# The Marinexplore and Cornell University Whale Detection Challenge

Group 12

Chi-Hsin Lo
A53311981
c2lo@eng.ucsd.edu

Shang-Yun Yeh
A53311603
shyeh@eng.ucsd.edu

Kalvin Goode
A12620672
KGoode@eng.ucsd.edu

## I. Introduction

Natural resources are undoubtedly vital to every living on earth. With the great care of human activity, we can maintain a sustainable ecology both on land and in the ocean. Our sound dataset comes from The Marinexplore and Cornell University, recorded in North Atlantic provided by the shipping industry [1]. When shipping goods throughout different countries, there is a possibility that ships may intervene in whale activity areas and disrupt their livings. In response, most of the ships had equipped auto-detection buoys to capture the right whale's characteristic up-calls and other marine activity. The goal is to recognize whale calls in these acoustic recordings by deep learning frameworks. Ultimately implement into navigation technology to alert and to propose alternative routes that protects marine lives.

In our work, we first tried light weight CNN models : GoogleNet, MobileNet and MNASNet for classifications. As a variation and seeking for improvement, we replace the classification layers to SVM and perform PCA dimension reduction before SVM. Achieving desirable results on baseline models, we asked ourselves if this can be done with smaller models. So we further implemented two smaller convolutional neural network models, both achieved ideal results. Lastly, we extended the 15-layer model into autoencoder and a classificatier, and achieved 91.33% accuracy as our final model.

## II. Related Work

In June 2017, Lonce Wyse used fast fourier transform to convert audio clips to spectrograms, and used convolutional neural networks for classification is his work Audio spectrogram representations for processing with Convolutional Neural Networks [10]. In October 2018, the article Acoustic Detection of Humpback Whales Using a Convolutional Neural Network [11] on Google AI Blog, Matt Harvey used per-channel energy normalization (PCEN) on their spectrogram data. In September 2011, Antoni et al. proposed Non-Local Means Denoising for image denoising.

Dr. Tang from University of Toronto has shown that using linear SVM in combination with a deep convolutional network gives a significant accuracy on many recognition tasks including MNIST and CIFAR-10. For any trained convolution networks, Tang replaced the last layer of the network with SVM for final classification, so the hidden variable data produce from the network is treated as an input of SVM. In Facial Expression Recognition Challenge hosted by ICML 2013 workshop, Tang implemented this technique and has recognized as a winning solution in the contest. It achieved an accuracy around 70%, which is 2% higher than 2nd place.

In 2017, Sangwook from Korea University proposed a new way to classify sounds from different environments. In the preprocessing step, He separated two channels of sound data and did covariance matrix of spectrums (COV) and Double FFT Image (DFI) to these two sound images respectly. Afterwards, he stacked those four spectrum images and put them into a 15-layer CNN model along with batch-normalization, Relu, and fully-connected layer and achieved average 83.6% accuracy. Inspired by this paper, we adapted this model to our data and also achieved around 86% accuracy.

In January 2020, Kirsebom and Frazao from Institute for Big Data Analytics used Resnet to detect and to handle diverse acoustic information on whale's upcalls. This deep learning infrastructure has 8 blocks composed of a convolution, batch normalization and ReLu, and at the end, it has Softmax for classification to show the probability on these two classes. As a result, they claim to achieve around 80% for recalls and 90% for precision. The data was recorded in Gulf of St. Lawrence and in Gulf of Maine in North Atlantic.

## III. Dataset and Features

The dataset contains 30000 training data with labels and 54503 testing data without labels, each being a 2-second long acoustic track with a sample rate of 2000Hz. It labels 1 for having whale's up-calls and 0 otherwise. However, since we lack testing labels we have to discard those testing samples for examining accuracy. Also, due to limited computing and storage resource we only use 3000 images for training, and 3000 images for validation.

