

The Identification Of The Endangered Whale

Ruoqi Zhang

ruz005@ucsd.edu

Kai Wang

k7wang@ucsd.edu

Yuekuan Luo

y8luo@ucsd.edu

Tianran Zhang

tiz217@ucsd.edu

Abstract

In this project, We are trying to use machine learning to identify the species of whales by analyzing their tails and unique markings found in footage. We downloaded the data from a Kaggle competition and separated the data into two parts: 80% of the data as training data and the others becomes the test data. After the data processing, we used CNN and HOG to extract features from the train dataset, then implemented simple CNN, simple KNN, ResNet-18, ResNet-101, Ensemble Network Inspired by Siamese Network to compare their performances. Based on the features extracted by HOG, we can reach the highest accuracy using KNN.

Keywords: Image classification, HOG, SVM, CNN, KNN, ResNet, Siamese Network

1. Introduction

It is well known that over-exploitation by the whaling industry led to serious declines in many of the world's populations of whales, although thankfully no species was brought to extinction and many are now in the process of recovering, although not all. After centuries of intense whaling, recovering whale populations still have a hard time adapting to warming oceans and struggle to compete every day with the industrial fishing industry for food. To aid whale conservation efforts, scientists use photo surveillance systems to monitor ocean activity. They use the shape of whales' tails and unique markings found in footage to identify what species of whale they're analyzing and meticulously log whale pod dynamics and movements.

For the past 40 years, most of this work has been done manually by individual scientists, leaving a huge trove of data untapped and underutilized. Under the circumstance that a few kinds of whales are endangered, if we are able to improve the accuracy of the identification, the rise of the efficiency of the protection is guaranteed.

Machine learning[2] is a kind of emerging scientific study of algorithms and statistical models which computer systems use for the purpose of performing a specific task effectively without using explicit and detailed instructions,



Figure 1: Capture the shape of the body

and it mainly relies on the intrinsic relationship of the data, such as patterns and inference instead. Because of the fact that under such algorithm, the computers can learn and make predictions themselves, machine learning is seen as a subset of artificial intelligence. Generally speaking, machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions instead of being explicitly programmed to perform the task. The algorithms have been used in a wide variety of applications, such as stock prediction, recommendation systems, email filtering, object classification and computer vision, such as autonomous driving, where it is unfeasible to develop an algorithm of specific instructions to perform the corresponding task.

In the past, people always use the shape obtained from the images to judge the species of the whales. The accuracy is not always ideal due to the fact that this task is completed manually and there are too many classes to identify. Whales' shapes can be regarded as the target of computer vision. If people can use machine learning method to analyze those data deeply, based on the physical function of those whales bodies, it will be a great contribution to help analyze how to protect those endangered whales with targeted plan.

Therefore, in this paper, we are trying to utilize several machine learning methods: the input to our algorithm is an image of whales' tail. We then use SVM, simple CNN, simple KNN, ResNet-18, ResNet-101 and Ensemble Network

Inspired by Siamese Network to improve the accuracy of the whales' classification step by step.

2. Related Works

Researchers have utilized many models in the field of image classification. One main direction is SVM, and the others are CNN and Deep learning.

SVM In 2012, Liu J., Kanazawa A., Jacobs D. and Belhumeur P. [13] proposed a novel approach to fine-grained image classification by using part localization. Their aim of this research is to improve the accuracy of the identification of the dog breed. In their paper, they extract corresponding parts to make classification. Their approach features a hierarchy of parts and breed-specific part location. The recognition rate can reach 90%.

In 2018, G. Mercier and M. Lennon [16] researched SVM with spectral-based kernel to classify satellite hyperspectral images. Their method is able to reduce the false alarm by traditional kernel.

Scholars have also done some similar works: In 2007, Begum Demir and Sarp Erturk [7] used Relevance Vector Machine. They propose that RVM has similar strategy with SVM but require fewer relevance vectors. With lower complexity, RVM can achieve similar classification as SVM.

