Humpback Whale Identification

Xingyu Jin
s1836694

Siqing Geng
s1841522

Jiwei Yu
s1833769

Siwei Wang
s1817788

Email: {X.Jin-22, S.Geng-1, J.Yu-32, S.Wang-92}@sms.ed.ac.uk

Abstract

For the sake of restoring whale populations, scientists manually annotate the shape of whales' tails and their unique properties through photo surveillance systems, so as to determine the whale species. The goal of the project is to build an algorithm to identify the whales in the pictures. We first operate Exploratory data analysis (EDA) on the database to summarize the main characters of the dataset. Subsequently, based on the main characters obtained, we pre-process the data, unify the image size, and convert the RGB image into a black-and-white image. We first employ a pre-trained Resnet101 model on the Happywhale dataset and the Mean Average Precision@5 (MAP@5) arrives 0.42. Then, we implement a Siamese Network from scratch and the MAP@5 reaches 0.67. Compared with Resnet101, Siamese Network can achieve a better result.

1 Introductions

The protection and recovery of whale populations have attracted global attention. However, adapting to warming oceans and competing daily with the industrial fishing industry for food remain serious problems. To help whale conservation, scientists use photo monitoring systems to monitor ocean activities. They use the shape of whale's tails and the unique markers found in the pictures to identify the whale species, and then carefully record the dynamics and movements of them[1]. Building a global database of whales helps scientists to understand population trends and conservation issues of whales, increasingly pressing questions as people use oceans ever more intensively.

In the past 40 years, most of this work has been done manually by scientists, leaving a huge trove of data untapped and underutilized. However, with the great improvement of technology, people can build systems to identify whales in images automatically instead of the manual classification recently. A basic but feasible idea is processing images in feature engineering and applying some machine learning methods like k-Nearest Neighbors (kNN) [6], Random Forests [2], Support Vector Machine (SVM) [15], etc., to recognize whales' tails. Furthermore, the wide use of deep learning has introduced Convolutional Neural Networks (CNN) [9] in image classification tasks, improving the performance of image classification tasks significantly [8].

For the dataset, there is an open platform named Happywhale[2] providing a database of over $25,000$ images of whales' tails gathered from research institutions and public contributors. We build a system based on this dataset to recognize different whales by their tails. The training data contains thousands of images of humpback whale flukes, and each individual whale has been identified by researchers and assigned an Id. While the test data only contains some images of whales, and their Ids need to be predicted by our system. The Id of whales is an incredibly powerful tool for research. It is very important when scientists track whales through their travels and immigration.

[1] https://www.kaggle.com/c/humpback-whale-identification.

[2] https://happywhale.com/home.

In this report, we explore and analyze the dataset firstly (Sec 2), and pre-process the data based on what we found (Sec 3). Then we present two methods to learn from training data: transfer learning with ResNet101 and Siamese Network with ResNet50 (Sec 4). Through observing experiment results, these methods can achieve a good performance in predicting Ids for whales (Sec 5).

2 Exploratory data analysis

In this part, descriptive statistics and inductive data analysis in the process of EDA are mainly introduced [14]. Through understanding the structure of data and visualizing the property of data, further work of pre-processing would be more effective, thus making a contribution to the final result of identification.

Happywhale's database provides totally $25,361$ images with $5,005$ unique classes in training dataset and there are $7,960$ images in test dataset. Firstly, the distribution of the dataset would be explored. Through grouping by the Ids of images, the count of images for each whale is obtained. Seaborn library is used to generate the distribution of the count of images as Figure 1 shown. It is clear that there are over $2,000$ whales only having 1 image sample, and about 32% of images coming from whales with 5 or less images. It is worth mentioning that there is a huge dis-balance in the data. About 38% of images are "new whale" rather than the whales with specific Ids.

Resolution distribution of images also needs to be considered. Figure 2 shows the proportion of different resolutions in training and test set respectively. Statistics information shows that there are $7,053$ different resolutions in the dataset. The resolution of $(1050, 700)$ accounts for the largest proportion in both training and test set. Most resolution types contain only a small number of image samples. To be concrete, for training set, the number of images represented by 60 types of resolution take up around 50 percentage, while for test set, the number of images represented by 50 different resolutions also make up 50 percentage. It means that although there are various image resolution types, the resolution size for most images is from one of a few options.