## IV. Methods

### A. Spectrograms

Spectrogram is often used to represent audio data. Data is first split into segments, each of same length and the spectrum of each segment is computed. The spectrum of each segment is computed by discrete fourier transform (DFT). Suppose the segmented data being $X$, its discrete fourier transform $Y$ is computed by

$$Y(k) = \sum_{j=1}^{n} X(j) W_n^{(j-1)(k-1)}$$
$$W_n = \exp(\frac{-2\pi i}{n})$$

In many packages including the one we used, the algorithm implementing DFT is called fast fourier transform (FFT). It has time complexity $O(n\log(n))$ instead of $O(n^2)$ time complexity of direct DFT, when computing on a data $X$ of length $n$.

### *B. Histogram equalization*

Histogram equalization is an algorithm for adjusting image pixel values to enhance contrast. The concept it to compute the pixel intensity histogram, and stretch the histogram along the intensity axis to enhance contrast. Consider an image $f$ with size $r \times c$ with pixel intensities ranging from $0$ to $L-1$ ($L = 256$ can be used for general cases). The normalized histogram of $f$ denoted $p$ is computed as

$$p_n = \frac{number\ of\ pixels\ with\ intensity\ n}{total\ number\ of\ pixels}, n = 0, 1, ..., L-1.$$

The histogram equalized image g will be defined by

$$g_{i,j} = \lfloor (L-1) \sum_{n=0}^{f_{i,j}} p_n \rfloor$$

Which is equivalent to stretching the pixel intensities bin.

$$T(k) = \lfloor (L-1) \sum_{n=0}^{k} p_n \rfloor$$

### *C. Per-channel Energy Normalization*

Per-channel Energy Normalization, also known as PCEN, suppresses background noise, emphasizes foreground signal, and has been shown in the researches that it out-performs the other pointwise operation and can preserve the locality structure of harmonic patterns along the mel-frequency axis. PCEN normalizes a time-frequency representation $S$ by performing automatic gain control within a time-window, followed by the nonlinear compression as below. Define $G$ as Gain Control, $B$ as Bias, $P$ as Power, and $\epsilon$ which is a small constant used to ensure numerical stability of the filter.

$$P[f,t] = (\frac{S}{(\epsilon + M[f,t])^G} + B)^P - B^P$$

The matrix $M$ is the result of applying a low-pass, temporal IIR filter to S:

$$M[f,t] = (1-b) \times M[f,t-1] + b \times S[f,t]$$

If b is not provided, it is calculated as:

$$b = \frac{(sqrt(1+4\times T^2)-1)}{(2\times T^2)}$$
$$T = \frac{time\ constant \times simple\ rate}{hop\ length}$$

### *D. Non Local Means Denoising*

Non-local Means Denoising [12] replaces the color of a pixel with an weighted average of neighbor pixels, with weight based on the pixel color similarities and standard deviation of noise. In this algorithm, a search window is used for pixel-wise computation. When computing, it only consider pixels within the search window region to do weighted average. The denoised pixel of a color image $u = (u_1, u_2, u_3)$ at a certain pixel p is computed by

$$\hat{u_i}(p) = \frac{1}{C(p)} \sum_{q \in B(p,r)} u_i(q) w(p,q)$$
$$C(p) = \sum_{q \in B(p,r)} w(p,q)$$

Where $i = 1, 2, 3$ and $B(p,r)$ indicates a size $(2r+1)\times(2r+1)$ size neighborhood centered at $p$. And the weight in the above formulas is computed using an exponential kernel

$$w(p,q) = \exp(\frac{-max(d^2-2\sigma^2)}{h^2})$$
$$d^2 = d^2(B(p,r), B(q,r))$$
$$= \frac{1}{3(2r+1)} \sum_{i=1}^{3} \sum_{j \in B(0,r)} (u_i(p+j) - u_i(q+j))^2$$

Where $\sigma$ denotes the standard deviation of the noise and $h$ is the filter parameter set depending on the value of $\sigma$.

