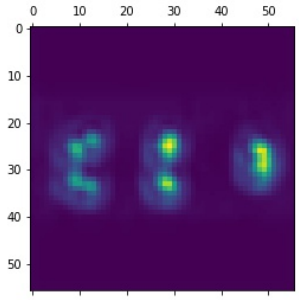
(e) 1st attention head heat map (good e.g. 2).



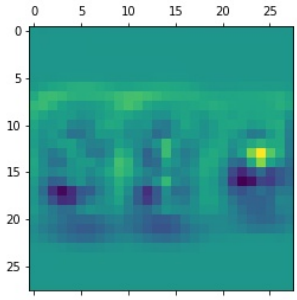
(f) 2nd attention head heat map (good e.g. 2).



(g) Good e.g. 3: Ground truth and predicted landmark location comparisons.

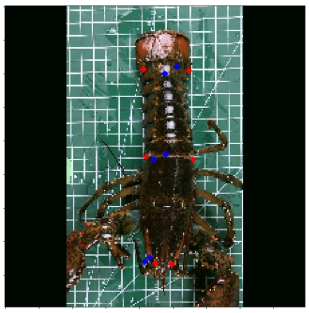


(h) 1st attention head heat map (good e.g. 3).

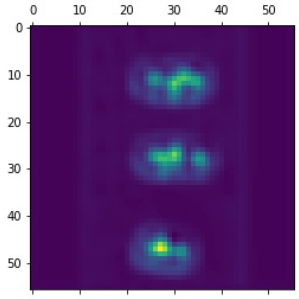


(i) 2nd attention head heat map (good e.g. 3).

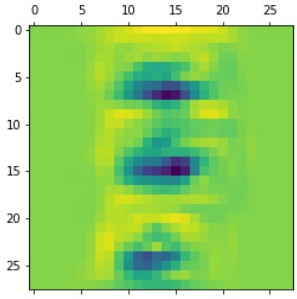
Figure 6.7: Visualization of the attention head weights highlight image areas the attention mechanism focussed on. The red dots represent the ground truth landmark locations and the blue dots, the predicted landmark locations. These 3 examples show the attention mechanism focusses, correctly, on the landmarks



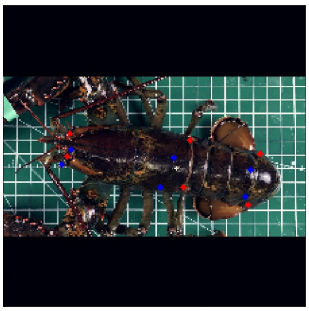
(a) Poor e.g. 1: Ground truth and predicted landmark location comparisons.



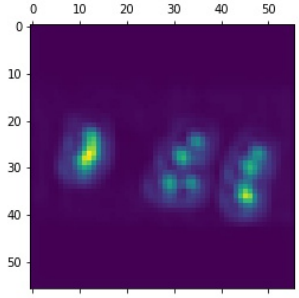
(b) 1st attention head heat map (poor e.g. 1)



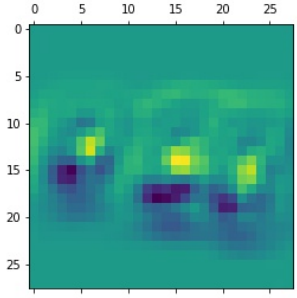
(c) 2nd attention head heat map (poor e.g. 1)



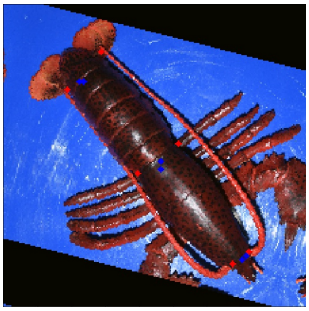
(d) Poor e.g. 2: Ground truth and predicted landmark location comparisons.



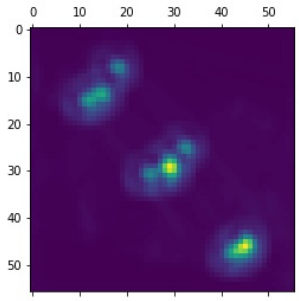
(e) 1st attention head heat map (poor e.g. 2).



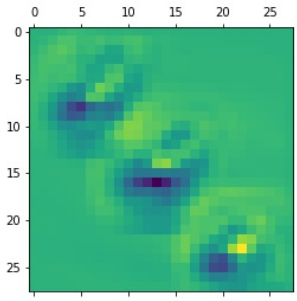
(f) 2nd attention head heat map (poor e.g. 2).



(g) Poor e.g. 3: Ground truth and predicted landmark location comparisons.

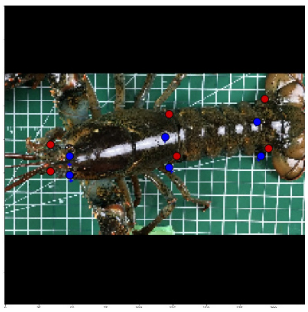


(h) 1st attention head heat map (poor e.g. 3).

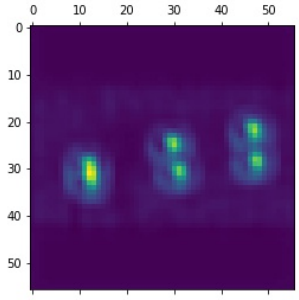


(i) 2nd attention head heat map (poor e.g. 3).

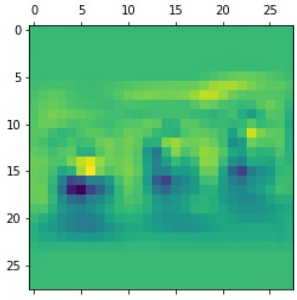
Figure 6.8: Visualization of the attention head weights highlight image areas the attention mechanism focussed on. Red dots represent the ground truth, and blue dots the predicted, landmark locations. These 3 examples show the attention model struggles to map the landmark positions – possibly due to the lobsters' pose.



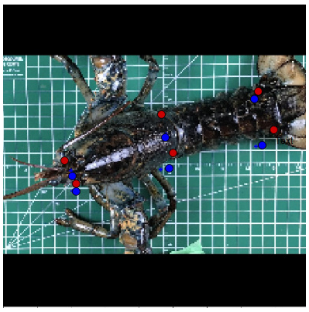
(a) Poor e.g. 4: Ground truth and predicted landmark location comparisons.



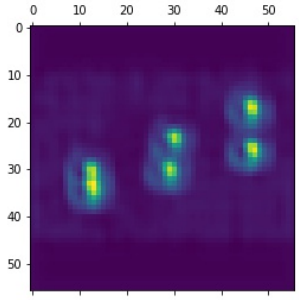
(b) 1st attention head heat map (poor e.g. 4).



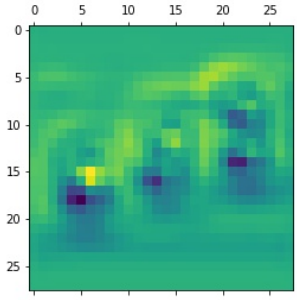
(c) 2nd attention head heat map (poor e.g. 4).



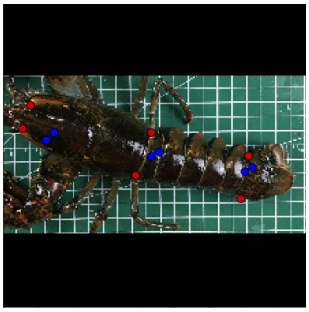
(d) Poor e.g. 5: Ground truth and predicted landmark location comparisons.



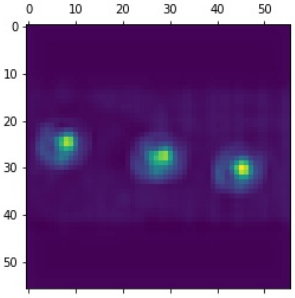
(e) 1st attention head heat map (poor e.g. 5).



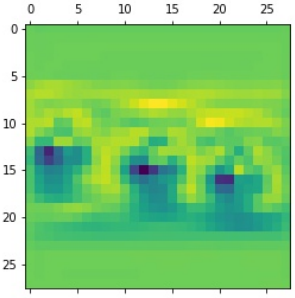
(f) 2nd attention head heat map (poor e.g. 5).



(g) Poor e.g. 6: Ground truth and predicted landmark location comparisons.



(h) 1st attention head heat map (poor e.g. 6).



(i) 2nd attention head heat map (poor e.g. 6).

Figure 6.9: Visualization of the attention head weights highlight image areas the attention mechanism focussed on. Red dots represent the ground truth, and blue dots the predicted, landmark locations. These 3 examples also show the attention model struggles to map the landmark positions – possibly due to the lobsters' pose.

Figures 6.8 and 6.9 show the landmark predictions for lobsters along with heat maps visually highlighting the attention heads' feature activations. However, these images show where the attention model struggles to map the exact landmark positions possibly due to the lobsters' unusual poses. An example of an unusual pose is when the lobster tail is tucked under the abdomen (Figure 6.8d). The landmark predictions are also slightly off when the lobster is placed at angle to the horizontal. (Figures 6.9a, 6.9g, and 6.8a). This suggests that the network performance is not rotational and pose invariant. The network performance can be improved by training with lobsters at different angles to the horizontal as well.

6.5 Concluding Remarks

The cascaded CNN approach uses a series of staged CNNs where each refines the landmark predictions from the previous one. The input image to a CNN is an image patch around the landmark predicted from the previous stage. The objective is to learn detailed features around the landmark points for more accurate predictions with subsequent stages. However, multiple cascaded regressors increase the computational and memory requirements. These staged regressors cannot be trained simultaneously since the input to subsequent CNNs depends upon the output from previous ones. CNNs with attention mechanisms[67] [88] learn to 'pay attention' to regions for fine-grained inference.

The modified VGG16 augmented with attention mechanism Section. 2.3 performs with similar accuracy to the cascaded CNNs. However, they predict landmarks in a single iteration and use a single model. A t-test between the two models confirms their similarity as the p-value is 0.33 (> 0.05 threshold). The attention mechanism has the advantage of a simpler implementation for comparable results.

The attention mechanism augmented CNN trained with wing loss makes the training process robust to outliers like occlusions since it emphasizes errors by giving more weight to smaller errors. However, the result with wing loss training shows similar accuracies to CNNs with attention. A t-test between the two confirms this with a p-value of 0.63 (> 0.05 threshold). This means the wing loss, while robust to outliers, is not producing any gains in accuracy.

In this chapter, CNNs were evaluated for mapping morphological landmarks onto

the various keypoints in the lobster image. The landmark mapping enables the characterization of quantitative lobster features like the length of from the eyes to the end of the carapace, claw length and widths, abdomen lengths, etc. In the next chapter, methods will be presented to address qualitative lobster attributes like gender, crusher claw location, etc.

Chapter 7

Fine-Grained Classification of Lobster Attributes

Properties such as gender, crusher claw location and abdominal somite bulge can characterize lobsters (individually or as a population). These properties are immutable i.e they do not change for a lobster in different environmental conditions. Mutable properties like tail spread and posture can be considered measures of the conditions the lobsters are kept in prior to entering the market.

Gender classification facilitates preservation of lobster populations. Female lobsters are considered a delicacy because their wider tail contains unfertilized eggs and have more meat. Currently, lobster processing plants manually classify lobster gender. The objective of this research and development is to automatically classify the lobster gender from an anterior (visual) view. Gender can be determined through the tail width – female lobsters have wider tails (Figure 7.1).

Visually, the lobster abdomen appears as a series of segments or somites. The amount of bulge in the first abdominal somite is an indication of whether the lobster has sustained injury. The visual classification of the first abdominal somite is important to assess the handling of lobsters in a processing plant.

Lobster stored in colder water are less active when removed from the water. Lobsters kept at higher temperatures display a more aggressive posture (claws extended upwards) when removed from the water. Lobster posture also gives some indication of the conditions the lobster were kept. Cooler temperatures are more desirable towards getting healthier lobsters to market.

Lobsters have two types of claws: pincher and crusher as shown in Figure 7.2. Their crusher claw is used to break open food with hard shells like crabs or clams. The crusher claw can be on either side of the lobster. The classification of which side is another attribute in their characterization or profile.

Lobster tail spread can be categorized as closed, partly spread or fully spread (Figure 7.3). Fully spread lobster tails are a sign of inactivity which is an indication

Figure 7.1: Gender as a classification class: (left) female; (right) male. The female lobster has a wider tail compared to a male lobster.

Figure 7.2: The crusher claw has a larger curvature and a white spot in between the claw teeth. The pincher claw is narrower than the crusher claw.

of poor lobster health.

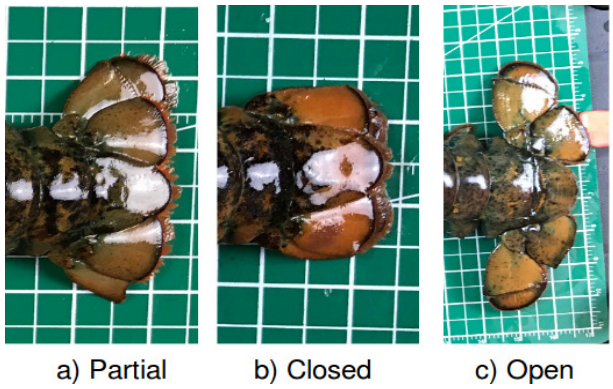


Figure 7.3: Tail spread as a classification class: (a) partially closed; (b) fully closed, and (c) fully open.

This chapter explores convolution neural networks for image classification of lobster based on five attributes: gender (male, female); detection and assessment of the first abdominal bulge (enlarged, or not); classification of crusher claw location (left, right); tail spread (partially closed, fully closed and fully open) and posture (non-aggressive, aggressive).

An attention mechanism to improve the classification accuracy of the vanilla VGG network is designed and used. Regular or vanilla CNN architectures do not deliberately extract detailed features from images. However, the attention mechanism learns to focus on regions of the lobster that differentiates their attributes . The attention mechanism proposed by Rodriguez et. al. [67] does not change the underlying information pathways of the CNN architecture. This means it can augment architectures like VGG and ResNet with no additional supervision unlike the attention mechanism proposed in [88]. Rodriguez et. al.'s attention mechanism can be easily inserted into an existing trained network to perform transfer learning (Section 2.2). This is the preferred approach for this thesis. Another relevant merit of the attention deficit method is that good results can be obtained from small training datasets – all that is available to this thesis.

The following section presents related work for fine-grained classification techniques that use convolution neural networks.