CNN Another widely used approach in species classification is based on Convolutional neural networks (CNNs). In 2015, Hsu, David [11] approach the dog breed classification by using CNNs based on LeNet and GoogLeNet architectures. The architectures for LeNet include (number of CONV-RELUPOOL) \times N + (number of FC) \times M + FC-120 and this architecture performs pretty well on many classification problems like as ImageNet. As for the GoogLeNet, it can reduce overfitting while maintaining good performance on classification.

Also in 2015, in their work Makantasis et al. [14] show that using CNN, hyperspectral images can be successfully classified. CNN can encode the spectral and spatial features of pixel. The low-to-high hierarchy of features improve the performance of classification greatly.

Deep Learning Recently, deep learning has drawn a lot of attentions in computer vision applications. Some of the tasks, such as classification [12] and detection [17], have achieved human level performance. There are several reasons which guarantee the amazing performance. The first is the emergence of the powerful GPU which allows more and more complicated model to be proposed and trained. Then, large datasets like ImageNet [8] provide tons of images for training which boosts the performance a lot. At last, more and more efficient networks are proposed every year. In addition, deep learning has been applied to many low-vision tasks such as image denoising [18], super resolution [9], and image restoration [15], etc., achieving state-of-the-art performance.

In this experiment, because the quality of the dataset is not ideal that implementing the Deep Learning will not guarantee the high accuracy and it will require large amount of time. Therefore, we try to mainly use CNN and SVM as two directions.

3. Dataset and features

3.1. Dataset

The dataset is associated with the Humpback Whale Identification Challenge data set. We downloaded the data from a kaggle competition ¹. This data set contains thousands of images of humpback whale flukes. We analyzed Happy Whale's database of over 25,000 images, gathered from research institutions and public contributors, and helped to open rich fields of understanding for marine mammal population dynamics around the globe.

There are 20288 images in the training data set with 4571 unique classes (Whale ID) and 5073 images in the test data set. The largest class is new-whale, which is the label of unrecorded humpback whale and most classes (more than 4000) contain less than 10 images.

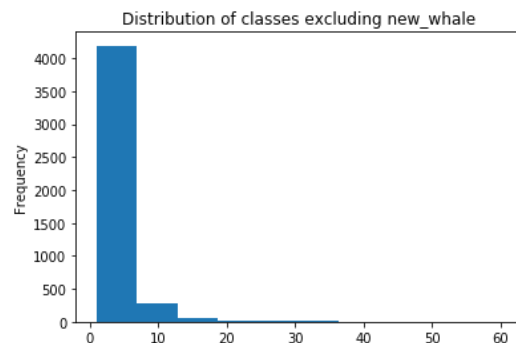


Figure 2: Distribution of the dataset

3.2. Data processing

We randomly shuffle data and use a 80/20 training and testing split.

In our experiment, We apply data augmentation to pre-processing dataset in order to increase the number of training data. We preserve label of each original image in transformation. our exact methods to transform each image are as follows:

- Rotate the images from 0° and 90°
- Flipping horizontally
- Flipping vertically

¹<https://www.kaggle.com/c/humpback-whale-identification/data>

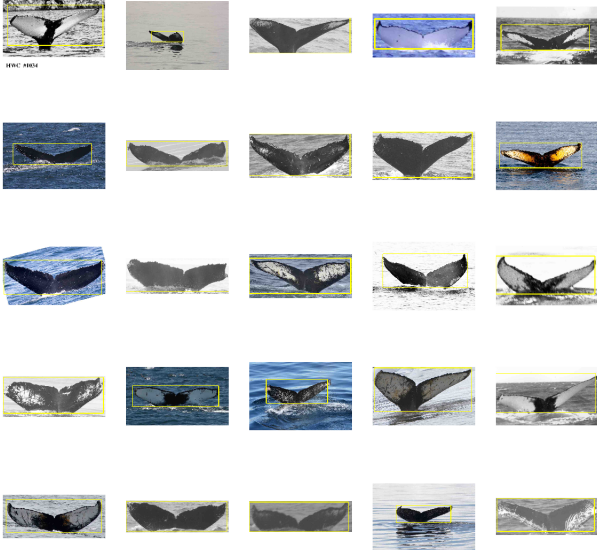


Figure 3: Data process

- Resize: because the original data has diverse height and width, we try to resize their scale to 100*100*3 for ResNet-18, SVM, KNN and Siamese Network, then 160*320*3 for ResNet-101.