### *E. Convolutional Neural Network*

Convolution layer is using a fix-sized kernel, which has the same depth as the input image and has different weighted pixels, to slide through the whole image, and at each position, it computes the dot-product with the corresponding pixels. Since the whole process is doing cross-correlation, doing convolution could extract the corresponding features which will be very useful when the model has learned some.

$$f[x,y] * g[x,y] = \Sigma_{j=0}^{\infty} \Sigma_{i=0}^{\infty} f[i,j] * g[i-x, j-y]$$

We usually normalize the input data to facilitate learning. However, as the model grows bigger and bigger, for the further layers, some input data might be so large that a slight change on the weight will make huge differences. Thus, we introduce batch-normalization into our model. Batch-normalization mean standard deviation normalizes each pixel with the batch layers on the same position and by doing so the weight could have more spaces to adjust its value, making the training faster. Use $B$ to denote a mini-batch of size m of the entire training set. Given $B$ the size of mini-batch and $m$ of the entire training set.

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i$$
$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2$$
$$\bar{x}_i = \frac{x_i - \mu_i}{\sqrt{\sigma_B}}$$

To eliminate overfitting, dropout is one of the popular choices. Dropout randomly zero some weights and therefore, the model is then forced to use the rest of the weights to learn features and couldn't easily fit the training data, which just like training multiple models and then combine all of them to make the prediction.

### *F. Support Vector Machine (SVM)*

SVM is a fast and efficient algorithm for classifying small labeled dataset. Therefore, it perfectly suits our need for determining if the whale sound exists or not. SVM maps data into higher dimension space which is easier to find the best hyper-plane that is the furthest to the both types of data linearly or non-linearly. We are using Linear-svc for our project because it is faster to converge, when the dataset is large, than the other methods. The kernel function for this method is specified to be a linear kernel, which does not involve inner-product.

$$k(x_i, x_j) = < x_i,\ x_j >$$

The loss function used here is the Hinge Loss which is famous for maximum-margin classification. Given $t = \pm 1$ as the ground truth and $y$ the output of our model, the formula is the below. The Loss function will only be zero if t and y is the same sign, which means that our predictions are correct.

$$\ell(y) = \max(0, 1 - t \cdot y)$$

Given the training vectors $x_i$, i=1,..., n, in two classes, and a vector $y_i$ as the answer, our goal is to find $w$ and $b$ such that sign$(w^T \phi(x) + b)$ is correct for the most samples, and the following is the primal problem we are solving,

$$\min_{w,b} \tfrac{1}{2} w^T w + C \textstyle\sum_{i=1} \max(0, y_i(w^T \phi(x_i) + b))$$

### G. Autoencoder

Autoencoder is an unsupervised learning model using machine learning to efficiently encode its training data. An autoencoder consists of two parts, encoder and decoder. The encoder reduces dimensions and tries to preserve the important features among the training data and the decoder is tries to reconstruct the encoded data to the original data. The way it determines if the input data is different from the training data is by calculating how different the output and original data is. If the difference for output data and original data is out of a specific range, we couldn't compress that input, then we will consider that it is not the same category of the input data. Since we address audio clips as spectrograms, so we use convolutional autoencoders as our backbone. We connect the encoded code to four linear layers to classify as final output. And train the entire structure with loss

$$loss = class\_loss + r * recover\_loss$$
$$class\_loss = CrossEntropyLoss(outputs\_class, labels)$$
$$recover\_loss = MSELoss(recover\_outputs, inputs)$$

We set r=0.05.

## V. EXPERIMENTS/RESULTS/DISCUSSION

### A. Base Models

We first tried three state of the art models - GoogleNet, MobileNet v2, and MNASNet 1.0. These three models are all light weight models that perform well on the ImageNet Challenge, with top1/top5 accuracy being 30.22/10.47, 28.12/9.71 and 26.49/8.456 respectively. We used these three baseline models to perform classification. The results are included in TABLE 1.