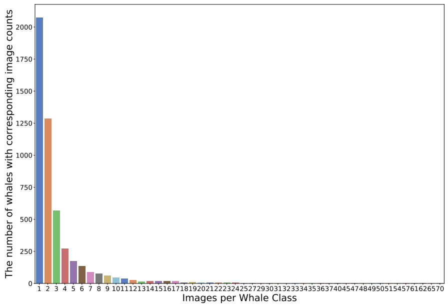


Figure 1: Distribution of number of images.

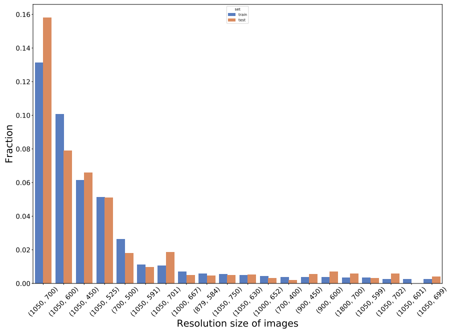
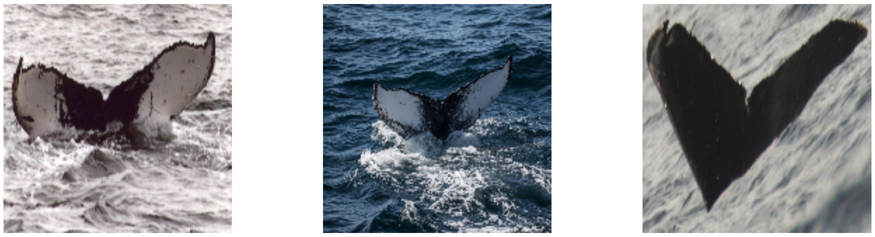


Figure 2: Distribution of resolution of images.

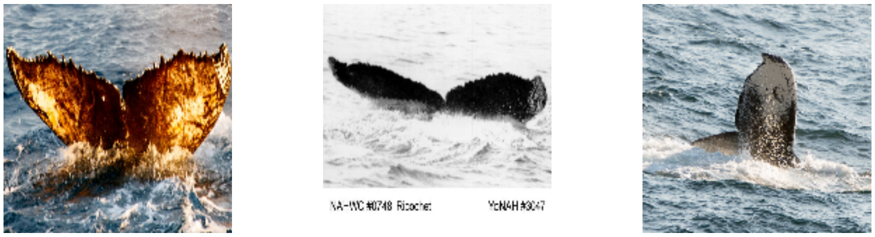


(a) Reddish

(b) Bluish

(c) Greenish

Figure 3: Examples of some colored images.



(a) Odd image (b) Image with text part (c) Whale with single fin

Figure 4: Examples of odd images.

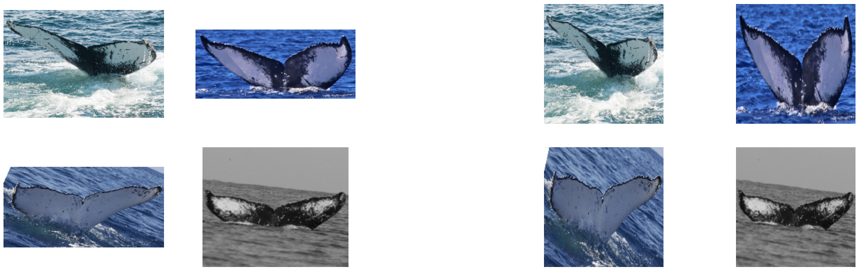
Furthermore, we explore color distribution in the dataset. It could be found that grey-scale images take up around 32% in training set and around 26% in test set. Moreover, some images are reddish or bluish or greenish. Figure 3 shows some examples. The possible factor could be light in the case of sunset. In addition, some odd images, such as abnormal whales and images with text, are explored as Figure 4.