7.1 Background and Related Work

The gender of a lobster can also be classified by examining the first set of swimming legs on the lobster's underside. The female lobster has light almost feathery legs whereas the male lobster's are harder and bony. As mentioned earlier, the gender can also be classified from the tail width (Figure 7.1) since female lobsters have wider tails. There have been numerous studies and tools developed for classification of gender in human faces that are applicable to general lobster classification. These are briefly presented next.

Jaswante et al [40] performed gender classification on human facial images. They use the Viola-Jones algorithm [83] to extract distances between landmarks such as eye-to-mouth, between the eyes, nose width, and mouth width. These feature distances are input into a neural network which is trained to predict the gender from the image. Unlike the two step process of Jaswante et. al., CNNs predict the lobster gender in a single step.

Levi et al [49] implemented a convolutional neural network for automatic age and gender prediction on human faces. They tested their accuracy on the Audience dataset [18]. Levi et al. trained the CNN from scratch and did not use any pre-trained weights. For the thesis work there is only a small dataset which might be insufficient to train a CNN from scratch. Dhomne et al.[16] used VGGNet[69] for gender classification which could also be used well with limited training datasets. Since there is a small dataset for the thesis work, a similar method will used on a pre-trained VGG[69] to classify lobster properties such as gender, crusher claw location, abdominal bulge, etc.

The tail width of the lobster is a discriminating feature for gender classification. Fine-grained classification approaches are suited to find differences that visually separate male and female lobsters. Next, fine-grained classification methods that use convolution neural networks are compared to the attention augmented convolution neural networks [67].

Zheng et al. [91] classified fine-grained bird categories with a multi-attention convolution neural network (MA-CNN). MA-CNNs contain channel grouping networks which take input feature representations from a convolution neural network such as

VGG-16. It produces multiple parts like the bird's head or tail by clustering the spatially correlated input feature representations. The parts classifier network classifies the image using the individual part. The channel grouping network and parts classifier are optimized alternatively. However, unlike the attention augmented CNN [67], the approach does not learn to focus on features at different scales. The attention augmented CNN selects information at different scales by extracting features from different layers of the CNN. MA-CNN uses features only from the last layer. This could lead to the CNN performing poorer for different object sizes.

Han et. al. [26] used an attention module that uses attribute information like gender, hair, etc. to choose category features in different regions of facial images. Along with that, the network also has a category-guided attention module which selects local image features based on category labels. These combined features provide a richer feature set for fine-grained classification. However, the lobster dataset used does not have any such attribute information. An attention mechanism [67] that does not use any extra attribute information was used.

Lin et. al. [52] proposed a bilinear CNN architecture for fine-grained recognition that used two features extractors and multiplied each of their outputs, at each image location, to consider the pair-wise correlations between features. This enables learning to focus on image regions for fine-grained classification. However, the two CNN branches must be trained from scratch which is difficult with a small dataset. Apart from that, the architecture does not consider features at multiple scales.

Jaderberg et al. [36] introduced a new learnable module called the spatial transformer network that improved fine-grained classification performance by learning an affine transformation layer. This transformation layer learns to rotate or scale an image to enhance the geometric invariance of the CNN model and then performs the classification. The spatial transformer network, again, requires large training datasets which is difficult as the present work only has a small one.

Cui et al. [13] proposed a method of capturing domain similarity using the Earth movers distance (distance between two probability distributions in a region). This showed better transfer learning can be achieved by fine-tuning with a similar dataset of images. For example, if classification needs to be performed on a limited fish dataset, then a network pre-trained with a similar dataset, such as marine animals, might give

better accuracy than the smaller dataset alone. Cui et. al. pre-trained a CNN on a subset from the Imagenet dateset based on the proposed similarity measures then, used that to fine-tune a CNN for the target domain. There are no publicly available lobster datasets that could be used to pre-train a CNN so this method was not as applicable.

As informed by the related work, the next sections outline the experimental methodology and describes the learning models used in this thesis.

7.2 Experimental Methodology

The following subsections describe the datasets (which drives the experimentation), image augmentation and the learning models developed for training and prediction.

7.2.1 Training Datasets

The data collected for classification of lobster classes like gender, abdominal bulge, crusher claw location, tail spread, and posture. Each class has its own dataset. Often the classes in the datasets are unbalanced. The gender dataset, for example, has 75 % males and 25 % females. To combat such unbalanced classes, the penalized cross-entropy loss (2.6.5) was applied.

Generally, about 80 % of a dataset was earmarked for training and 20% for testing. The training set was increased by 2× more images through augmentation of the original images. Therefore, the training set is 3× the original training set. The distributions of the datasets will be presented in this section.

Gender

The training dataset for gender classification is based on multiple view of 114 individual lobsters which yielded a total of 159 images for this dataset. Approximately 73 % are male. Table 7.1 shows the ratio of male and female lobsters in this dataset. It shows the number of original images used for training was 127. With the augmented images, the total training set is 381 images. The 32 test images contain no augmented images.

Table 7.1: Dataset (159 images) class, training and testing distributions for gender classification

	class		training		testing
labels	male	female	orig set	orig + augmented	orig set
ratio	0.75	0.25	0.79	3×	0.21
number	120	39	127	381	32

Table 7.2: Dataset (228 images) class, training and testing distribution for abdominal bulge classification

	class		training		testing
labels	normal	abnormal	orig set	orig + augmented	orig set
ratio	0.67	0.33	0.79	3×	0.20
number	150	78	182	546	46

Abdominal Bulge

228 lobster images were side views which is useful to assess abdominal bulge. About 33% of these lobster images showed the first abdominal somite bulged out (Figure 7.4). Note, the tail of the bottom lobster is partially detached from the rest of the body – corroborating that this lobster has sustained injury. For contrast, Figure 7.5 shows a different view where the first abdominal somite is detached from the lobster. Table 7.2 shows the distribution of lobsters with and without abdominal somite bulge in the dataset. The majority of the dataset does not have the abdominal somite bulge.

Crusher Claw Location

A dataset containing 355 lobster images was used to classify the side that the crusher claw is located. Table 7.3 shows the percentage of lobsters that have the crusher claw on the right side which is 52 % and 48 % have it on the left side. Generally, the crusher claw has 50 % chance of being on either side of the lobster. This is another feature that is part of a lobster's profile and could help distinguish it from another lobster. As shown in Figures 7.2 and 7.6, the crusher claw is larger than the pincher claw and has higher curvature. As well, the crusher claw has white spots on the inner

Figure 7.4: Abdominal somite bulge is a classification class and health indicator: (bottom) an unhealthy lobster with a bulge on the first abdominal segment which is partially detached; (top) first abdominal segment is normal and the lobster is healthy.

Figure 7.5: Abdominal somite bulge is a classification class and health indicator: An unhealthy lobster with a bulge in the first abdominal somite. The abdomen is detached as the internal white portions are now visible.

Table 7.3: Dataset (355 images) class, training and testing distributions for crusher claw location classification

	class		training		testing
labels	left	right	orig set	orig + augmented train	orig set
ratio	0.46	0.54	0.79	3×	0.21
#	163	192	283	849	72

Figure 7.6: Crusher claw location is a classification class: (left) crusher claw on the right side and (right) crusher claw on the left side.

surface.

Tail Spread

Table 7.4 shows the class ratio of the lobster tail spread in the dataset. The dataset for tail spread classification contained 367 images of which: 125 are fully open; 57 are partially open and 185 are closed. Figure 7.3 shows examples of tails that are partially open, fully open and closed. The fully closed tail spread is an indication of activity and thus good health for a lobster.

Table 7.4: Dataset (367 images) class, training and testing distributions for tail classification

	class			training		testing
labels	open	partially open	closed	orig set	orig set + augmentation	orig set
ratio	0.34	0.15	0.50	0.79	2×	0.20
#	127	57	183	293	586	74

Table 7.5: Dataset (546 images) class, training and testing distributions for posture classification.

	class		training		testing
labels	aggressive	non-aggressive	orig set	orig set + augmentation	orig set
ratio	0.35	0.65	0.79	3×	0.21
#	79	149	182	546	46

Posture

The dataset for posture classification has 546 side view images (best view to assess posture) of which 183 have an aggressive posture. Table 7.5 shows the dataset distribution for posture classification. Fig. 7.7 shows examples where the top lobster has a non-aggressive pose and the bottom one has an aggressive pose. An aggressive pose is a sign of activity and therefore a lobster that is healthy.

7.2.2 Model Design

Vanilla CNN

For feature extraction, a pre-trained VGG16 CNN (Fig. 2.1) initialized with weights trained on ImageNet dataset was used. The deeper layers of the pre-trained networks were made trainable to fine-tune the accuracy.

As shown in Figure 7.8, the pre-trained VGG16 was appended with two fully-connected layers. The number of fully connected layers chosen was two since the test accuracy was marginally better for 2 hidden layers across most classifiers (Figure 7.9e). The final layer is a fully-connected layer with units equal to the number of classes to be distinguished along with softmax activation. For gender classification, the final fully-connected layer contains a single unit corresponding to whether it is a male or a female. For abdominal somite bulge classification, a single unit is used as well since it was necessary to classify whether the first abdomen bulges out. Similarly, to determine which side the crusher claw was on, a single unit was used. For tail classification, three units were used to classify whether the tail is fully spread, partially open or fully closed.

For gender classification, 64 neural network units were selected for the first two

Figure 7.7: Posture is a classification class: (top) a non-aggressive pose and (bottom) an aggressive posture with claws spread upwards.

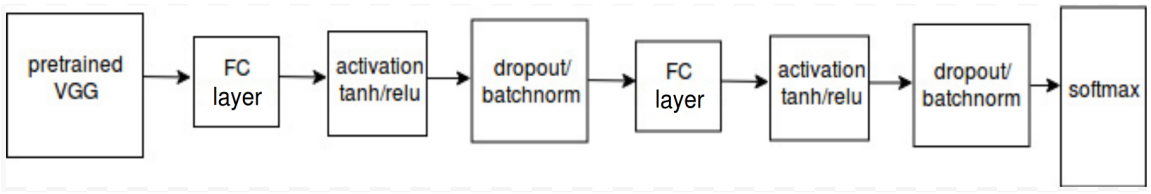
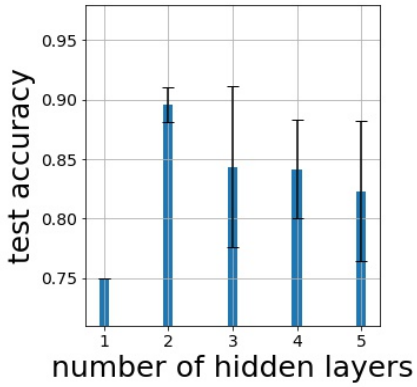
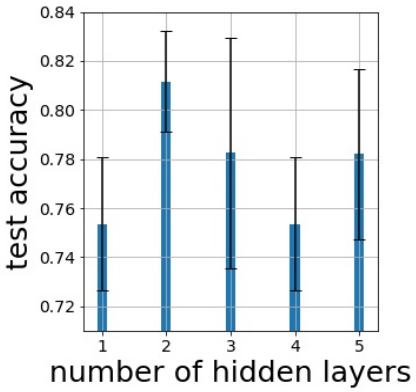


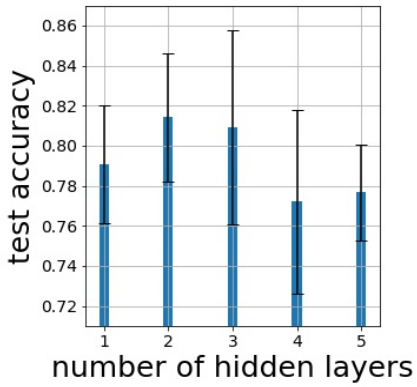
Figure 7.8: The modified VGG16 network developed in this thesis. Transfer learning was used to initialize the pre-trained VGG16 (Figure 2.1) with weights trained on the Imagenet dataset and appended with two fully-connected layers. Then, the network was re-trained with the custom dataset. The utility of *tanh* / *relu* activation and dropout / batch normalization layers were evaluated.



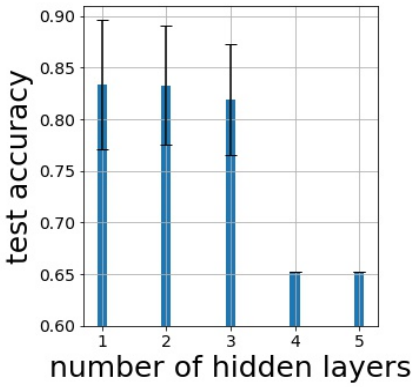
(a) The test accuracy for *gender classification* is maximum for 2 hidden layers.



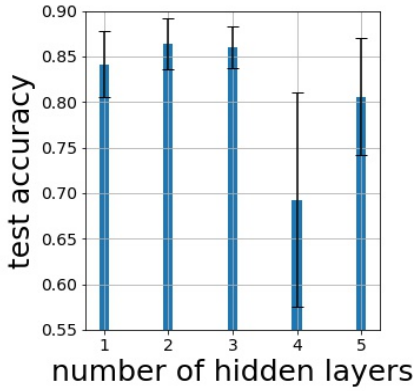
(b) The test accuracy for *abdominal bulge classification* is maximum for 2 hidden layers.



(c) The test accuracy for *crusher claw location classification* is similarly optimal for 1 - 3 hidden layers. Accuracy is marginally higher for 2 layers so this was chosen.



(d) The test accuracy for *posture classification* is similar for 1- 3 hidden layers after which it decreases. Accuracy is marginally higher for 2 layers so this was chosen.



(e) The test accuracy for *tail spread classification* is similar for 1 - 3 hidden layers. The accuracy is only marginally higher for 2 so this was chosen.

Figure 7.9: Test accuracies for lobster traits with number of hidden layers. Results were only marginally better, across all datasets, therefore, the choice was 2 across all classifiers.

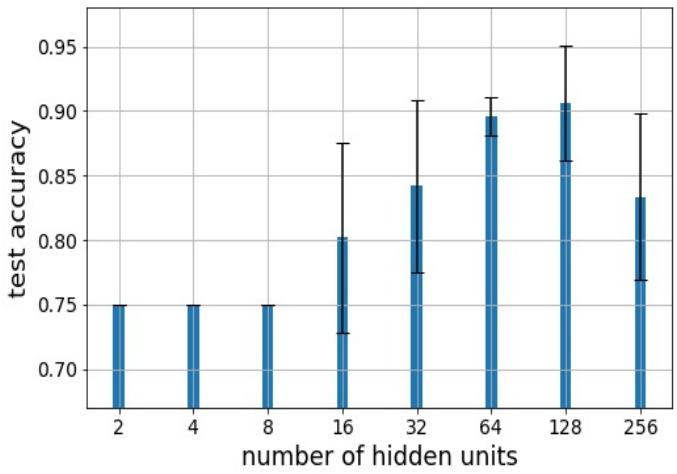


Figure 7.10: The test accuracy for *gender classification* increases with number of hidden units until it saturates around 128. Since the difference between 64 and 128 was marginal, 64 hidden units was selected.

fully-connected layers because the accuracy did not notably increase beyond that (Figure 7.10).