### B. Base Models + SVM

With PCA and SVM added at the end of layer in each model, we computed accuracy with various PCA components. In particular, the last output layer before fully connected softmax layer for googlenet, mobilenet and mnasnet have output of 1024, 1280 and 1024 features, respectively. By replacing softmax function, we performed PCA from these features down to various dimension from 8, 16 to 512 with 16 dimensions increment. Lastly, we added linear SVM to classify each spectrogram with output of 0 or 1. The result of accuracy with selected amount of dimension is shown in the table. For googlenet, the average accuracy is 74.94% and the best accuracy reached 78.82% with 16 dimensions. For mobilenet, the average accuracy is 72.9% and the best accuracy reached 80.6% with 8 dimensions. For mnasnet, the average accuracy is 77.54% and the best accuracy reached 86% with 48 dimensions. Generally, mobilnet performed the best in recognizing whale-up calls and mnasnet has the worst performance overall, while googlenet stays almost stable with various dimension. The error curves on having SVM general behave as expected. Error increases as dimension increases, since it is difficult to have good estimates in these features that possibly have noises interfering the model parameters. However, we were also expected SVM can improve the model by having lower error compare with original models.

TABLE I
BASELINE SVM RESULT

| Test Accuracy (%) | PCA Dimension Reduction and SVM | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Vanilla** | **8** | **16** | **32** | **48** | **96** | **192** |
| Google | 0.8363 | 74.8 | **78.8***| 78.1 | 78.0 | 73.7 | 75.8 |
| Mobilenet v2 | 0.8523 | **80.6*** | 73.4 | 75.4 | 76.5 | 73.1 | 71.3 |
| Mnasnet 1.0 | 0.8477 | 80.8 | 77.1 | 78.0 | **86.0*** | 77.6 | 79.6 |

*Highest in the model

### C. Small Models

Inspired by the papers, we also wanted to know how it works for even smaller CNN models, 2-layer and 15-layer. In order to speed up the training while not increase too much computation complexity, we configured our model of batch-size of 32 and 0.001 as our learning rate, which is big enough to speed up the training and small enough to find the local minimum.

For the 2-layer models, we used two convolution layers with batch-norm, ReLU, and dropout layer followed by a fully connected layer passing through a softmax layer to compute the output. The optimizer here we choose to use is Stochastic gradient descent (SGD). It calculates the contribution of each node to the error and adjusts the weight accordingly. Assuming $\eta$ is the learning rate, $w$ is the weigth, and $Q(w)$ is the loss function for the current weight. The optimization algorithm becomes the following,

$$w = w - \eta \times \nabla Q(w)$$

To solve one of the common problems for sound classification, overfitting, the original structure used dropout layers to solve that. However, adding the dropout layers only got extra 0.5% of accuracy to the model, which meant that the model is too small for the dataset so that even with dropout, the rest of the model could still learn to overfit it. Thus, it prompted us to try on a bigger model, a 15-layer one.

The 15-layer model was also consisted of 9 convolution layers, 3 pooling layers, all of them were equipped with Batch-normalization and ReLU, and 3 fully connected layers.

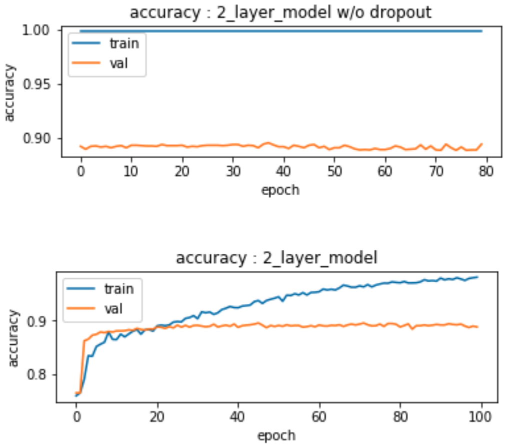

Fig. 1. The accuracy of 2_layer model with and without dropout

Even so, the overfitting was still severe. Therefore, we also implemented L2 regulation on the data in order to solve this issue. However, after our testing, we found out that, although without dropout, the model quickly overfitted and the accuracy saturated, the accuracy was still quite high and almost as high as the one with the dropout layer. Therefore, we speculated that we has reached the maximum for this model because when both models start overfitting, the results were the same.