Therefore, through EDA it can be summarized that there is a huge dis-balance in the dataset. About 38% of images are from "new whale" and 32% of whales only have less than 5 image samples. The rest 30% comes from whale classes with $6 - 73$ images. Moreover, this dataset contains $7,053$ different types of resolutions. In terms of distribution of colors, around 30% of images are grey-scale, and some images with different colors would cause difficulties for identification.

3 Data preparation

Based on the EDA, we can pre-process our data to make preparation for the later experiments.

The first step is resizing the images in our dataset. We find that the images of whales come from different sources with different image resolution sizes, so we cannot apply the same operation towards them unless they are transformed into the same size. Figure 5 shows an example of resizing images.



(a) Original images with different sizes (b) Resized images with the same size

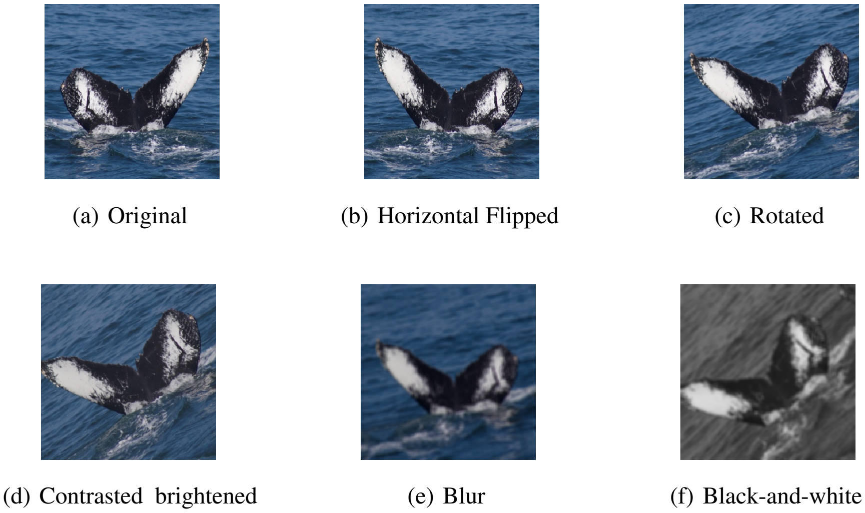
Figure 5: An example of resizing images.

The analysis about the color distribution of the dataset indicates a quite large variances of colors among these images. Thus, we need to operate some image transformations that are very agnostic to the RGB spectrum. Based on some previous experiments[3], we observe that it can achieve approximately the same accuracy by comparing two colored images, or two black-and-white images. However, comparing a colored image with a black-and-white image results in much lower accuracy. Thus, the

[3] https://www.kaggle.com/martinpiotte/whale-recognition-model-with-score-0-78563.

simplest and efficient solution is to convert all images to black-and-white to keep consistency with efficiency.

According to the analysis of the dataset, there are many classes with only one or a few samples, which means that the system could not learn them thoroughly and recognize them easily. The problem could be solved by applying augmentation operation towards pictures as an effective way to expand the dataset. Furthermore, the augmentation can also solve some other image problems we discovered previously. For example, some whales have yellow spots and some pictures are reddish, which can happen due to the sunset. We can randomly transform the brightness and contrast of some images to provide more examples. Based on the investigation of our dataset, we decide to adopt horizontal flipping, random rotation, random contrast, random brightness and random blur as the basic operations in the augmentation process. Figure 6 shows the transformation process when applying these operations onto an example image.



(a) Original

(b) Horizontal Flipped

(c) Rotated

(d) Contrasted brightened

(e) Blur

(f) Black-and-white

Figure 6: The transformation process when applying augmentation operations onto an example image.

As shown in Sec 2 (Figure 4), we find that there are some odd images of whales when exploring the dataset. Hence we look through the whole dataset and finally come out with 43 odd images. We delete these images from our dataset to avoid any misleading for models. After these pre-process of data, we can normalize all of the images to generalize the statistical distribution, and get prepared for our experiments.