We do note that this down-sampling may result in a loss of information. We normalize the image pixel values from 0- 255 to 0-1 then perform mean subtraction over the entire training set. We record this mean and apply it to the validation and testing set.

3.3. Features

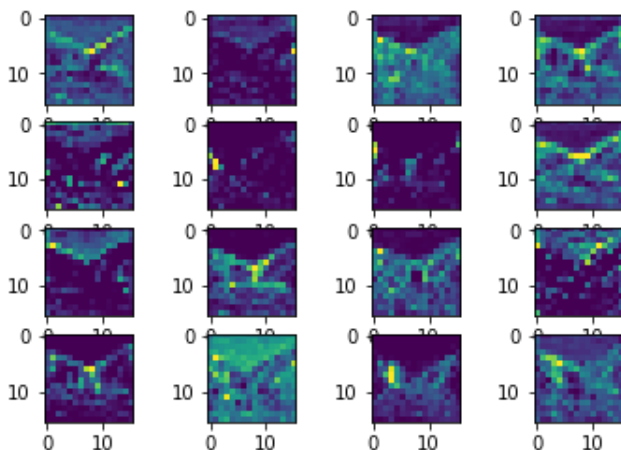


Figure 4: Extraction of the features

In this project, we utilize the simple CNN network to extract higher level features from the image. There are totally

64 channels of features part of which are visualized above. On the account of the fact that the accuracy is not ideal, then we use HOG to extract another version of features.

4. Methods

4.1. Feature extraction

CNN features Several convolution layers followed by max-pooling and an activation function play the role to extract features. The reason why CNN can extract the correct feature representation $\phi(x) \in \mathbb{R}^m$ from the base features $x \in \mathbb{R}^D$ due to the number of trainable weights and biases in the network. Greater this number, more complex transformations can be modeled.

HOG features The technique[6] counts occurrences of gradient orientation in localized portions of an image.

4.2. Models

In this project, totally six models are utilized: SVM, Simple CNN Network, Simple KNN, ResNet-18, ResNet-101 (Transfer Learning), Ensemble Network Inspired by Siamese Network.

SVM Support-vector machine(SVM)[4] are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting).

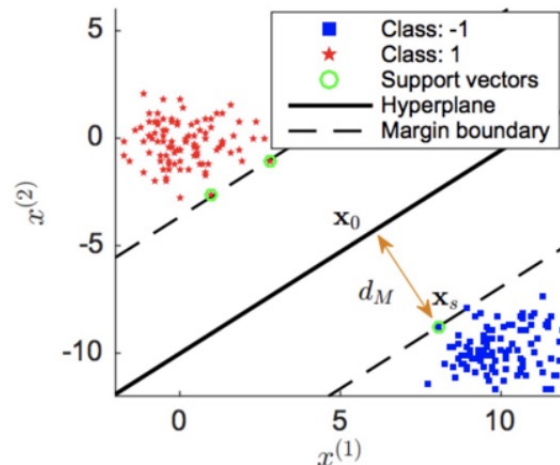


Figure 5: SVM

For linear separable data points, SVM is going to find a hyperplane, defined as:

$$w^T x + b = 0 \quad (1)$$

that maximize the margin, define as:

$$\frac{|w^T x_n + b|}{\|w\|_2} \quad (2)$$

By applying normalization that $\min_i |w^T x_i + b| \equiv 1$ the problem becomes:

$$\min_{w,b} \|w\|^2 \quad (3)$$

subject to

$$y_i(w^T x_i + b) \geq 1 \quad (4)$$

This convex problem can be solved by its dual problem, and the answer is:

$$w^* = \sum_{w \in \mathcal{S}V} \alpha_i^* y_i x_i \quad (5)$$

$$b^* = -\frac{1}{2} \sum_{w \in \mathcal{S}V} \alpha_i^* y_i (x_i^T x^+ + x_i^T x^-) \quad (6)$$