For abdominal somite bulge classification, 128 neural network units were selected for the first two fully-connected layers since the test accuracy did not increase notably beyond that (Figure 7.11).

For crusher claw location classification, 32 neural network units were selected for first two fully-connected layers since the test accuracy was maximum at 32 units (Figure 7.12).

For tail spread classification, 64 neural network units were selected for first two fully- connected layers since the test accuracy did not notably increase beyond that (Figure 7.13).

For aggressive posture classification, the test accuracies appear similar because of considerable overlap of the error bars as shown in Figure 7.14. However, there is a general increasing trend in test accuracy until 256 hidden units. Therefore, 256 neural network units were selected for the first two fully-connected layers because the test accuracy did not notably increase beyond that (Figure 7.14).

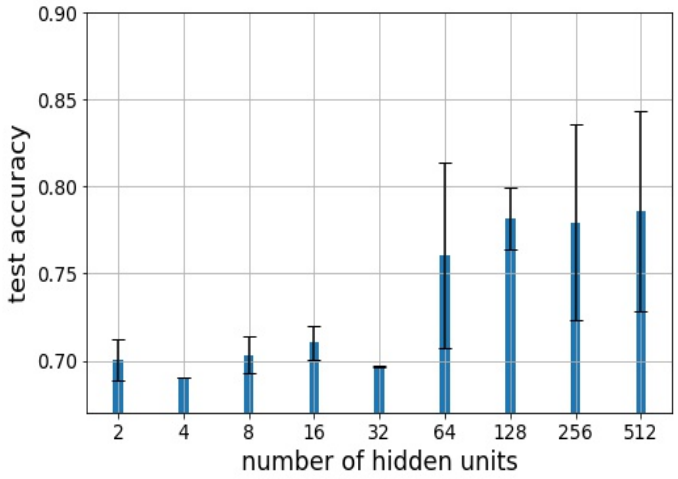


Figure 7.11: The test accuracy for *abdominal somite bulge classification* increases with the number of hidden units until it saturates around 128. Therefore, 128 hidden units was chosen.

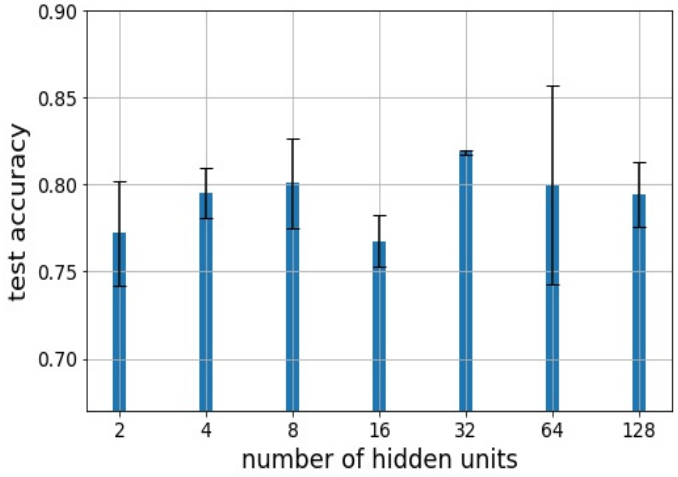


Figure 7.12: The test accuracy for *crusher claw location* increases with the number of hidden units until it saturates around 128. 32 hidden units was selected as it yielded the maximum value.

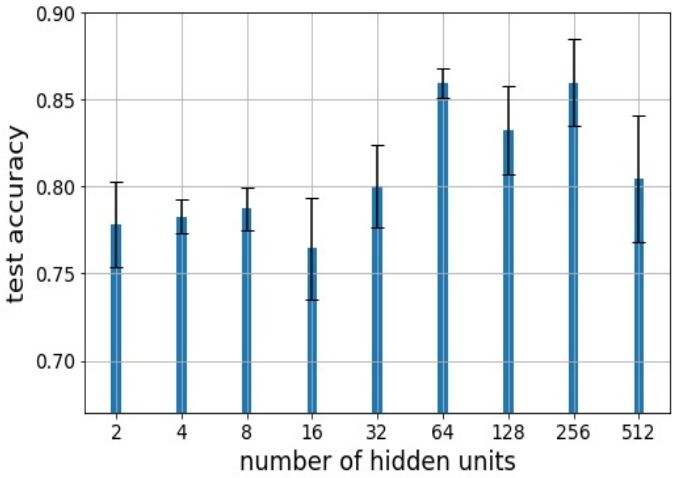


Figure 7.13: The test accuracy for *tail spread classification* increases with the number of hidden units until it saturates around 128. Since the difference between 64 and higher units were marginal, 64 was selected.

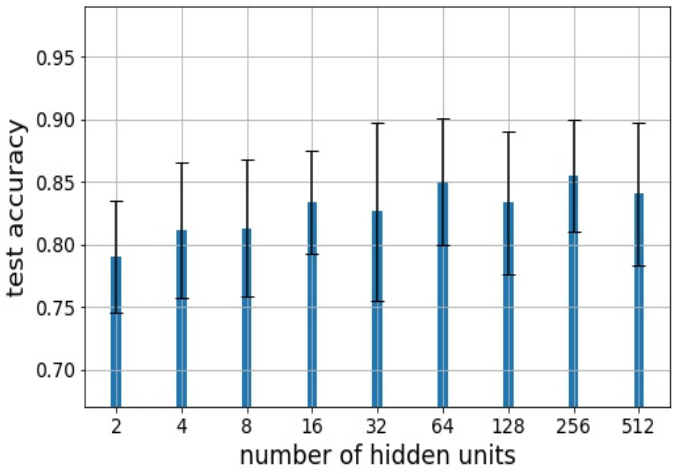


Figure 7.14: The test accuracy for *aggressive posture classification* increases with the number of hidden units until it saturates around 256.

CNN with Attention Modules

The attention mechanism for CNNs proposed by Rodriguez et. al.[67] for fine-grained classification was used in this thesis. The modified VGG16, as shown in Figure 7.8, is augmented with the above-mentioned attention mechanism with two attention modules. The feature activations from ConvBlock3 and ConvBlock4 (Figure 2.1) are fed into two separate attention modules with attention width K. K is a hyperparameter whose value is set before the model is trained. The last layer contains units equal to the number of classes to be classified.

The VGG16 network prediction is combined with the predictions from the two attention modules with attention gates, g (Eq. 2.11).

7.2.3 Experimental Methodology

The modified VGG16 (Figure 7.8) model was evaluated against the same modified VGG16 network augmented with attention modules to gage the impact of the attention modules. The pre-trained VGG16 (Figure 2.1) was appended with two fully-connected layers. The efficacy of two activation layers, *tanh* and *relu*, for the layer after both fully-connected layers were evaluated. After the activation layer, dropout or batch normalization layers can be added. The batch normalization and dropout layers can be used together. For the sake of simplicity either of these layers can be used. Dropout layers are for model regularization and batch normalization for faster convergence of deep neural networks. However, batch normalization can also act as a regularizer [35]. Two dropout rates, 0.4 and 0.2, were used. Figure 7.8 shows the combination of activation and dropout/batch normalization layers that were evaluated for VGG and VGG with attention modules.

Table 7.6 gives counts of the trainable parameters for each classifier in the Figure 7.8 architecture. The attention VGG has 1.2M trainable parameters more than the vanilla VGG. This is not significant in terms of computation cost compared to the total parameters in the vanilla VGG i.e. 15M including the trainable and non-trainable parameters.

The architectures were evaluated using both the ADAM (Section 2.5.2) and SGD optimizers (Section 2.5.1). The ADAM optimizer was used since it converges faster as explained in Section 2.5. However, experiments were also performed using SGD

Table 7.6: The number of classifier trainable parameters for the CNN networks considered – vanilla VGG has 1.6M and attention VGG, 2.8M. The attention modules have 1.2M more parameters which is not significant in terms of computation cost compared to the total parameters in the vanilla VGG i.e. 15M.

classifier model	vanilla VGG	VGG with attention
posture	1,610,177	2,814,728
crusher side	1,610,177	2,814,728
abdominal somite	1,610,177	2,814,728
gender	1,610,177	2,814,728
tail spread	1,610,307	2,967,430

since the ADAM optimizers do not always converge to the optimal solution.

As explained in Section 2.5, choosing a large learning rate can cause the loss value to diverge or oscillate over successive epochs and not converge to the global minimum. The training loss for vanilla VGG (Figure 6.3) was observed to diverge using learning rate 1e-2 for all classifiers and also for some classifiers with learning rate 1e-3. The vanilla VGG converged to the optimum for learning rate 1e-4. The attention VGG performed consistently across all three learning rates. Therefore, the architectures were evaluated across learning rates 1e-2, 1e-3 and 1e-4.

The binary classification models (except tail spread) were evaluated using accuracy as measured through the $f1$-score described earlier (Section 2.6.4).

With the experimental methodology and their performance metrics determined, the results from these experiment are presented next.

7.3 Results

The following subsections presents results for lobster classification for each of the attributes, i.e gender, abdominal bulge, crusher claw location, tail spread and posture. (1) The variation of the test accuracy with the number of attention heads (1, 3, 5, 7, and 9) is presented. Beyond 9 attention heads, the computational overhead becomes high with no discernible improvement in the mean test accuracy.
(2) Then, the test accuracies using the aforementioned vanilla and attention VGG with different VGG architectures and learning rates were evaluated. Three learning

rates, 1e-4, 1e-3 and 1e-2, were studied. The general trend across different classifiers is that the attention VGG exceeds the baseline performance across all architectures and learning rates. The test accuracy for vanilla VGG increases with decreasing learning rates i.e the vanilla VGG performs best with learning rate 1e-2 (the highest). SGD and ADAM optimizers were studied. As described in Section 2.5, SGD optimizers were additionally considered because the ADAM optimizer fails to converge under certain conditions. Having said that, similar test accuracies were observed for both optimizers.

(3) Then, the VGG (vanilla and with attention) performance (mean and standard deviation) was compared for training / testing accuracies and $f1$ scores against the baseline.

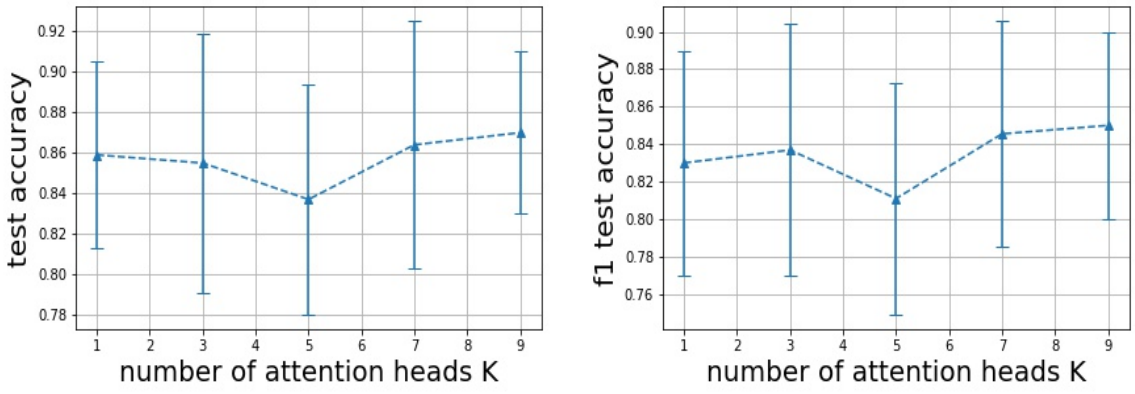
(4) The results were rolled up in confusion matrices to show the classification performance for true positives and true negatives for an attribute. A classification accuracy of 75 % was arbitrarily considered acceptable for the purposes of comparisons since it is midway between 50 % (random guessing) and 100 % (complete certainty).

(5) Finally, mis-classified examples for all attributes were presented and discussed in an attempt to determine the weaknesses in the two CNN models developed and studied.

7.3.1 Lobster Posture Classification

Figures 7.15a and 7.15b shows the variation of posture classification test accuracies and $f1$ scores with the number of attention heads (1 - 9). The mean accuracy does not increase beyond 9 attention heads (not shown). There is considerable overlap in test accuracies for the the five values indicating that the test accuracy does not depend strongly on the number of attention heads. A student t-test for $k = 14$ and $k = 5$ gives a p-value of 0.56 which fails to reject the null hypothesis. Hence, the test accuracies do not differ across $k = 1$ and $k = 5$.

Figure 7.16 shows the variation in test accuracy for posture classification across different architectures for the modified VGG16 neural network (Figure 7.8). The architecture is compared without (orange) and with (blue) attention modules. The baseline (green) is the accuracy obtained by predicting the most common label in the dataset. The trend in test accuracies of vanilla VGG is affected by the choice of



(a) Test accuracies are similar with different number of attention heads.

(b) $f1$ test accuracies are similar across different number of attention heads.

Figure 7.15: Lobster posture classification using: (7.15a) test accuracies and (7.15b) $f1$ scores. There is no strong dependence on number of attention heads for either. The attention modules are not making much difference.

architecture. The test accuracies generally follow an increasing trend with decreasing learning rates over the range from 1e-4,1e-3 and 1e-2. The attention augmented VGG follows similar accuracy trends across architecture and optimizers. The attention VGG outperforms the baseline for all architectures and outperforms the vanilla VGG at learning rate 1e-2 (the highest) across all architectures and optimizers. The vanilla VGG performs best with the lowest learning rate, 1e-4.

Table 7.7 shows training and test accuracies and $f1$ scores for vanilla VGG and VGG with attention modules for *posture classification*. Both models achieve comparable test accuracies of 85%. A student t-test (Section 2.6.1) performed on the test accuracies from five fold cross-validation across both models yields a p-score of 0.45 (> 0.05 threshold). This fails to reject the null hypothesis because the distributions are equal. Therefore, the accuracies from both models are comparable.

Fig 7.17 shows a confusion matrix for the lobster *posture classification*. The non-aggressive postures are classified with a mean 93% (3% standard deviation (std)) accuracy and aggressive postures are classified with mean 81% (6% std) accuracy using attention VGG. The non-aggressive postures are classified with mean 91% (2% std) accuracy and aggressive postures are classified with mean 77% (9% std) accuracy using vanilla VGG. The test accuracies across both classes i.e aggressive and non-aggressive posture, are better than the benchmark 75%.