4 Learning methods

This section describes two Neural Networks and corresponding tricks that are used in the experiment. It is formative to have three different datasets: training set, validation set and test set, but in this project only training set and test set are provided. In order to monitor the training procedure (i.e. avoid overfitting or underfitting), the training set was randomly split into training set and validation set with the ratio of $9 : 1$.

4.1 Transfer Learning with ResNet101

The project is to predict whales' Ids of images in the test set, which is an image classification task. Convolutional Neural Network (CNN) is an application of deep learning algorithm in the field of image processing and CNN has been applied in a wide range of computer vision tasks [1] because of its high performance and its simplicity in training. Thus, we decided to use CNN. Compared with other networks, Residual Networks (ResNets) [5] has been evaluated on the ImageNet [3] dataset and the result won the $1st$ place on the ILSVRC 2015 classification task. The residual nets with a depth of up to 152 layers which is 8 times deeper than VGG nets [11], can still have lower complexity on ImageNet dataset [5]. Considering the balance between time and complexity, we use ResNet101 finally.

ResNets can address the degradation problem. The main idea is to make the layers fit a residual mapping explicitly instead of hoping each few stacked layers directly fit a desired underlying mapping [5]. Figure 7 shows a building block. x is the input, $F(x) := H(x) - x$, $H(x)$ is defined as the desired underlying mapping. We let the stacked nonlinear layers fit mapping of $F(x) := H(x) - x$. Shortcut connections simply perform identity mapping and their outputs are added to the outputs of the stacked layers (see Fig. 7). The original mapping is recast into $F(x) + x$. The formulation of $F(x) + x$ can be realized by feedforwarding neural networks with shortcut connections (shown in Fig. 7).

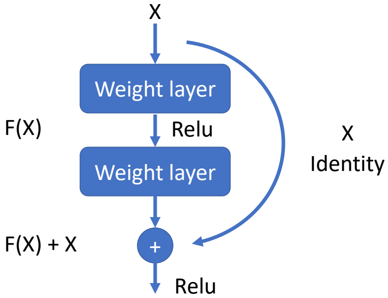


Figure 7: Residual learning: a building block.

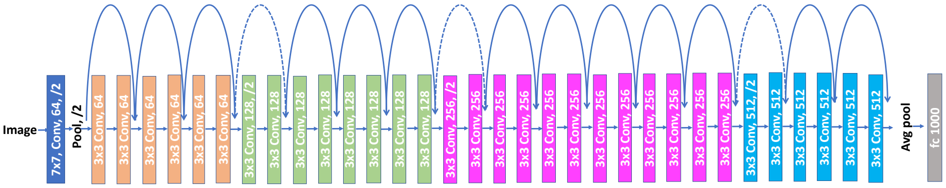


Figure 8: A ResNet with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions.

From the Figure 8, we can see the inserted shortcut connections. There are three kinds of shortcut connections. When the input and output are of the same dimensions, the shortcut connections can be used directly. When the dimension increases, there are another two options. One is that the shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. The other is used to match dimensions (done by 11 convolutions) [5].

After training the ResNet from scratch, we find the result is not as good as expected and the model really costs time to converge. An idea comes into our mind, using transfer learning to improve the accuracy and speed up the training process. The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task [13]. To be specific, it is to improve learning in new tasks by transferring knowledge that has been learned from related tasks [13].

The provided pre-trained model has $1,000$ classes. However, there are $5,005$ classes in this project as mentioned in EDA (Section 2). Thus, we remove the original classifier (which contains the last average pooling layer, fully-connect layer and Softmax layer) and then add a new classifier to satisfy our needs. The number of output of the last fully-connected layer of the new classifier is $5,005$. Finally, we fine-tune our model by freezing the convolution base. The main idea is keeping the convolution base in its original form, using a pre-trained model as a fixed feature extractor, and then providing its output to the following classifier.

4.2 Siamese Network with ResNet50

As mentioned in Exploratory data analysis (Section 2), there are $5,005$ classes or Ids and 32% of them only have 5 or less than 5 images. This leads to a huge difficulty for common classifiers which use softmax as the activation function of last fully-connected layer and Cross Entropy as the loss function.