For prediction, the output will be:

$$f(x) = \text{sgn}[\sum_{w \in \mathcal{S}V} \alpha_i^* y_i x_i^T + b^*] \quad (7)$$

If the training set is not linear separable, the slack variable $\zeta \geq 0$ is introduced to allow misclassification happen. Then the optimization problem is:

$$\text{argmin}_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \zeta_n \quad (8)$$

subject to

$$s_n y_n \geq 1 - \zeta_n, n = 1, \dots, N \quad (9)$$

The value of C controls the tradeoff between max margin and misclassification.

Simple CNN Network

1) Kernel Neural Network Model

A neural network[3] is put together by hooking together many of the simple neurons, so that the output of a neuron can be the input of another. Taking the figure below as an example, we will explain here the framework of Neural Network. Our neural network has parameters:

$$(W, b) = (W^1, W^2, b^1, b^2) \quad (10)$$

Where $W_{ij}^{(l)}$ denotes the parameter associated with the connection between unit j in layer l, and unit i in layer l+1. Also, $b_i^{(l)}$ is the bias associated with unit i in layer l + 1. We will write a_i^l to denote the activation of unit i in layer

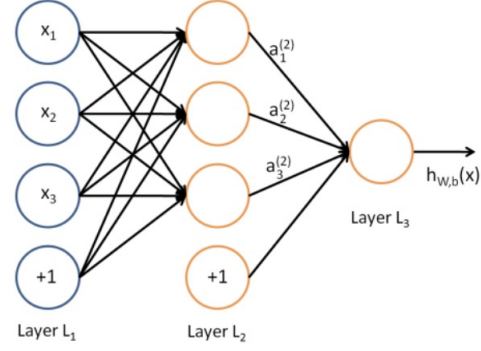


Figure 6: A single layer of CNN

j. Given a fixed setting of the parameters W, b , our neural network defines a hypothesis $h_{W,b}(x)$ that outputs a real number. Specifically, the computation that this neural network represents is given by:

$$a_1^2 = f(W_11^{(l)}x_1 + W_12^{(l)}x_2 + W_13^{(l)}x_3 + b_1^{(l)})$$

$$a_2^2 = f(W_21^{(l)}x_1 + W_22^{(l)}x_2 + W_23^{(l)}x_3 + b_2^{(l)})$$

$$a_3^2 = f(W_31^{(l)}x_1 + W_32^{(l)}x_2 + W_33^{(l)}x_3 + b_3^{(l)})$$

$$h_{W,b}(x) = f(W_11^{(2)}a_1^{(2)} + W_12^{(2)}a_2^{(2)} + W_13^{(2)}a_3^{(2)} + b_1^{(2)}) \quad (11)$$

2) Backpropagation Algorithm

We can train our neural network using batch gradient descent. In detail, for a single training example (x, y) , we define the cost function with respect to that single example to be:

$$J(W, b; x, y) = \frac{1}{2} \|h_{w,b}(x) - y\|^2 \quad (12)$$

We can then define the overall cost function to be:

$$J(W, b) = \left[\frac{1}{m} \sum_{i=1}^m J(W, b; x^{(i)}, y^{(i)}) \right] + \frac{\lambda}{2} \sum_{l=1}^{m-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (W_{ji}^{(l)})^2 \quad (13)$$

The first term in the definition of $J(W, b)$ is an average sum-of-squares error term. The second term is a regularization term that tends to decrease the magnitude of the weights and helps prevent overfitting. Then the backpropagation algorithm is give below. First perform a feedforward pass, computing the activations for layers L_2, L_3 , and so on up to the output layer L_n . For each output layer unit i in layer n , set:

$$y_i^{n_i} = \frac{\partial}{\partial z_i^{(n_i)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 \quad (14)$$

For each node i in layer l set

$$y_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} s_{l+1} W_{ji}^{(l)} \delta_j^{(l+1)} \right) (f' z_i^{(l)}) \quad (15)$$

Compute the desired partial derivatives, which are given as:

$$\begin{aligned} \frac{\partial}{\partial W_{ij}} J(W, b; x, y) &= a_j^{(l)} \delta_i^{(l+1)} \\ \frac{\partial}{\partial b_i} J(W, b; x, y) &= \delta_i^{(l+1)} \end{aligned} \tag{16}$$

To train our neural network, we can now repeatedly take steps of gradient descent to reduce our cost function $J(W, b)$.

CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output. The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply on CNNs.

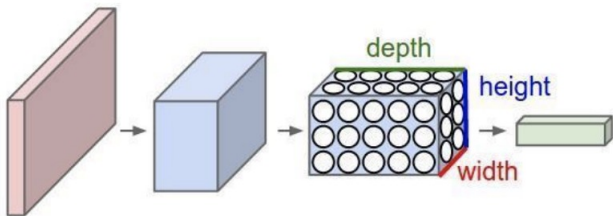


Figure 7: The process of CNN

3) Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is a commonly used optimization algorithm. It straightforwardly adopted the purpose of minimizing an objective function that has a form of sums. A random batch is selected for each iteration, namely stochastically, under which the objective function will be minimized through each iteration.

$$w_{new} = w - \eta \nabla Q(W) \tag{17}$$

ResNet It[10] is made up of small residual blocks shown in the following figure

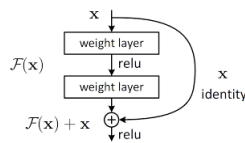


Figure 8: Construction of the ResNet

The purpose for the direct connection with the input to the output is to give a direct path for the error to back propagate through the entire network.

Simple KNN K-NN[5] is a non-parametric lazy learning algorithm, which means that it does not make any assumptions on the underlying data distribution, also, it does not use the training data points to do any generalization. At its most basic level, it is essentially classification by finding the most similar data points in the training data, and making an educated guess based on their classifications. Given data with N unique features, the feature vector would be a vector of length N, where entry I of the vector represents that data points value for feature I. Each feature vector can thus be thought of as a point in R^N . Once we have formed our training data-set, which is represented as an $M \times N$ matrix where M is the number of data points and N is the number of features, we can now begin classifying. For each classification query, the gist of k-NN is to:

- Compute a distance value between the item to be classified and every item in the training data-set
- The k closest data points (the items with the k lowest distances)
- Conduct a majority vote among those data points the dominating classification in that pool is decided as the final classification

After doing all of the above and deciding on a metric, the result of the k-NN algorithm is a decision boundary that partitions R^N into sections. Each section represents a class in the classification problem. The boundaries need not be formed with actual training examples, instead, they are calculated using the distance metric and the available training points.

Siamese Network Its[1] a merged network with two or more subnetworks. Those networks are identical and they share the weights, just like siamese twins. By extracting features, the connected part compared the L-2 distance to classify the images.

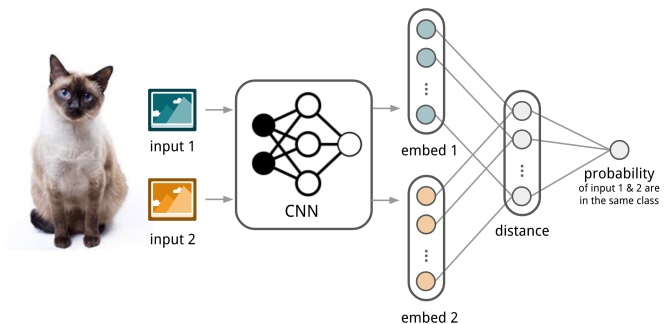


Figure 9: Construction of the Siamese Network

5. Experiments

The most tricky part of our topic is huge-amount-class classification. For our training set, we have over twenty thousand images and there are more than four thousand classes. To deal with this problem, traditional machine learning methods might not be suitable. Therefore, the first method we came up was the CNN, which in most cases shows a better performance when dealing with a complex image problem since convolutional neural network has a wonderful performance on extracting image features.