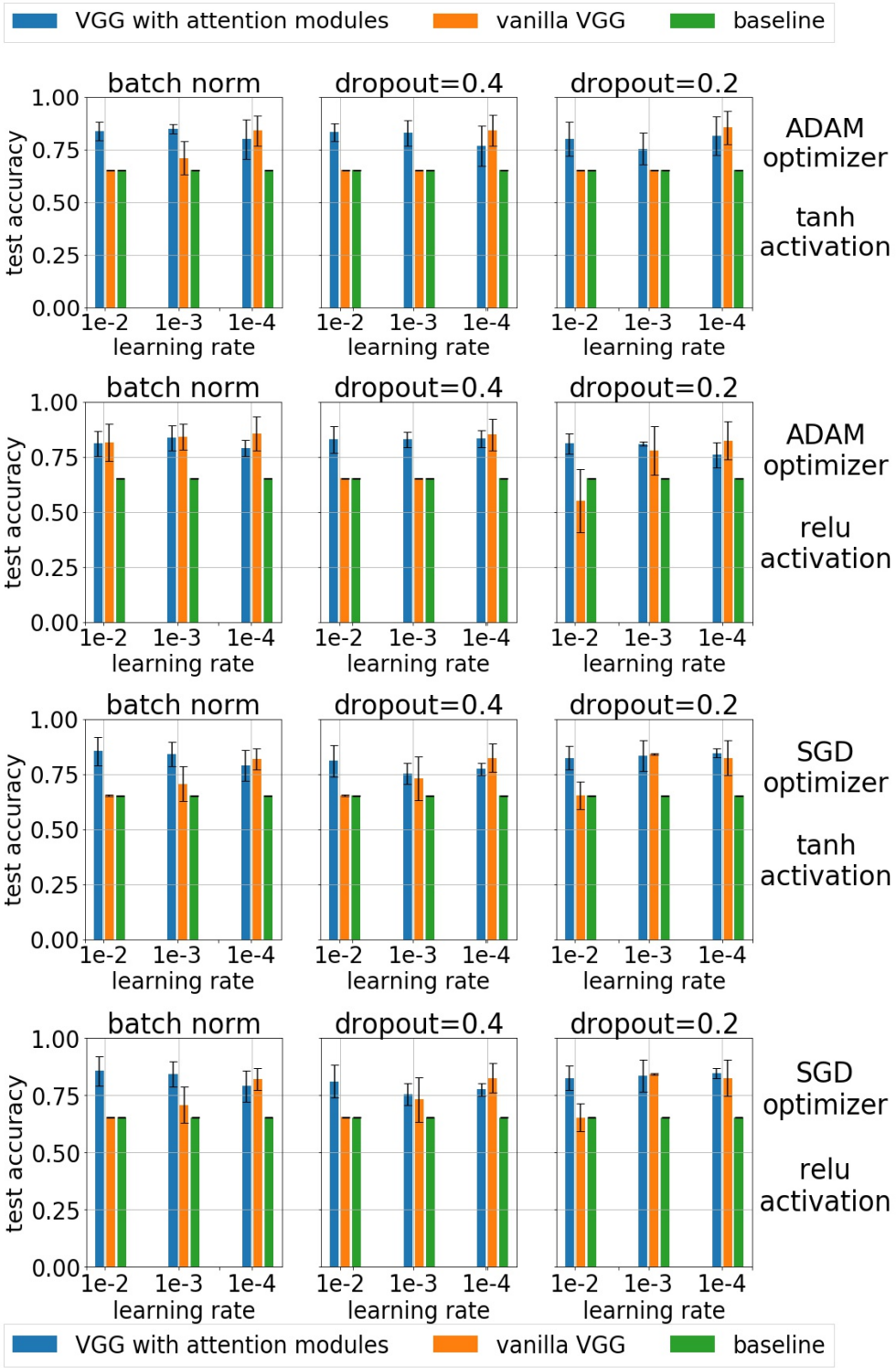
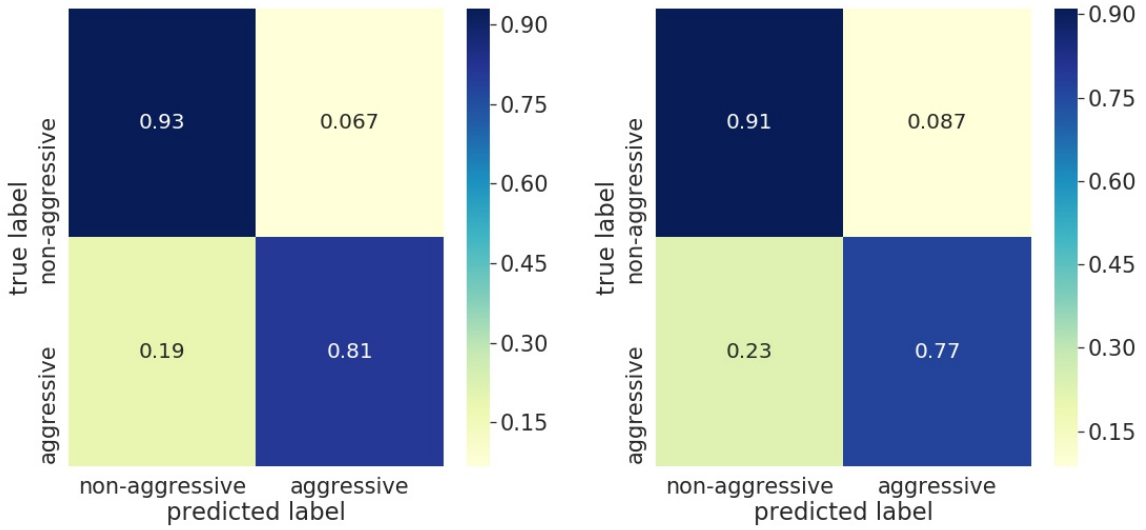


Figure 7.16: Posture classification accuracies across different optimizers and network architectures. The test accuracies generally increase with decreasing learning rates. The attention VGG outperforms the baseline for all architectures and learning rates.

Table 7.7: VGG (vanilla and with attention) performance shows similar test accuracies of 85% for posture classification. Attention modules do not make much difference. Both models though outperform the baseline model which predicts the most common label.

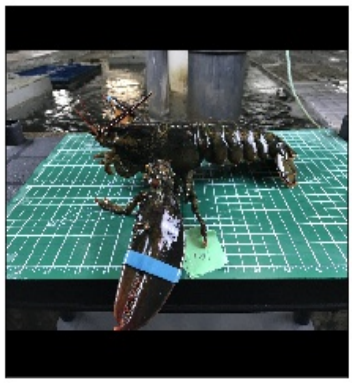
model	vanilla VGG		VGG with attention		baseline	
	mean	std dev	mean	std dev	mean	std dev
training accuracy	0.95	0.05	0.99	0.01	0.65	0.003
testing accuracy	0.85	0.04	0.86	0.06	0.65	0.003
training $f1$ score	0.94	0.05	0.99	0.01	0.35	0.001
testing $f1$ score	0.81	0.05	0.84	0.07	0.35	0.001



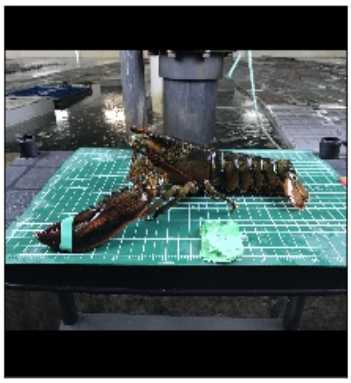
(a) Attention augmented VGG test accuracies across both classes are better than the acceptable benchmark of 75%.

(b) Vanilla VGG test accuracies across both classes are better than the acceptable benchmark of 75%.

Figure 7.17: *Posture classification* for both (7.17a) attention VGG and (7.17b) vanilla VGG have test accuracies across classes that exceed the acceptable benchmark of 75%.



(a) Example of aggressive posture mis-classification possibly due to the raised tail which is not as observable to the classifier.



(b) Example of aggressive posture mis-classification possibly due to the raised head which is not as observable to the classifier.

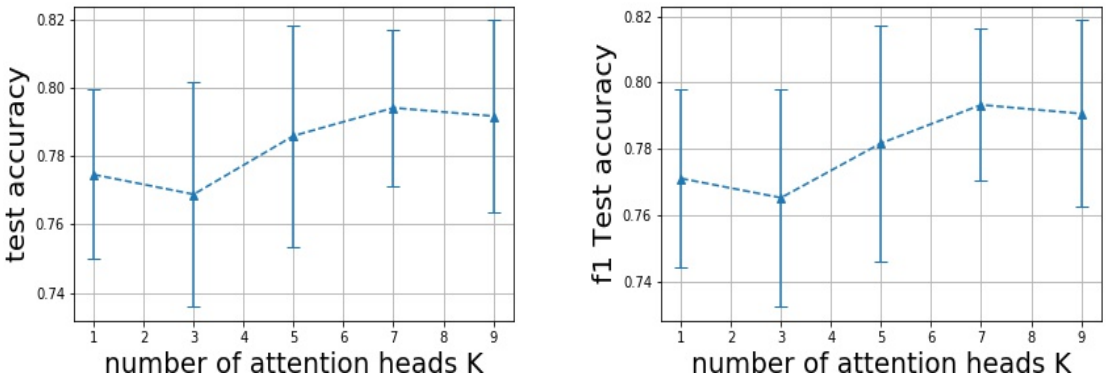
Figure 7.18: Lobster postures which are mis-classified due to their ambiguous postures.

Figure 7.18 shows examples that are incorrectly predicted by the classifier. The mis-classified examples have lobsters in an aggressive mode as expressed through an elevated tail or head which possibly leads to incorrect classification. For future consideration, the classifier can be improved by including more examples of the lobster in a partially aggressive mode (raised head or tail) in the training dataset.

7.3.2 Crusher Claw Location Classification

Figures 7.19a and 7.19b show test accuracies and $f1$ scores for crusher claw location classification as functions of the number of attention heads ($1 - 9$). There is considerable overlap in test accuracies for the the five values indicating that accuracy does not vary strongly with the number of attention heads. A student t-test performed for $k = 1$ and $k = 7$ yields a p-value of 0.28 which fails to reject the null hypothesis. Hence, the test accuracies do not differ across $k = 1$ and $k = 7$.

Figure 7.20 shows the variation in *crusher claw classification* test accuracies across different architectures for the modified VGG16 neural network (Figure 7.8). The architecture is compared without (orange) and with (blue) attention modules. The



(a) Test accuracies are similar across the number of attention heads K.

(b) The $f1$ scores are similar across the number of attention heads K.

Figure 7.19: Lobster crusher claw location classification (7.19a) test accuracies and (7.19b) $f-1$ scores are similar across the number of attention heads K. The attention modules are not making much difference.

baseline accuracy is in green. The baseline accuracy is from predicting the most common label. The trend in test accuracies of vanilla VGG is affected by the choice of architecture. The test accuracies of vanilla VGG generally follow an increasing trend with decreasing learning rates over 1e-2,1e-3 and 1e-4. However, the attention augmented VGG accuracies do not vary with architecture and optimizers. The attention VGG outperforms the baseline across architectures and classifiers and also outperforms vanilla VGG when using learning rate 1e-2 across architectures and optimizers. The vanilla VGG performs best with the lowest learning rate, 1e-4.

Table 7.8 VGG (vanilla and with attention) performance shows comparable test accuracies of 80% for *crusher claw location classification*. A student t-test performed on the test accuracies from five fold cross-validation across both models gives a p-score of 0.48 (> 0.05 threshold). This fails to reject the null hypothesis which means the distributions are equal. Therefore, the accuracies from both models are comparable.

Figure 7.21 shows the crusher claw classification confusion matrices. The attention VGG classifier correctly predicts the crusher claw side to the left with 77% (9% std) and to the right side with 82% (9% std) test accuracy. The test accuracies across both classes are better than the acceptable benchmark of 75%.

Figure 7.22 shows mis-classified crusher claw location examples by the classifier. The examples are for images where the crusher claw is partially visible or in a plane

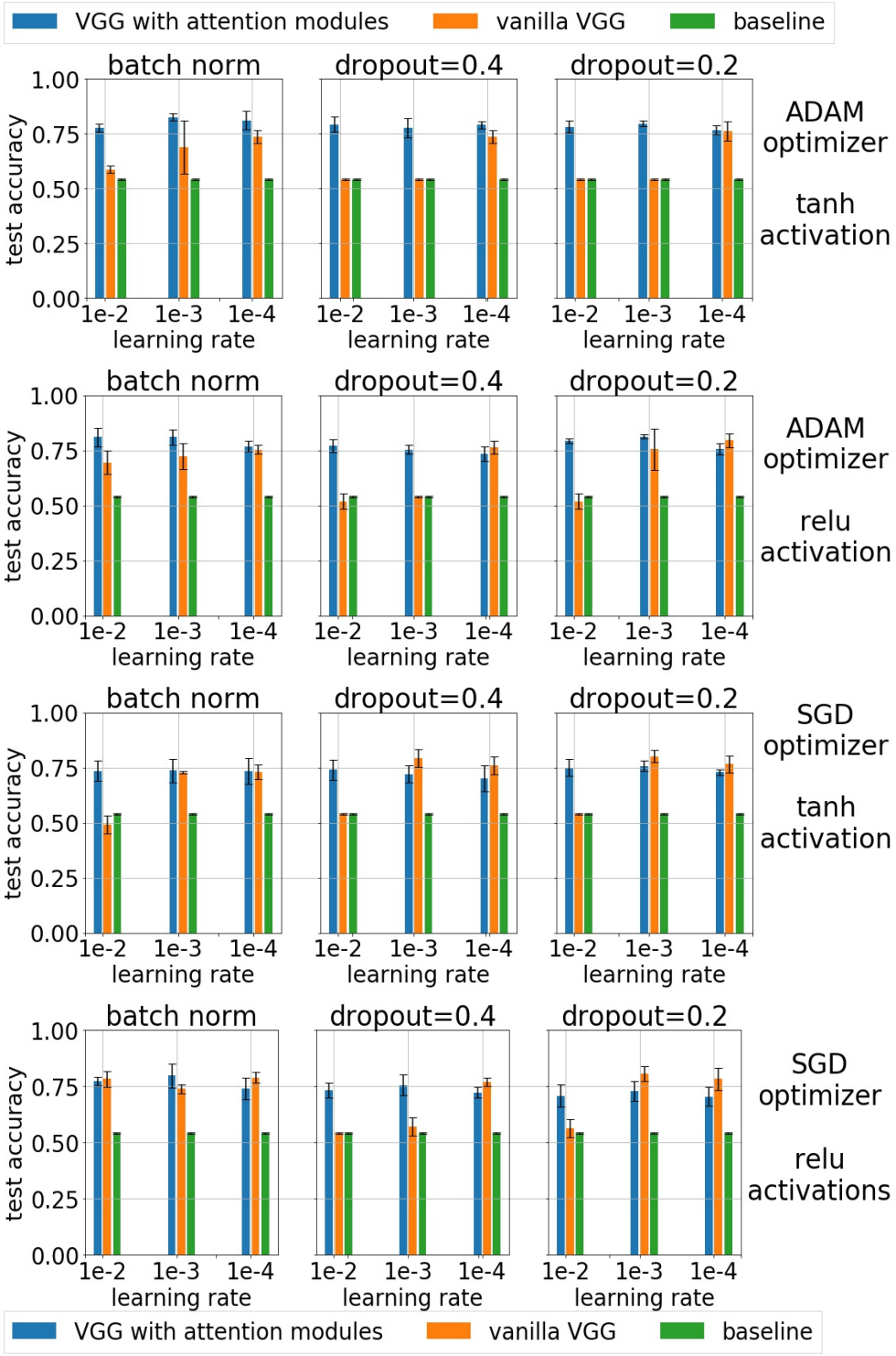
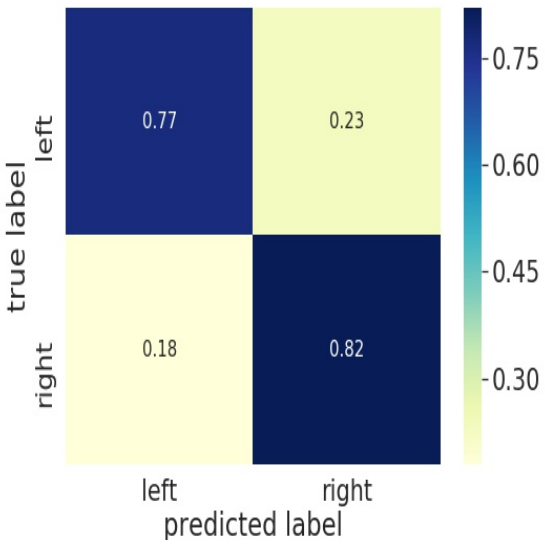