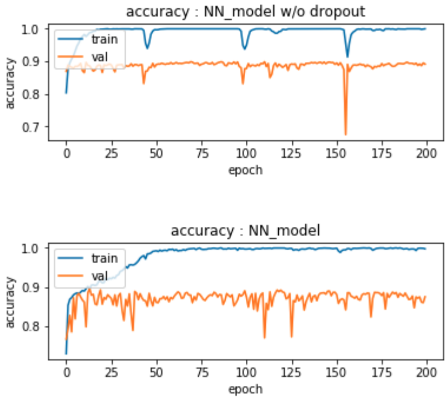

Fig. 2. The accuracy of 15_layer model with and without dropout

### D. Autoencoder

We trained our autoencoder framework of the 15 layer model, with all parameters and optimizer the same as described in section C and $r = 0.05$. We reached a slightly better accuracy than only the CNN framework. This matches our anticipation that with the autoencoder framework and combined loss of both recover loss and classification loss, the encoded code better represents the whole image and thus performance should be better. We should also note that changing the parameter r, that is, the ratio between the classification loss and recover loss will yield a big difference. But we omit the detailed discussion due to limited space.

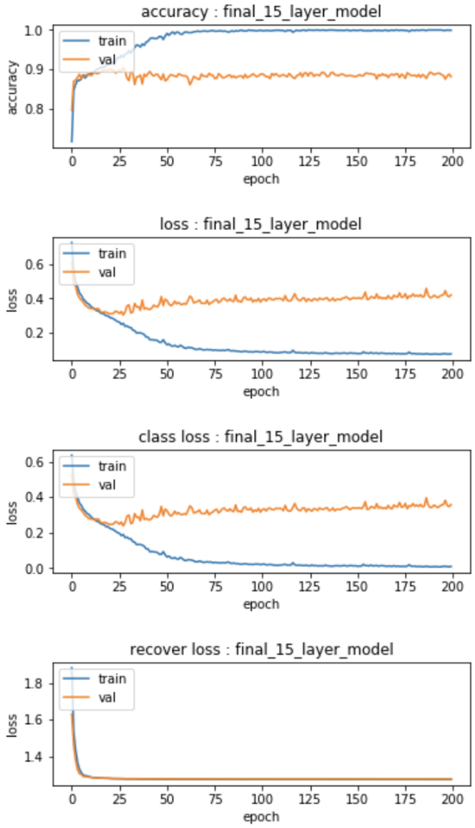

Fig. 3. The accuracy and loss of final_15_layer model

TABLE II
SMALL MODEL AND AUTOENCODER RESULTS

| Test Accuracy | **Accuracy** |
|---|---|
| 2_layer_model w/o dropout | 0.89 |
| 2_layer_model | 0.8957 |
| 15_layer_model w/o dropout | 0.895 |
| 15_layer_model | 0.8953 |
| final_15_layer_model | 0.9133 |

## VI. CONCLUSION AND FUTURE WORK

In this project, we implemented three deep learning frameworks to identify the existence of whale up-calls in the 2-second audio tracks. For preprocessing, we first used FFT to convert audio clips into spectrograms. Then, we enhance the contrast by equalizing the spectrogram histograms. Third, we used Non-Local Means Denoising to denoise the images, and then used PCEN to normalizes time-frequency amplitudes. The models we used include MobileNet v2, GoogleNet, MNASNet 1.0, their variations by replacing classification layers to SVMs, 2-layer CNN model, 15-layer CNN model and their autoencoder version. The experimental result shows that, our final version of 15-version autoencoder model reaches highest accuracy. Even though state of the art models work well on our dataset too, but simpler models works even better, we think this is because of the simplicity of our dataset, and we used pretrained weight for larger models.

## CONTRIBUTIONS

All members in this group has significance importance in the completion of this project. Chi-Hsin Lo proposed various kinds of machine learning techniques, developed the structure of the code and scheduled a complete plan for members to follow and controls code and writing qualities. Shang-Yun Yeh had collaboratively discussed and debated during model selections with others and preprocessed the dataset for others to use, also his proactiveness and attentiveness on improving the smaller models really smoothen the whole process. Finally, Kalvin Goode had provided numerous research paper related to this project and seek and tested many possible improvement in these models with efficiency. With this possibly his first deep learning project, his rapid learning curve is also highly commendable.