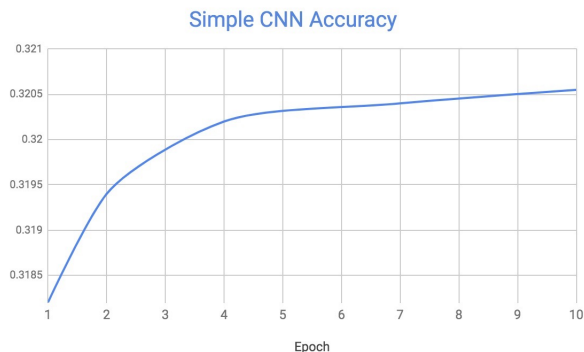


Figure 10: The accuracy of the simple CNN Network

5.1. Baseline

After doing some data augmentations, we tried Convolutional Neural Network with 2 convolutional layers as our baseline. Figure 10 and shows the the changes of accuracy. The accuracy function keeps increasing as running more epochs, level off to 32 %. We thought the reason we had a bad accuracy on CNN was that the depth of the network is too shallow and the huge-amount classes makes it really hard to make an accurate classification.

5.2. ResNet18 & ResNet101

The above results lead us to try some deeper network structures: ResNet18 and ResNet101. Figure 11 shows the result of the ResNet18 and ResNet101. It can be easily seen that we had some improvement on the performance by using these two models, but it looks still not ideal.

5.3. Siamese-Network

To get a better result, we also tried an Ensemble Network Model Inspired by Siamese Network.

The Siamese Network is a merged network with two or more sub-networks, which are identical and share the same weights, just like siamese twins in Figure 12. By extracting features, the connected part compared the L-2 distance to classify the images. The reason we used a similar ensemble

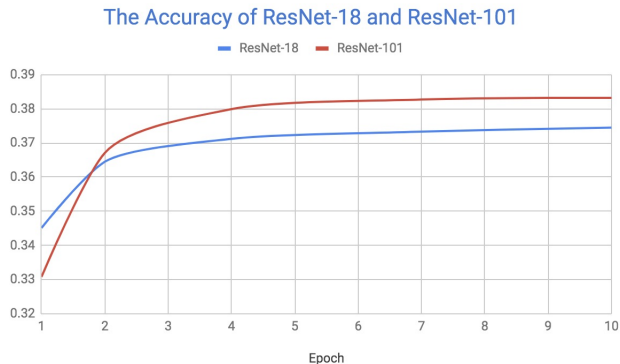


Figure 11: The accuracy of the ResNet-18 ResNet-101



Figure 12: Siamese Twins

network instead of using Siamese Network directly was that the Siamese Network will be too large and too slow to train and classify. Inspired by the Siamese Network, we tried the ensemble of two simple CNN models. From Figure 13, we can see that after running ten epochs, the accuracy can reach 47.64%, which is a small improvement from the previous models.

5.4. Conclusion on Neural Network Based Models

By trying the previous four models, simple CNN, ResNet18, ResNet101 and ensemble of two simple CNN models, we found that it was still hard to get an ideal accuracy by only making the network structure deeper. Therefore, we came up two directions: 1. Trying some new models other than neural network based models. 2. Changing the feature extracting method to improve the usefulness of the extracted features.

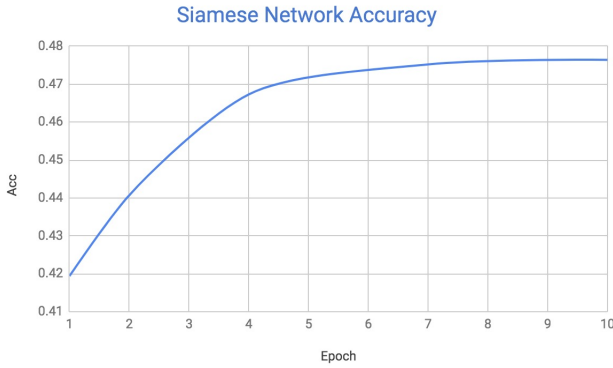


Figure 13: The accuracy of the Siamese Network

Table 1: The accuracy of all the models

Models	Accuracy
Simple CNN	0.3237
ResNet-18	0.3745
ResNet-101	0.3832
Siamese Network	0.4764
SVM	0.4113
Simple KNN	0.4825