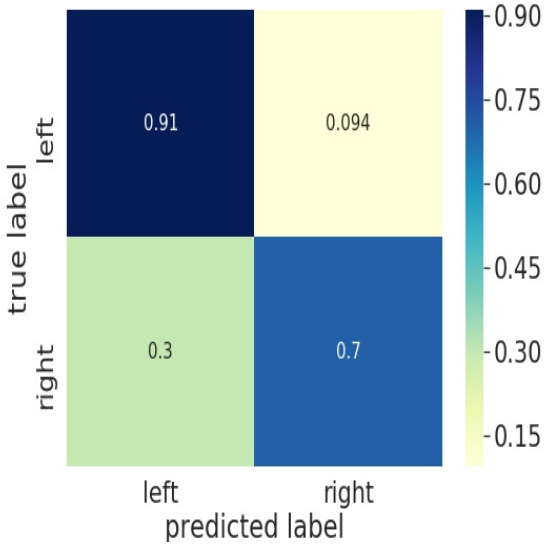
Figure 7.20: *Crusher claw location classification* accuracies using different optimizers and network architectures. The test accuracies generally increase with decreasing learning rates. The attention VGG exceeds the baseline across all architectures and optimizers.

Table 7.8: VGG (vanilla and with attention) performance shows similar test accuracies of 85% for *crusher claw location classification*. Both models outperform the baseline model which predicts the most common label.

model	vanilla VGG		VGG with attention		baseline	
	mean	std	mean	std	mean	std
train accuracy	0.84	0.01	0.95	0.03	0.54	0.002
test accuracy	0.81	0.04	0.79	0.02	0.54	0.002
train $f1$ score	0.83	0.02	0.95	0.02	0.35	0.001
test $f1$ score	0.80	0.04	0.79	0.02	0.35	0.001



(a) Attention augmented VGG test accuracies across both classes are better than the acceptable 75% benchmark.



(b) Vanilla VGG test accuracies across both classes are better than the acceptable 75% benchmark only for the left claw.

Figure 7.21: *Crusher claw location classification* test accuracies for: (7.21a) attention VGG exceed the 75% bench while for the (7.21b) vanilla VGG the classification on the right side is less than the benchmark 75% classification and well above that on the left side. The reasons why are unclear.

that is at an angle to the horizontal plane. A good (with the lobster flat in the horizontal plane) view of the crusher claw shows a white spot on the inside surface. For future consideration, the classifier can be improved by including another label for examples of crusher claws that are partially visible, thereby putting these lobsters into a separate class.

7.3.3 Gender Classification

Figures 7.23a and 7.23b shows gender classification test accuracies and $f1$ scores with the number of attention heads (1 – 9). There is considerable overlap in test accuracies for the the five values indicating that accuracy does not vary strongly with the number of attention heads. A student t-test for $k = 1$ and $k = 3$ yields a p-value of 0.06 which fails to reject the null hypothesis. Hence the test accuracies does not differ across $k = 1$ and $k = 3$.

Figure 7.24 shows the gender classification test accuracies across different architectures for a modified VGG16 neural network. The architecture is compared without (orange) and with (blue) attention modules. The baseline is depicted in green. The baseline accuracy is from predicting the most common label.The test accuracies of vanilla VGG generally follow an increasing trend with decreasing learning rates over 1e-2,1e-3 and 1e-4. The attention VGG16 outperforms the vanilla VGG16 for all optimizer and network architectures considered.

Table 7.9 shows gender classification train / test accuracies and $f1$ score for vanilla VGG and VGG with attention modules. Both models achieve comparable test accuracies of 90%. A student t-test performed on the test accuracies from five-fold cross validation across both models yields a p-score of 1.0 (> 0.05 threshold). This fails to reject the null hypothesis which implies that the distributions are equal. Hence the accuracies from both models are comparable.

Fig 7.25 shows the gender classification confusion matrix. The attention VGG classifies the female with an accuracy of 62% (2% standard deviation) and the vanilla VGG classifies with an accuracy of 79% (6% standard deviation). The test accuracies across classes for male prediction is better than the acceptable benchmark of 75%. However, the test accuracy for female prediction using attention VGG is less than the acceptable benchmark of 75%. The female lobster is hard to classify possibly due to

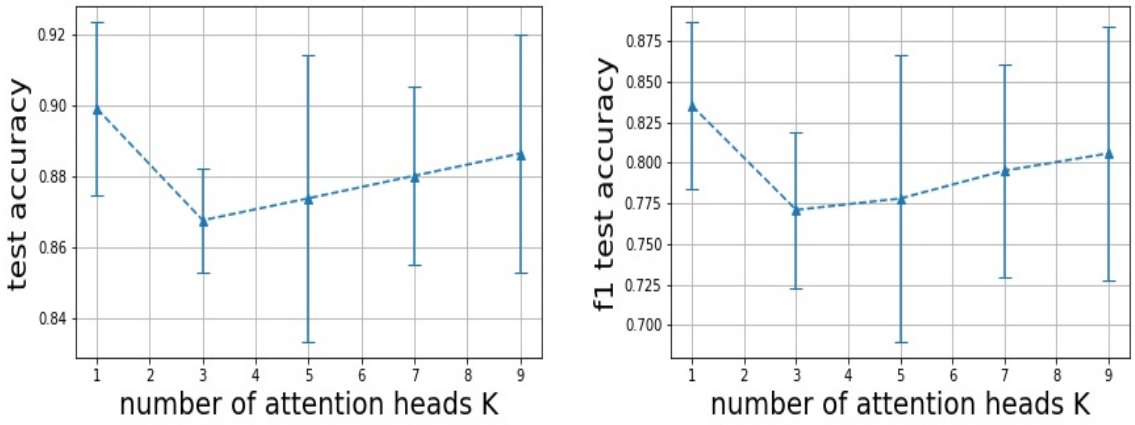
(a) The crusher claw is mis-classified as on the left side – possibly due to the pincher claw being only partially visible.

(b) Crusher claw is mis-classified as on the right side. The crusher claw is twisted out of the horizontal plane and thus appears smaller, and thus mistaken for the pincher claw, in a plan view classification.

(c) Crusher claw mis-classified as on the right side. Same reason as in (b).

(d) Crusher claws that are mis-classified possibly due to an tilted view of the left claw not providing the best perspective.

Figure 7.22: Examples of mis-classified crusher claw location. Mainly due to the crusher claw being twister out of the horizontal plane and possibly mistaken for the pincher claw.



(a) Test accuracies are similar across the number of attention heads K.

(b) $f1$ scores are similar across the number of attention heads K.

Figure 7.23: *Gender classification*: (Fig 7.23a) test accuracies and $f1$ scores (Fig 7.23b) are similar across the number of attention heads. The attention modules are not making much difference.

Table 7.9: VGG (vanilla and with attention) performance shows similar test accuracies of 85% performance for *gender classification*. The student t-test between both classifiers gives a p-value of 1.0 (> 0.05 threshold) indicating both models have similar accuracies.

model	vanilla VGG		VGG with attention		baseline	
	mean	std dev	mean	std dev	mean	std dev
train accuracy	0.99	0.01	0.98	0.02	0.75	0.009
test accuracy	0.91	0.05	0.90	0.02	0.75	0.009
train f1 score	0.99	0.01	0.97	0.03	0.35	0.001
test f1 score	0.86	0.07	0.84	0.05	0.35	0.001

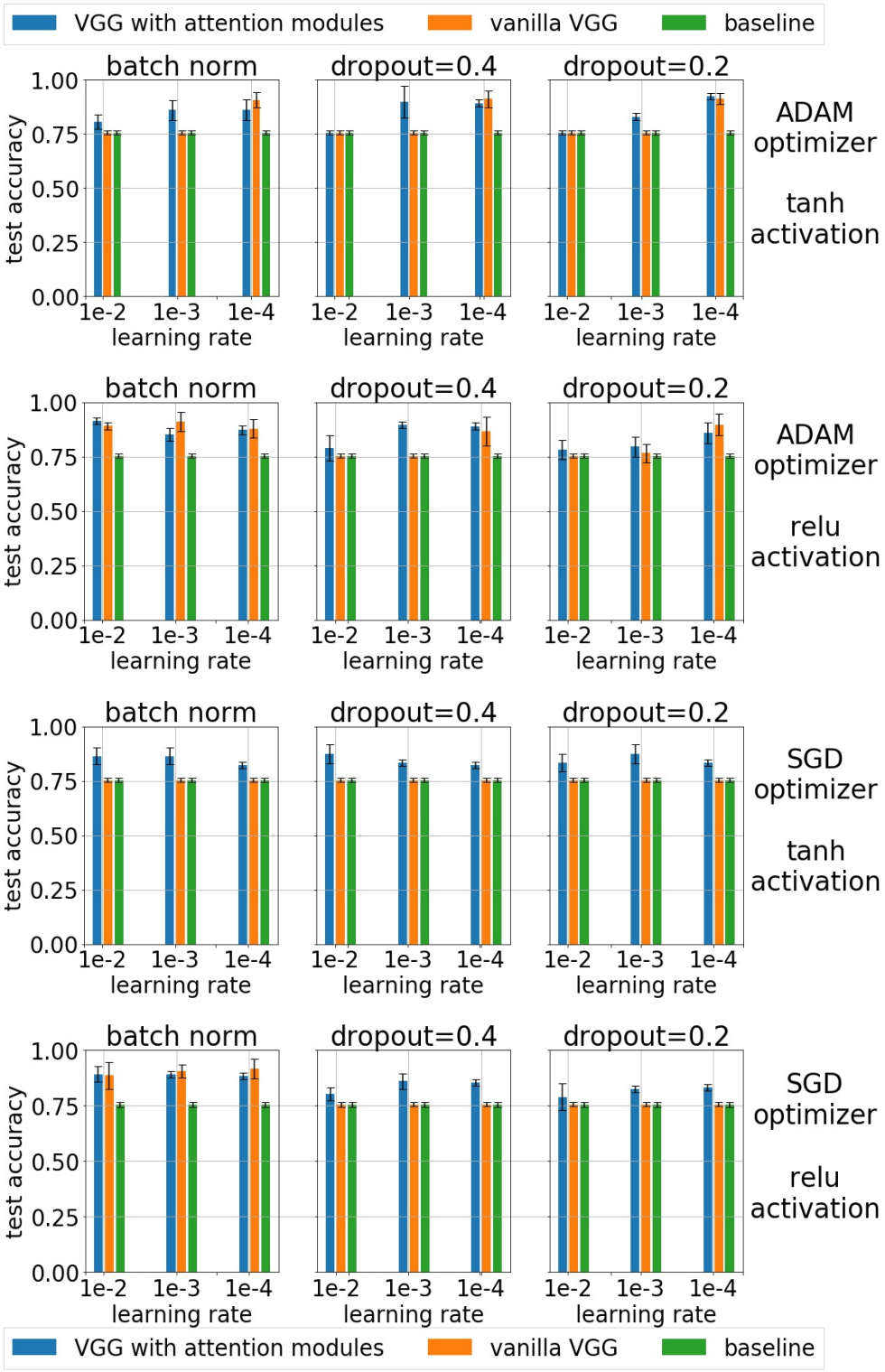
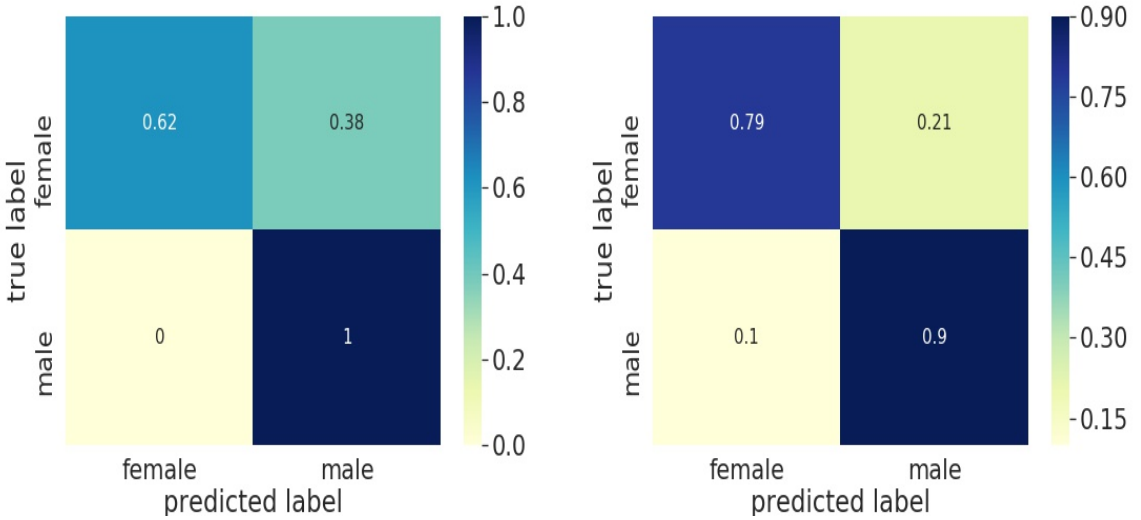


Figure 7.24: Gender classification accuracies across using different optimizers and network architectures. The test accuracies generally increase with decreasing learning rates. The attention VGG outperforms the baseline for all architectures and optimizers.