## REFERENCES

[1] Kaggle competition link : https://www.kaggle.com/c/whale-detection-challenge/overview

[2] Oliver S. Kirsebom, Fabio Frazao, Yvan Simard, Nathalie Roy, Stan Matwin, Samuel Giard. Performance of a Deep Neural Network at Detecting North Atlantic Right Whale Upcalls. arXiv preprint arXiv:2001.09127, 2020. https://arxiv.org/abs/2001.09127

[3] Vincent Lostanlen, Justin Salamon, Mark Cartwright, Brian McFee, Andrew Farnsworth, Steve Kelling, and Juan Pablo Bello. Per-Channel Energy Normalization: Why and How.

[4] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, Quoc V. Le. MnasNet: Platform-Aware Neural Architecture Search for Mobile. https://arxiv.org/abs/1807.11626

[5] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-Local Means Denoising, Image Processing On Line, 1 (2011), pp. 208–212.

[6] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. https://arxiv.org/abs/1801.04381

[7] Yichuan Tang. Deep learning using linear support vector machines. Workshop on Challenges in Representation Learning, ICML, 2013. https://arxiv.org/abs/1306.0239

[8] Sangwook Park, Seongkyu Mun, Younglo Lee, Hanseok Ko. ACOUSTIC SCENE CLASSIFICATION BASED ON CONVOLUTIONAL NEURAL NETWORK USING DOUBLE IMAGE FEATURES, 2017. http://dcase.community/documents/workshop2017/proceedings/DCASE 2017Workshop_Park_214.pdf

[9] Annamaria Mesaros, Toni Heittola, Tuomas Virtanen. A MULTI-DEVICE DATASET FOR URBAN ACOUSTIC SCENE CLASSIFICATION, 2018. https://arxiv.org/pdf/1807.09840.pdf?fbclid=IwAR2JVWohi4U2MKGcd H3OutQwIjfzp8-ih_kFeGf6jVNd3K5wYHGt2pY4_Kg

[10] Lonce Wyse. Audio spectrogram representations for processing with Convolutional Neural Networks, 2017. https://arxiv.org/pdf/1706.09559.pdf

[11] Matt Harvey. Acoustic Detection of Humpback Whales Using a Convolutional Neural Network, 2018. https://ai.googleblog.com/2018/10/acoustic-detection-of-humpback-whales.html

[12] Antoni Buades, Bartomeu Coll, Jean-Michel Morel. Non-Local Means Denoising, 2011. http://www.ipol.im/pub/art/2011/bcm$_nlm/$

## REPLY TO CRITICAL REVIEWS

### A. Group 13

Very glad to watch your presentation. It seems you are all doing a very good job in this project, and Whale Detection is also a very interesting and informative topic when wildlife conversion and animal behavior detection is popular in the world. This presentation is a good example that machine learning can be used in audio classification, specifically used in whale sound detection. You group extract spectrogram as main feature and use some python package to handle it (cv2, torchversion, etc. ) . what's more, after introducing how to extract the main feature, you explain three models that you used to train the audio data: GoogleNet, MobileNet, MNASNet. And constitute them with different classifier (SVM, autoencoder, Vanilla data) , and compare their efficiency and accuracy. The result seems good, and the best accuracy reach over the 90Congratulations ! But there are also some suggestions I can give you to improve your project and make it more convincing. - In this project, I don't really understand the dataset constitution and do you use the cross validation or not? Maybe you can explain more clearly about the dataset you used in your presentation and give us link of it. - What's more, you refer the best accuracy (more than 90as strong evidence to prove this model is useful and reliable. That's because maybe this dataset has relation with the training dataset, or just because you choose lucky and suitable pretrained parameters. I suggest you can separate the total dataset into 10 parts, and use 9 as training set and 1 as test set. Then iterate all dataset, cross validate those and then average the results. - I hear about your group code review, and in general, it is very clear and good code review. But it pays too much attention on the result analyzation which has been talked in your ppt representation. I wish you could focus more on the code itself. Why you write this part of codes. What's the function of it? Why you choose this layer or this optimizer? how do you extract the feature and handle it? is there any problems in the process of your training? and how do you solve them and so on. So we can get more useful knowledge about the machine learning coding. - And I think the code review should be made by every member of group like presentation. But I only hear one people sound which little confused me. If other member do some jobs in this project, maybe you should also put some comments on the code review. These are all my suggestions, and I wish you could have great success on your future jobs. Thank you, have a good day.