5.5. Classification Models

We tried two classification methods: SVM and KNN. Table 1 shows the accuracy of all these models. The accuracy of simple KNN performs better than the Siamese Network and even SVM is better than the complicated ResNet-101. We thought the possible reason might be that we are trying to classify the different species of whales and should be assigned to the models focused on classification, so these two models gave us a higher accuracy other than the neural network based models although they usually have a good performance on dealing with a complex problem. Finishing this possible direction, we moved to the second one: improve feature extracting part.

5.6. New Feature Extracting Method: HOG

After finding more suitable models, we moved to try some new feature extracting methods to see if we can improve our accuracy. The new method we tried is HOG due to the fact that the SVM combined with HOG always shows better performance when facing classification problems. Then we used the new features extracted by using HOG on the two classification models, SVM and KNN. From Table 2, it can be easily seen that we had a remarkable improvement by using HOG feature extracting method.

Table 2: The accuracy of all the models(After HOG)

Models	Accuracy
SVM	0.5727
Simple KNN	0.6482

6. Conclusion

We utilized different machine learning models to classify different species of whales. The feature extraction methods we tried are CNN and HOG. The models we tried include: Simple CNN, ResNet18, ResNet101, Ensemble network of Simple CNN, SVM and K-NN. Our experimental result shows that HOG + KNN has the best performance, which has the accuracy around 65%. For the future work, if we had more time, we might want to try some other models such as DenseNet + GBDT. Also we want to try some new data pre-processing tools such as Class Merge and Bounding Box.

7. Contributions

At the beginning of this quarter, we met a couple of times to pick up the topics and also discussed the model designs. Tianran Zhang and Yuekuan Luo contributed on downloading the data from Kaggle by using Kaggle API and pre-processing the data, i.e. data split, data augmentation, etc. Ruoqi Zhang and Kai Wang contributed on implementing all the models. After we trained all the models Tianran Zhang and Yuekuan Luo contributed on comparing the results and doing the analyzation. In the end, we worked on the paper together and each person contributed on some parts.

Acknowledgement

Here, we thank for the great work done by our teammates. And we appreciate the department supporting us with the helpful GPU clusters.

References

- [1] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016. 5
- [2] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006. 1
- [3] J. Bouvrie. Notes on convolutional neural networks. 2006. 4
- [4] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 3
- [5] T. M. Cover, P. E. Hart, et al. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967. 5
- [6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *international Conference on computer*

- vision & Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE Computer Society, 2005. [3](#)
- [7] B. Demir and S. Erturk. Hyperspectral image classification using relevance vector machines. *IEEE Geoscience and Remote Sensing Letters*, 4(4):586–590, 2007. [2](#)
- [8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009. [2](#)
- [9] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014. [2](#)
- [10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. [5](#)
- [11] D. Hsu. Using convolutional neural networks to classify dog breeds. *CS231n: Convolutional Neural Networks for Visual Recognition [course webpage]*, 2015. [2](#)
- [12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. [2](#)
- [13] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur. Dog breed classification using part localization. In *European conference on computer vision*, pages 172–185. Springer, 2012. [2](#)
- [14] K. Makantasis, K. Karantzas, A. Doulamis, and N. Doulamis. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4959–4962. IEEE, 2015. [2](#)
- [15] X. Mao, C. Shen, and Y.-B. Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems*, pages 2802–2810, 2016. [2](#)
- [16] G. Mercier and M. Lennon. Support vector machines for hyperspectral image classification with spectral-based kernels. In *IGARSS 2003. 2003 IEEE International Geoscience and Remote Sensing Symposium. Proceedings (IEEE Cat. No. 03CH37477)*, volume 1, pages 288–290. IEEE, 2003. [2](#)
- [17] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. [2](#)
- [18] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012. [2](#)