(a) Attention VGG test accuracy for female classification lesser than 75% benchmark but better than random guessing.

(b) Vanilla VGG class-wide accuracy is better than the acceptable 75% benchmark.

Figure 7.25: Gender classification test accuracies for both: (7.25a) attention and (7.25b) vanilla VGG greatly exceed the 75% benchmark for males. The female prediction with the attention VGG is less than the 75% benchmark but better than random guessing.

some female lobsters not having wider tails when they are not bearing any eggs. The test accuracy for female prediction is still better than random guessing.

Figure 7.26 shows images of mis-classified lobsters. The images have light shadows around the tail regions and since the gender is determined by looking at the tail width, this could be the possible reason for their mis-classification. The classifier could incorrectly associate the shadow as an extension of the tail. For future consideration, computer vision methods for contrast improvement, such as histrogram equalization, can be used to remove shadows from the image before feeding to the classifier, which should improve test accuracies for these cases.

7.3.4 Tail classification

Figure 7.27a and 7.27b shows tail spread classification test accuracies and $f1$ scores with the number of attention heads (1 - 9). There is considerable overlap in test accuracies for the five values indicating that accuracy does not vary strongly with the number of attention heads. A student t-test performed for $k = 7$ and $k = 9$ yields

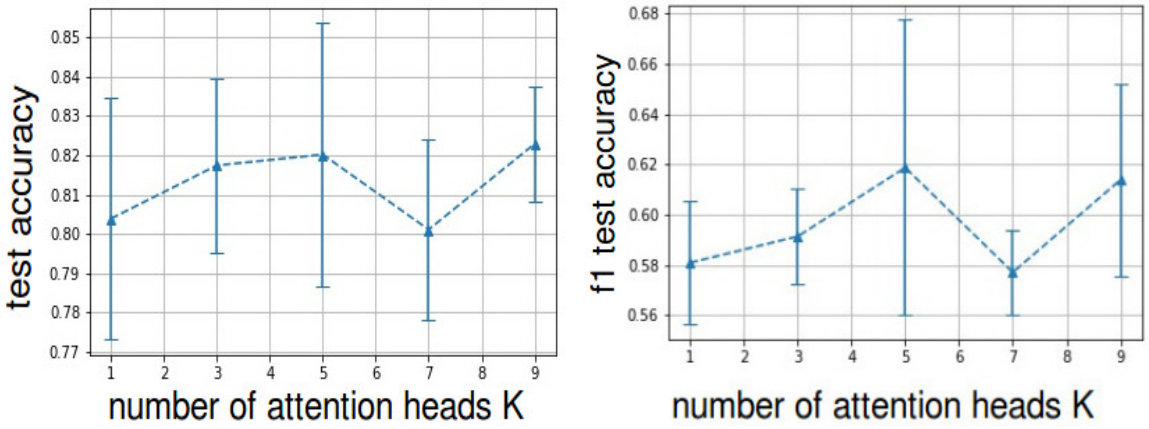
(a) Male mis-classified as female. The shadow on the tail section may be incorrectly associated with the tail and give it an artificially wider (female) tail.

(b) Male mis-classified as female. Same reason as (a).

(c) Female mis-classified as male. The narrower female tail is due to it not presently bearing eggs.

(d) Female mis-classified as male. Same reason as (c).

Figure 7.26: Examples of lobster gender mis-classifications due to: (7.26a), (7.26b) shadows on the male tail section and (7.26c), (7.26d) females not at the egg-bearing stage.



(a) The test accuracies remain similar across the number of attention heads, K.

(b) The $f1$ test scores remain similar across the number of attention heads, K.

Figure 7.27: *Tail spread classification* for: (Fig 7.27a) test accuracies and (7.27b) $f1$ scores are similar across the number of attention heads K.

a p-value of 0.16 (> 0.05 threshold) which fails to reject the null hypothesis. Hence the test accuracies does not differ across $k = 7$ and $k = 9$. The attention modules do not make much difference in the accuracies.

Figure 7.28 shows the variation in *tail spread classification* test accuracies across different architectures for a modified VGG16 neural network. The architecture is compared without (orange), and with, attention modules (blue). The baseline is depicted in green. The baseline accuracy is calculated by predicting the most common label. The test accuracies for the vanilla VGG are similar to the baseline accuracies for learning rate equal to 1e-2. For learning rates 1e-3 and 1e-4, the vanilla VGG and attention VGG have similar test accuracies and outperform the baseline. Attention VGG outperforms baseline for all network architectures.

Table 7.10 shows tail spread classification training / testing accuracies and $f1$ scores for vanilla VGG and VGG with attention modules. Both models achieve comparable test accuracies of 85%. A student t-test performed on the test accuracies from five-fold cross validation across both models yields a p-score of 0.39 (> 0.05 threshold). This fails to reject the null hypothesis which implies that the distributions are equal. Hence the accuracies from both models are comparable. The attention modules do not make much difference.

Fig 7.29 shows a *tail spread classification* confusion matrix. Attention VGG7.29a

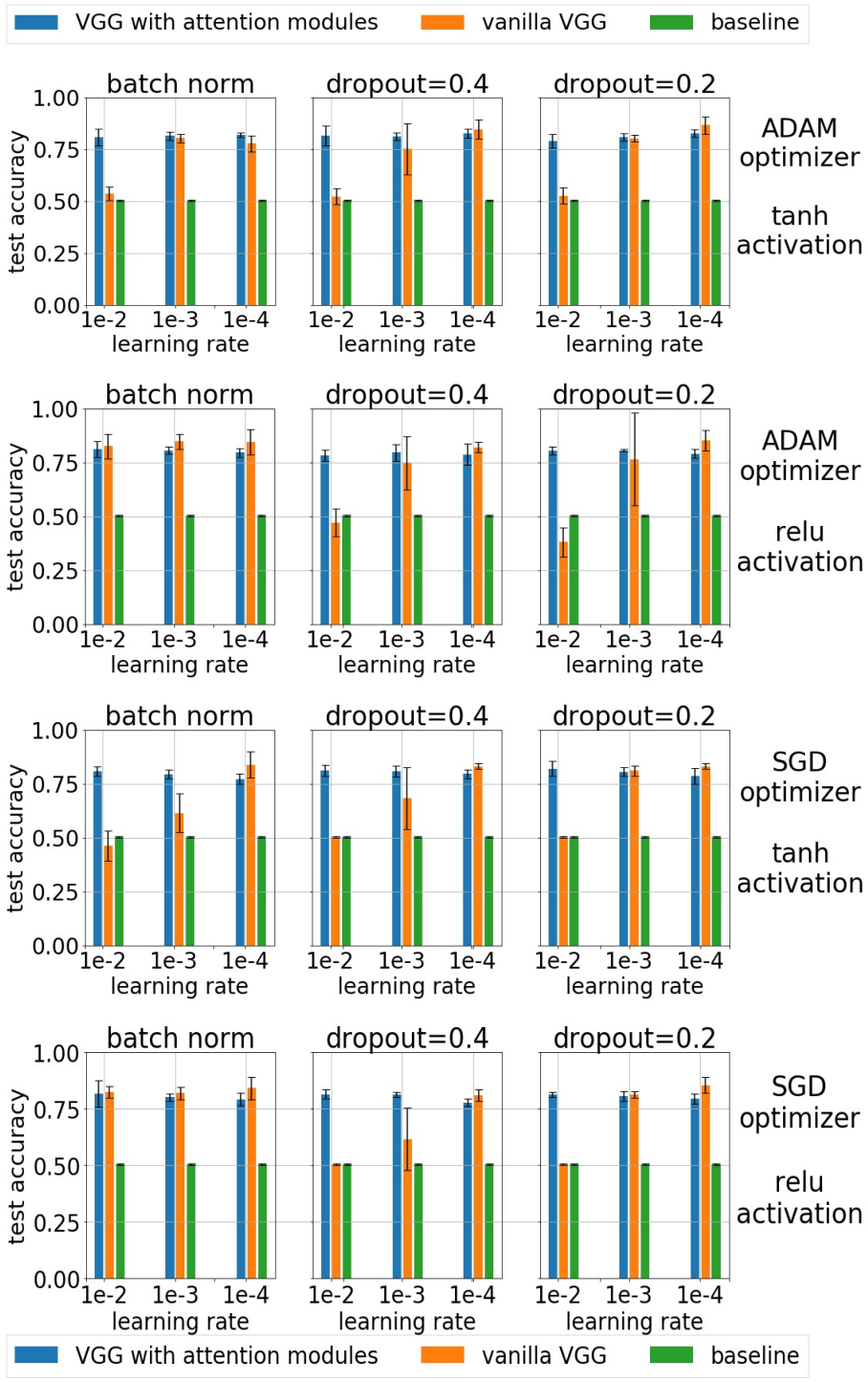


Figure 7.28: *Tail spread classification* accuracies using different optimizers and network architectures. Vanilla VGG is similar to baseline and outperformed by attention VGG when using 1e-2 learning rates with ADAM/SGD. With 1e-3 and 1e-4 learning rates attention VGG and vanilla VGG are similar to one another. The baseline is the accuracy obtained by predicting the most common label.

Table 7.10: VGG (vanilla and with attention) performance test accuracies of 85% are similar for both models. Both models exceed the baseline accuracy which predicts the most common label.

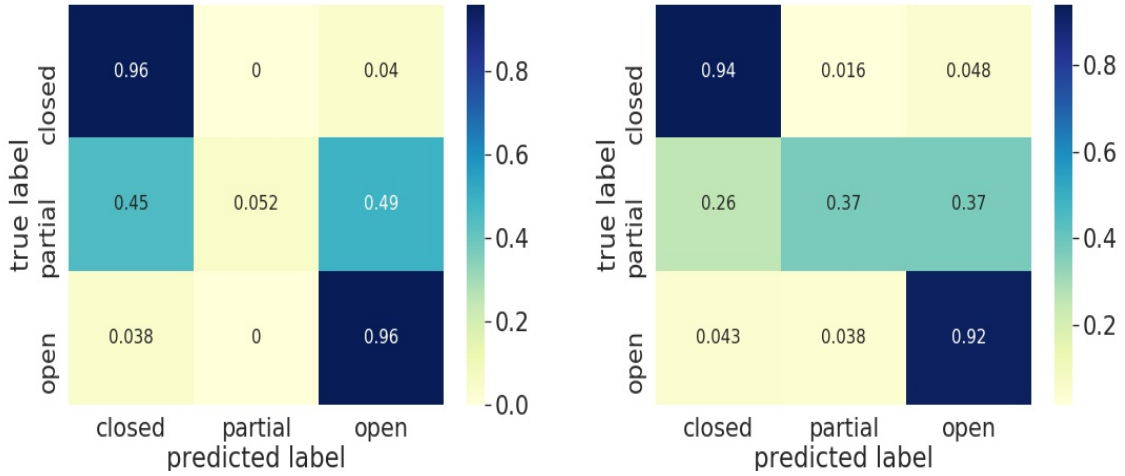
model	vanilla VGG		VGG with attention		baseline	
	mean	std dev	mean	std dev	mean	std dev
train accuracy	0.85	0.03	0.85	0.02	0.50	0.003
test accuracy	0.84	0.03	0.82	0.03	0.50	0.003
train f1 score	0.76	0.08	0.67	0.07	0.35	0.001
test f1 score	0.74	0.07	0.62	0.06	0.35	0.001

classifies the closed and open tail with mean 96% (4% standard deviation) and 96% (2% standard deviation) accuracy respectively. The class-wise accuracy for fully open and fully closed tail is greater than the acceptable benchmark of 75%. However, the partially open tail are classified with accuracy 5% which is less than the acceptable benchmark of 75%. The vanilla VGG7.29b also has less accuracy than the acceptable benchmark of 75%. Both classifiers struggle with the classification of partially open tails. This is not unexpected given the ambiguous nature of partially open tails. A larger class of partially open tails might be a start to address this.

Fig7.31 shows examples of tails mis-classified by the VGG16. The mis-classified images have partially opened tails which the classifier could consider as fully open or fully closed since there is no intermediate stage clearly defined as partially open. This is reflected in the confusion matrix as well. For future consideration, instead of classifying three categories i.e open, partially open and closed, it would be more effective to regress for a measure of tail spread on a scale of 0 to 1. Subsequently, the categories can be assigned different ranges on the scale in an attempt to quantify the degree of tail open-ness.

7.3.5 Abdominal somite bulge classification

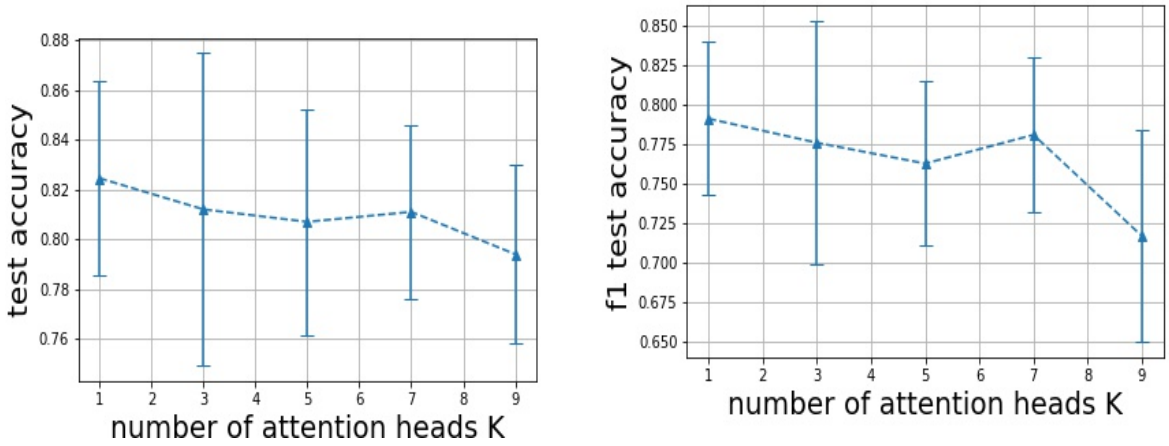
Figures 7.30a and 7.30b show abdominal somite classification test accuracies and $f1$ scores with number of attention heads (1 - 9). There is considerable overlap in test accuracies for the the five values indicating that accuracy does not vary strongly with the number of attention heads. A student t-test performed for $k = 1$ and $k = 9$ gives a



(a) Augmented attention VGG closed and open tail classification is $> 75\%$. The test accuracy for partially open tail classification accuracy is worse than random guessing.