Our response: Thank you for your suggestions. First, to clarify, we had divided our presentation into each group member, it had one female voice and two distinct male voices. We paid more attention to results than to coding because we believe people from this class have a variety of backgrounds and would be much boring if emphasis too much on coding because not many people have computer science backgrounds. From the questions in the third bullet point, we have included detailed answers in the report, apologies for not include them in the presentation and future presentation.

### B. Group 24

This presentation focuses on Whale detection, which is a really interesting topic. Firstly, it's the first time I know what a spectrogram is, you guys introduce it really clearly. And I think it's a good choice to solve this problem because it can show the frequency feature of different time windows. It's a wise choice to utilize these best CNN-model, GoogleNet, MobileNEt and

MNASNet since your feature is a two-dimension image. The performance and the efficiency of these models are the best in CNN networks. After the CNN part, you guys use autoencoder. This is a new kind of model we learned from this class, it's good to try it in this problem and it should work well. The architecture of your model and the result of your experiment is clear in your presentation, the utilities of figures and tables are good and clever. Something improvement: I think it's good to just focus on the FFT of the audio signal first, maybe there has been enough information in it and you don't have to deal with the CNN signal. For me, this is the first thing that came into my mind when I saw this problem and I think your group should at least implement it and analyze why the performance of this idea is not good. It's better to state why you guys choose to transfer one-dimension data to 2-dimension data and state the benefit of this step. I think there's too much on the slides, you can just give us some points of your project and show the other details in the presentation. All in all, this is an interesting and a really good project.

Our response: We really appreciate such detailed feedback. It helps us understand some technical terminologies others may have not exposed to in this field, and we had made clarity on those in the report. After some adjustments from the suggestions above, we decided to dig in more on autoencoder and show some improvements in the reports.

### *C. Group 40*

Group 12 research on the classification of detection of whale based on audio. The data is from 3000 out of 30000 training data and have 2 labels. In feature extraction part, they use spectrogram as main feature and provide some package used. After use models (GoogleNet, MobileNet, MnasNet) to complete scaling, they use different classifier (SVM, Vanilla, autoencoder) and compare the result. The accuracy reaches 90 percent, which is quite good.

I am glad to watch this video about whale detection based on audio detection. I think this is a very interesting topic and a very good example of audio detection application. Feature extraction part is very clear, including the meaning of feature and the package to implement it. I think it is quite cool to use models to scale. The presentation of model explanation is clear, and the figure is vivid.

Thank you for your compliments. We did spend a lot of time doing this project.

Improvement Total video is over 30 minutes long, I think it's quite long compared to requirement. Since we have done so many experiments, it was so hard to squeeze all the information within 20 mins, we appologize for that. In the dataset introduction, I am a little confused about the data used. I think its better to describe training and test data clearer. For the dataset, we divided the original labeled data into training data and testing data because it was the only way to evaluate out model. In feature extraction part, I think I am confused about why you guys want to extract features in this way, rather than how to do in the slide. The way we extracted features was the same as our slide. I am a bit confused about this question. Some figures of the result are very small, I think it may work well to highlight some important point in the figures on the slide. We will improve that in our future presentation. Thank you! Code review duplicate lots of previous slide information, which I think can be reduce. I think your group did a good job. We appreciate that! Thank you.