(b) Vanilla VGG closed and open tail classification is $> 75\%$. The test accuracy for partially open tail classification accuracy is worse than random guessing.

Figure 7.29: *Tail spread classification* test accuracy for closed and open tail using (7.25a) attention and (7.25b) vanilla VGG exceeds the 75% benchmark. The test accuracy for partially spread tail with attention or vanilla VGG is much poorer than random guessing.



(a) The test accuracies remain similar across number of attention heads K.

(b) The $f1$ scores remain similar across number of attention heads K.

Figure 7.30: *Abdominal bulge classification*: (7.30a) test accuracies and (7.30b) $f1$ scores are similar across number of attention heads, K. This means the attention modules do not make much difference.

(a) Partially open tail mis-classified as fully open. Not much difference between the two.

(b) Partially open tail mis-classified as fully open. Same reason as (a).

(c) Partially open tail mis-classified as closed. The full tail is not visible.

(d) Closed tail mis-classified as partially open. Not much visible difference between the two.

Figure 7.31: Examples of lobster tail spread mis-classification: (7.31a), (7.31b) and (7.31c) are partially open mis-classified as fully open and (7.31d) closed tail mis-classified as partially open. The classifier struggles to classify partially open tails which can be mis-classified as fully open or closed.

Table 7.11: VGG (vanilla and with attention) performance test accuracies for *abdominal bulge classification* show similar accuracies and $f1$ scores. Both models exceed the baseline model which predicts the most common label.

model	vanilla VGG		VGG with attention		baseline	
	mean	std dev	mean	std dev	mean	std dev
train accuracy	0.97	0.02	0.92	0.05	0.69	0.003
test accuracy	0.79	0.04	0.82	0.04	0.69	0.003
train f1 Score	0.97	0.02	0.89	0.07	0.35	0.001
test f1 Score	0.73	0.05	0.79	0.05	0.35	0.001

p-value of 0.29 (> 0.05 threshold) which fails to reject the null hypothesis. Therefore, the test accuracies does not differ across $k = 1$ and $k = 9$ and the attention modules do not make much difference.

Figure 7.32 shows the abdominal bulge classification test accuracies across different architectures for the modified VGG16 neural network (Figure 7.8). The architecture is compared without (orange), and with, attention modules (blue). The baseline accuracy is in green. The baseline accuracy is obtained from predicting the most common label. The trend in test accuracies of vanillla VGG is affected by the choice of architecture. The test accuracies generally follow an increasing trend with decreasing learning rates across 1e-2,1e-3 and 1e-4. The attention augmented VGG accuracies do not change across different architecture and optimizers. The attention VGG outperforms vanilla VGG when using learning rate 1e-2 across architectures and optimizers. The vanilla VGG performs best with the lowest learning rate 1e-4.

Table 7.11 shows abdominal somite classification train / test accuracies and $f1$ scores of vanilla VGG and VGG with attention modules. Both models achieve comparable test accuracies of 80%. A student t-test performed on the test accuracies from five-fold cross validation across both models gives a p-score of 0.24 (> 0.05 threshold). This fails to reject the null hypothesis which means the distributions are equal. Therefore, the accuracies from both models are comparable.

Figure 7.33 shows an abdominal bulge classification confusion matrix. The test accuracies for predicting a lobster with no abdominal bulge using an attention VGG Fig. 7.33a or vanilla VGG Fig. 7.33b is better than the acceptable benchmark

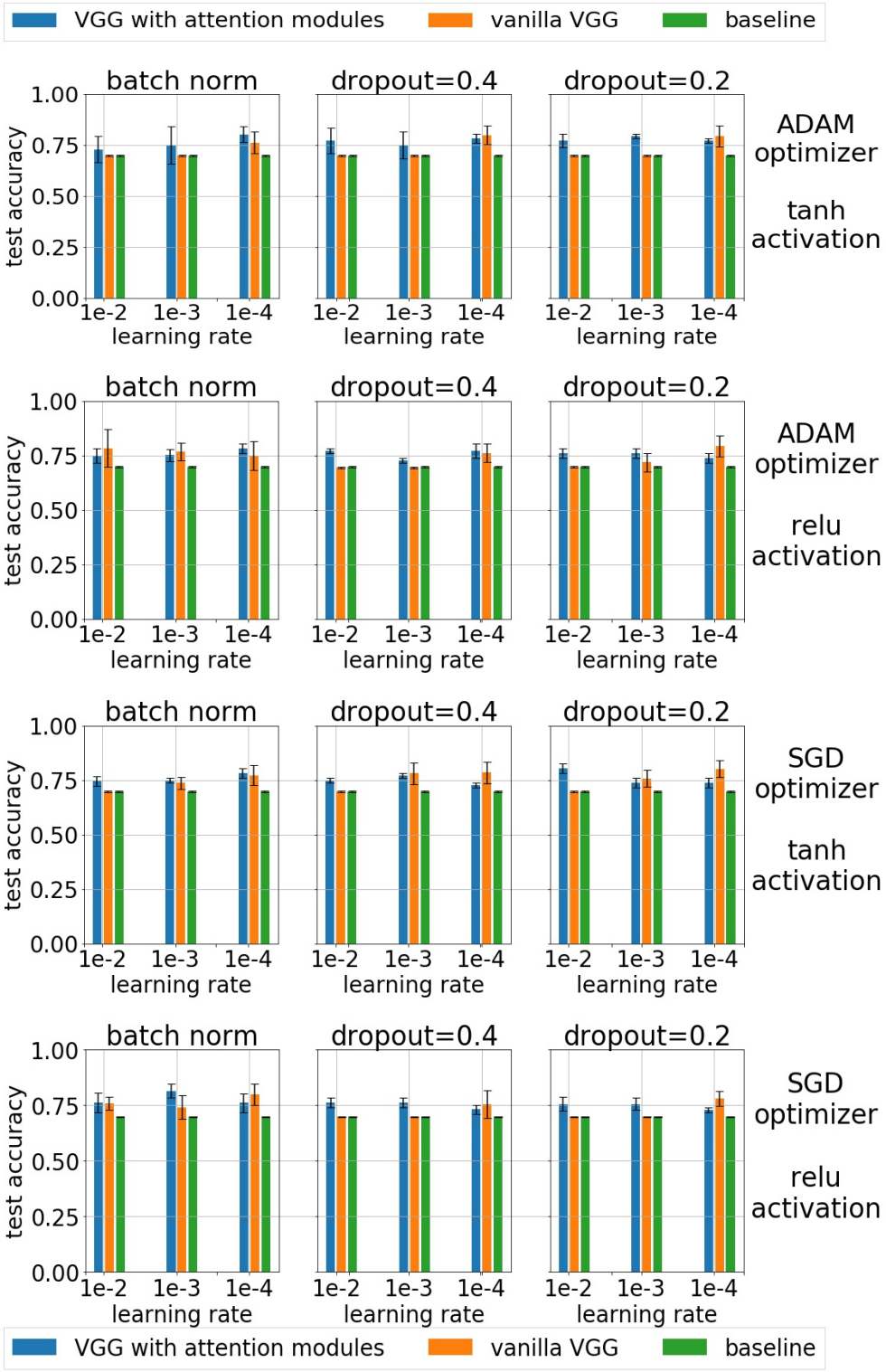
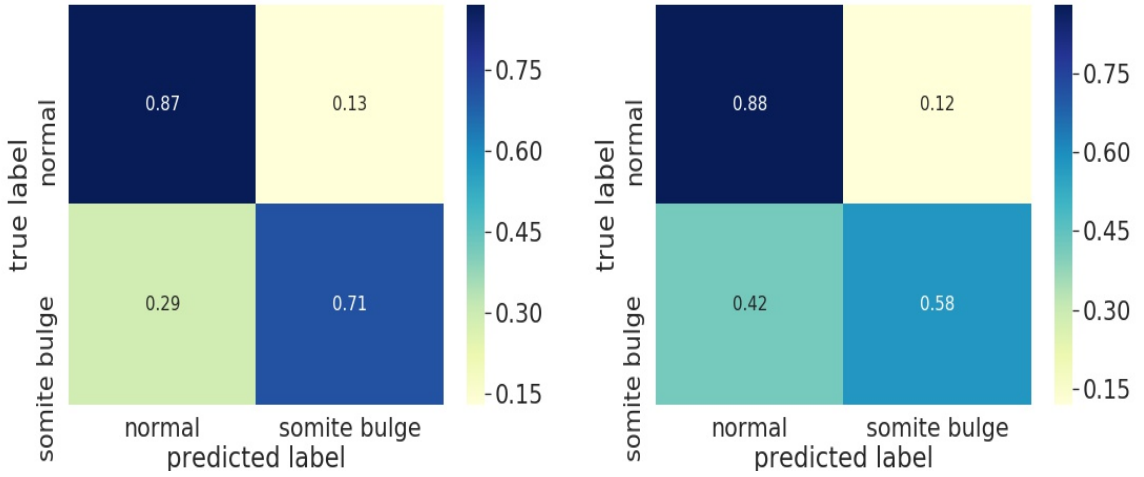


Figure 7.32: Abdominal somite classification accuracies using different optimizers and network architectures. The test accuracies for vanilla VGG generally increase with decreasing learning rates. The attention VGG exceeds the baseline for all architectures and optimizers.



(a) Attention VGG test accuracy is better than random guessing but < the 75% benchmark to detect an abdominal bulge.

(b) Vanilla VGG test accuracy better than random guessing but < the 75% benchmark to detect an abdominal bulge.

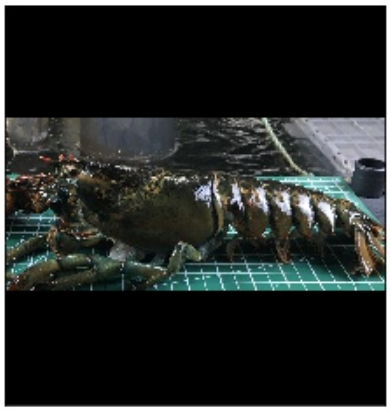
Figure 7.33: *Abdominal bulge classification* test accuracy with: (7.33a) attention and (7.33b) vanilla VGG is only better than random guessing. The test accuracy for a normal abdominal classification with both models exceeds the 75% benchmark.

accuracy of 75%. However, the test accuracy for predicting a lobster with a somite bulge using an attention VGG or vanilla VGG is less than the acceptable benchmark accuracy of 75% but better than random guessing. It appears the abdominal bulge is dependent on the lobster pose as well.

Figure 7.34 shows examples of images mis-classified abdominal bulges by the VGG16. The images show lobsters in aggressive postures, which will distend the abdomin, may be a reason for the mis-classification. For future consideration, the lobster images with aggressive postures could be filtered out with the aggressive posture classifier in the Section.7.3.1.

7.4 Concluding Remarks

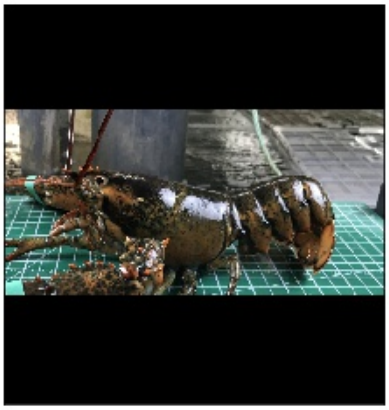
The vanilla VGG and VGG16 with attention modules across different network architectures and optimizers were evaluated. The best architectures for both vanilla VGG16 and VGG16 with attention modules all have similar accuracies across the five classifiers i.e gender, abdominal bulge, crusher claw location, tail spread, and posture. However, unlike the vanilla VGG16, the attention VGG16 outperforms the



(a) False classification of abdominal bulge possibly due to the lobster raising its tail

(b) False classification of abdominal bulge possibly due to the lobster raising its head

(c) False classification of abdominal bulge possibly due to different image perspective angle.



(d) False detection of abdominal bulge possibly due to the lobster curling its tail.

Figure 7.34: Examples of incorrect abdominal bulge classification. In Figs. 7.34a,7.34b and 7.34c the abdominal bulge is falsely detected possibly due to lobster activity. Fig 7.34d is a slightly different angle of the lobster pose which may lead to false detection.

baseline for all architectures and learning rates. The test accuracies for vanilla VGG16 generally increases for decreasing learning rates. The attention VGG allows larger learning rates by possibly making the loss curve flatter unlike the vanilla VGG16 loss curve which is sharper since the network fails to optimize with larger learning rates.

Chapter 8

Conclusion

It is possible for machine learning techniques, like convolutional neural networks, to assess lobster attributes to high accuracies with a processing rate of 48 images per second. The methods presented in Chapter 7 assess the qualitative traits of lobsters. Table 8.1 summarizes the proposed CNN models' achieved accuracies.

The CNN-based methods can map landmark points on a lobster and extract different morphometric distances on the lobster body such as carapace and tail lengths, widths, etc. The carapace length is useful in faclilating lobster fishing regulations that do not allow catching lobster below a certain length.

The mean squared error in landmark detection is 11.45 pixels which is achieved using the attention-based CNN as explained in Chapter 6. The error is 4% of an average lobster length (300 pixels from the dataset or 25-30 in cm) which is an acceptable error and provides a rapid means to assess a catch.

The methodology presented in Chapter5 using Siamese networks to identify unique lobsters is able to achieve a top-3 accuracy of about 84%. In addition to high accuracies, the processing time taken by these machine learning models is near real-time using a laptop with an Intel core i7-7700 processor with 16 GB RAM and GeForce GTX 1060 GPU. This inferencing can be performed on the order of milliseconds which

Table 8.1: Lobster classification accuracies with proposed CNN architectures

attribute	classification accuracy (%)
aggressive posture	85
crusher claw location	80
gender	90
tail spread	82
abdominal bulge	80

is necessary in order to process lobsters at a pace expected in a processing plant.

Addtionally, the use of transfer learning to achieve high accuracies from small datasets was demonstrated in several chapters. The use of pre-trained architectures like VGG16, Resnet50, etc. to achieve high accuracies was demonstrated for classification and regression tasks for lobster traits like lobster gender and landmark mappings, respectively.

The other contribution was the use of an attention mechanism initially proposed by Rodriguez et. al.[67] for fine-grained classification. This mechanism was modified for landmark localization and was demonstrated to perform better than vanilla CNNs (Chapter 6). This mechanism can be added on top of neural networks without changing the existing neural network pathways and thus it is possible to make use of transfer learning. The attention mechanism was evaluated for fine-grained classification in Chapter 7 across different architectures and learning rates. The CNN perfomed with similar accuracies both with, and without, the attention mechanism, but the vanilla CNN was more sensitive to the choice of architecture and learning rates compared to the attention CNN.

The machine learning methods presented in Chapters 6 and 7 assess the mutable and immutable traits of lobsters. The immutable traits of the lobster are persistent and include the crusher claw location, gender, abdominal bulge, landmark mapping (Chapter 6) and lobster size(Chapter 3). These immutable traits can be components of a profile that could be used to potentially tag individual lobsters towards greater lobster traceability.

The more variable traits like tail spread and posture give an indication of the conditions the lobsters are maintained in prior to market. They contribute less to lobster traceability since these traits could change with the lobster's environment.

The combined confidence score of the tail spread, posture and abdominal bulge attributes is 0.6 (aggregate product of the test accuracies). This is useful because even a 50% reduction in filtering out lobsters in poor health results in a quality improvement of 15% (assuming 30% of lobsters are in poor health for whatever reasons).

The lobster gender classifier can correctly classify female lobsters 70% of the time. This is useful since female lobsters are a delicacy and sellers can demand higher prices for 20% of their stock (on average, 30% of a lobster catch is female).

The thesis provides a solution for lobster traceability and sustainability. The sustainability is maintained by filtering female lobsters and lobsters whose carapace is lesser than the regulated length. Currently, there are no automated methods for regulating the lower limit in carapace length as well the female lobster catch. The trained human observers can monitor 1.5-2% of the fleet and if the automation can capture at least 50% of the lobster fleet, that would results in at least 10% increase in lobster female population replenished into the ocean(aggregate product of 50%,30%,70%). Additionally, the landmark detection algorithm can perform landmark localization with an error of 0.5-1cm. Assuming the regulation length of 8cm, approximately 87% of the lobsters under the regulation length can be replenished back into the ocean.

8.1 Future Work

In Chapter 7, a comparison was performed between VGG and the attention VGG for five datasets across different architectures and learning rates. Although the attention and vanilla VGG accuracies were similar, the vanilla VGG was observed to be more sensitive than attention VGG for different architectures and learning rates. This observation is similar to the work by Li et al.[50] where they visualize the loss landscape for Resnet-50 (Section 2.2) with and without skip connections. The Resnet-50 was observed to have a non-convex and more chaotic landscape without the skip connections and led to training problems. In the future, it would be more revealing to perform a loss landscape comparison for CNN with and without VGG and observe the effect of the attention mechanism on the geometry of the landscape.

Existing state-of-the-art methods make use of heatmap-based regression methods for landmark localization. These methods are more robust to challenges such as occlusions. It would be useful to observe the landmark localization performance on the lobster dataset mentioned in Chapter6 using techniques by Newell et al.[60] and Payer et al.[64].

There are other lobster body traits which could be classified and tracked for improved lobster traceability as well as indications of lobster health. For example, lobsters kept in cramped spaces do not have full length antennae because of the claws of other lobsters. Lobsters may also be missing limbs or entire claws for the same reason. Additionally, lobster shell disease is an indication of poor lobster health caused

Figure 8.1: Shell disease spread across the carapace and tail is measure of lobster health.

by a certain bacteria (Figure 8.1). This shell disease can be transmitted to other surrounding lobsters. Future work may consider machine learning methods to classify lobsters with shell disease and missing or partial claws, limbs and antennae which can enrich the database for lobster traceability.

Bibliography

[1] Lobster uk hatchery: A marine conservation charity in the uk website: https://www.nationallobsterhatchery.co.uk/.

[2] Aleju. aleju/imgaug, July 2015.

[3] Peter N Belhumeur, João P Hespanha, and David J Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. Technical report, Yale University New Haven United States, 1997.

[4] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[5] Bridget Benson, Junguk Cho, Deborah Goshorn, and Ryan Kastner. Field programmable gate array (fpga) based fish detection using haar classifiers. *American Academy of Underwater Sciences*, 2009.

[6] Abhir Bhalerao and Gregory Reynolds. Ruler detection for autoscaling forensic images. *International Journal of Digital Crime and Forensics (IJDCF)*, 6(1):9–27, 2014.

[7] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[8] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[9] Cgvict. cgvict/rolabelimg, Oct 2017.

[10] Timothy F Cootes, Christopher J Taylor, David H Cooper, and Jim Graham. Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59, 1995.

[11] Corrado Costa, F Antonucci, C Boglione, P Menesatti, Marc Vandeputte, and B Chatain. Automated sorting for size, sex and skeletal anomalies of cultured seabass using external shape analysis. *Aquacultural engineering*, 52:58–64, 2013.

[12] Corrado Costa, Francesca Antonucci, Federico Pallottino, Jacopo Aguzzi, Da-Wen Sun, and Paolo Menesatti. Shape analysis of agricultural products: a review of recent research advances and potential application to computer vision. *Food and Bioprocess Technology*, 4(5):673–692, 2011.

[13] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4109–4118, 2018.

[14] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and image processing*, 14(3):227–248, 1980.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[16] Amit Dhomne, Ranjit Kumar, and Vijay Bhan. Gender recognition through face using deep learning. *Procedia Computer Science*, 132:2–10, 2018.

[17] Piotr Dollár, Peter Welinder, and Pietro Perona. Cascaded pose regression. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1078–1085. IEEE, 2010.

[18] Eran Eidinger, Roee Enbar, and Tal Hassner. Age and gender estimation of unfiltered faces. *IEEE Transactions on Information Forensics and Security*, 9(12):2170–2179, 2014.

[19] Zhen-Hua Feng, Josef Kittler, Muhammad Awais, Patrik Huber, and Xiao-Jun Wu. Wing loss for robust facial landmark localisation with convolutional neural networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2235–2245. IEEE, 2018.

[20] Fisheries and Oceans Canada. nf.18.063 lobster carapace length, Apr 2018.

[21] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[23] Thierry Gosselin, Bernard Sainte-Marie, and Jean-Marie Sévigny. Individual identification of decapod crustaceans ii: Natural and genetic markers in snow crab (chionoecetes opilio). *Journal of Crustacean Biology*, 27(3):399–403, 2007.

[24] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *ICCV 2009-International Conference on Computer Vision*, pages 498–505. IEEE, 2009.

[25] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *null*, pages 1735–1742. IEEE, 2006.

[26] Kai Han, Jianyuan Guo, Chao Zhang, and Mingjian Zhu. Attribute-aware attention model for fine-grained representation learning. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 2040–2048. ACM, 2018.

[27] Snigdhaa Hasija, Manas Jyoti Buragohain, and S Indu. Fish species classification using graph embedding discriminant analysis. In *Machine Vision and Information Technology (CMVIT), International Conference on*, pages 81–86. IEEE, 2017.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] GR Hillman, B Wursig, GA Gailey, N Kehtarnavaz, A Drobyshevsky, BN Araabi, HD Tagare, and DW Weller. Computer-assisted photo-identification of individual marine vertebrates: a multi-species system. *Aquatic Mammals*, 29(1):117–123, 2003.

[30] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[31] Shota Horiguchi, Daiki Ikami, and Kiyoharu Aizawa. Significance of softmax-based features in comparison to distance metric learning-based features. *arXiv preprint arXiv:1712.10151*, 2017.

[32] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[33] Ching-Lu Hsieh, Hsiang-Yun Chang, Fei-Hung Chen, Jhao-Huei Liou, Shui-Kai Chang, and Ta-Te Lin. A simple and effective digital imaging approach for tuna fish length measurement compatible with fishing operations. *Computers and Electronics in Agriculture*, 75(1):44–51, 2011.

[34] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[35] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[36] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[37] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[38] Ian Jolliffe. *Principal component analysis*. Springer, 2011.

[39] Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

[40] A Khan and Bhupesh Gour. Gender classification technique based on facial features using neural network. *Int J Comput Sci Inf Technol*, 4:839–43, 2013.

[41] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[42] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.

[43] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[44] DA Konovalov, JA Domingos, C Bajema, RD White, and DR Jerry. Ruler detection for automatic scaling of fish images. In *Proceedings of the International Conference on Advances in Image Processing*, pages 90–95. ACM, 2017.

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[46] Hugo Larochelle and Geoffrey E Hinton. Learning to combine foveal glimpses with a third-order boltzmann machine. In *Advances in neural information processing systems*, pages 1243–1251, 2010.

[47] Rasmus Larsen, Hildur Olafsdottir, and Bjarne Kjær Ersbøll. Shape and texture based classification of fish species. In *Scandinavian Conference on Image Analysis*, pages 745–749. Springer, 2009.

[48] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[49] Gil Levi and Tal Hassner. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–42, 2015.

[50] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399, 2018.

[51] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[52] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015.

[53] Tony Lindeberg. Scale invariant feature transform. 2012.

[54] Lei Liu, Zongxu Pan, and Bin Lei. Learning a rotation invariant detector with rotatable bounding box. *arXiv preprint arXiv:1711.09405*, 2017.

[55] Alison B MacDiarmid, Megan D Oliver, Robert A Stewart, and Dharini Gopal. Conservation of unique patterns of body markings at ecdysis enables identification of individual spiny lobster, jasus edwardsii. 2005.

[56] Ruiyi Mao, Qian Lin, and Jan P Allebach. Robust convolutional neural network cascade for facial landmark localization exploiting training data augmentation. *Electronic Imaging*, 2018(10):374–1, 2018.

[57] Philipp Mitteroecker and Philipp Gunz. Advances in geometric morphometrics. *Evolutionary Biology*, 36(2):235–247, 2009.

[58] Baback Moghaddam, Tony Jebara, and Alex Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, 2000.

[59] NaturalIntelligence. Naturalintelligence/imglab, Oct 2017.

[60] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[61] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*, 2018.

[62] SO Ogunlana, O Olabode, SAA Oluwadare, and GB Iwasokun. Fish classification using support vector machine. *African Journal of Computing & ICT*, 8(2):75–82, 2015.

[63] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.

[64] Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler. Regressing heatmaps for multiple landmark localization using cnns. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 230–238. Springer, 2016.

[65] Talmo D Pereira, Diego E Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S-H Wang, Mala Murthy, and Joshua W Shaevitz. Fast animal pose estimation using deep neural networks. *bioRxiv*, page 331181, 2018.

[66] Dhruv Rathi, Sushant Jain, and Dr S Indu. Underwater fish species classification using convolutional neural network and deep learning. *arXiv preprint arXiv:1805.10106*, 2018.

[67] Pau Rodrıguez, Josep M Gonfaus, Guillem Cucurull, F Xavier Roca, and Jordi Gonzalez. Attend and rectify: a gated attention mechanism for fine-grained recovery. 2018.

[68] V Louise Roth. On homology. *Biological Journal of the Linnean Society*, 22(1):13–29, 1984.

[69] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[70] Narin Sontigun, Kabkaew L Sukontason, Barbara K Zajac, Richard Zehner, Kom Sukontason, Anchalee Wannasan, and Jens Amendt. Wing morphometrics as a tool in species identification of forensically important blow flies of thailand. *Parasites & vectors*, 10(1):229, 2017.

[71] Concetto Spampinato, Daniela Giordano, Roberto Di Salvo, Yun-Heh Jessica Chen-Burger, Robert Bob Fisher, and Gayathri Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on Analysis and retrieval of tracked events and motion in imagery streams*, pages 45–50. ACM, 2010.

[72] Richard E Strauss and Fred L Bookstein. The truss: body form reconstructions in morphometrics. *Systematic Biology*, 31(2):113–135, 1982.

[73] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[74] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[75] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3476–3483, 2013.

[76] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1891–1898, 2014.

[77] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[78] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[79] Laszio Tora, John White, Christel Brou, Dlane Tasset, Nicholas Webster, Elisabeth Scheer, and Plerre Chambon. The human estrogen receptor has two independent nonacidic transcriptional activation functions. *Cell*, 59(3):477–487, 1989.

[80] George Trigeorgis, Patrick Snape, Mihalis A Nicolaou, Epameinondas Antonakos, and Stefanos Zafeiriou. Mnemonic descent method: A recurrent process applied for end-to-end face alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4177–4187, 2016.

[81] Tzutalin. tzutalin/labelimg, Sept 2015.

[82] Kazuaki Ueda, Takashi Baba, Yuji Nakagawa, and Kaname Amano. Detection of scale intervals in digital images. In *Data Engineering Workshops, 2005. 21st International Conference on*, pages 1232–1232. IEEE, 2005.

[83] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[84] DJ White, C Svellingen, and NJC Strachan. Automated measurement of species and length of fish by computer vision. *Fisheries Research*, 80(2-3):203–210, 2006.

[85] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.

[86] Xiangyang Xu, Shengzhou Xu, Lianghai Jin, and Enmin Song. Characteristic analysis of otsu threshold and its applications. *Pattern recognition letters*, 32(7):956–961, 2011.

[87] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[88] Lei Yue, Xin Miao, Pengbo Wang, Baochang Zhang, Xiantong Zhen, and Xianbin Cao. Attentional alignment network.

[89] Sebastian Zambanini, Michael Herrmann, and Martin Kampel. An automatic method to determine the diameter of historical coins in images. In *Scientific Computing and Cultural Heritage*, pages 99–106. Springer, 2013.

[90] Jie Zhang, Shiguang Shan, Meina Kan, and Xilin Chen. Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In *European Conference on Computer Vision*, pages 1–16. Springer, 2014.

[91] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. Learning multi-attention convolutional neural network for fine-grained image recognition. In *Int. Conf. on Computer Vision*, volume 6, 2017.

[92] Erjin Zhou, Zhimin Cao, and Qi Yin. Naive-deep face recognition: Touching the limit of lfw benchmark or not? *arXiv preprint arXiv:1501.04690*, 2015.

[93] Zhi-Hua Zhou. When semi-supervised learning meets ensemble learning. In *International Workshop on Multiple Classifier Systems*, pages 529–538. Springer, 2009.

[94] Donald W Zimmerman and Bruno D Zumbo. Rank transformations and the power of the student t test and welch t'test for non-normal populations with unequal variances. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 47(3):523, 1993.