This task is quite similar to human face recognition in which there are a huge number of different faces with only a few images. Inspired by [12, 10], metric similarity learning with Siamese Network was chosen [7]. In one propagation, a pair of images are fed into the two branches which are used to extract the features and finally generate two embedding vectors respectively by fully-connected layers. The similarity is calculated by Euclidean Distance (used in this task), Manhattan Distance or Cosine Similarity. The contrastive loss function [4] aims to make the similarity of two inputs from the same class higher and make it lower otherwise. According to [4], the exact loss function is

$$L(W, Y, \vec{X}_1, \vec{X}_2) = (1-Y)\frac{1}{2}(D(X_1, X_2))^2 + (Y)\frac{1}{2}max(0, m - D(X_1, X_2))^2 \tag{1}$$

where W represents weights, (Y, $\vec{X}_1$, $\vec{X}_2$) is a batch of labeled sample pairs. $X1$ and $X2$ are two embedding vectors respectively, and Y represents whether the two vectors are from the same class. D stands for the Euclidean Distance between two vectors. m is a positive value called margin.

In terms of the branches (also called Embedding Nets), CNN is used to extract deep features, followed by Fully-connected layers. In the project, we continue to use ResNets [5] to encode input images. However, instead of ResNet101, we use ResNet50 which consists of 50 layers (excluding the last average pooling layer and FC layer) because this is computationally cheaper, which means that it is more feasible for us to train the model with limited time and GPU resources. The usage of ResNet101 will potentially improve the performance. The pre-trained ResNet50 model is used and the parameters of Conv1, Conv2 and Conv3 are fixed during fine-tuning.

To implement Siamese Network, we do certain additional work on pre-processing. In order to embed tails more accurately, removal of the background is necessary. Less background information gives less redundant and noisy features to the embedding vectors. Therefore, we crop the original images by the bounding boxes around. The bounding boxes are generated by object detection networks, which has already been published by other competitors. Not only can this improves the quality of embedding, but also reduces the computation due to the smaller size of images. Accurate masks that are produced by image segmentation approach will potentially bring a superior performance of the feature embedding.

As Siamese Network requires a pair of inputs with the label of whether they are the same Whales, we implement a new data loader in Pytorch. The algorithm randomly chooses a sample excluding "new whale" and selects another sample with 50% probability from the same class and 50% probability from a different whale. The reason why "new whale" is not selected is that the class occupies too much proportion (38%). This unbalanced distribution will make the network over-fit the "new whale" but under-fit other whales.

5 Evaluation

5.1 Metric

In this project, we use Mean Average Precision@5 (MAP@5) to evaluate.

$$MAP@5 = \frac{1}{U}\sum_{u=1}^{U}\sum_{k=1}^{min(n,5)} P(k) * rel(k)$$

where U is the number of images, $P(k)$ is the precision at cutoff k, n is the number predictions per image, and $rel(k)$ is an indicator function equaling 1 if the item at rank k is a relevant (correct) label, zero otherwise.

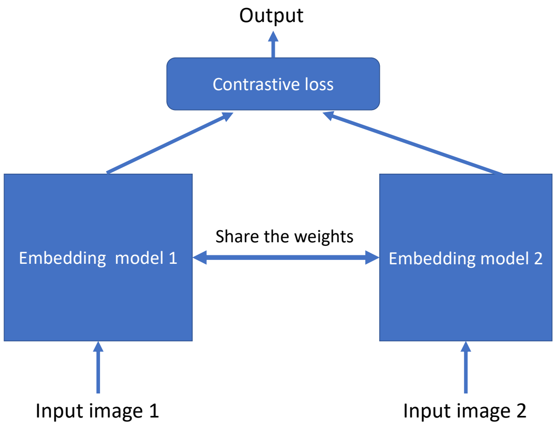


Figure 9: The architecture of Siamese Network. In the two branches, two same Networks share their weights. Two inputs are fed into the two branches to extract features and encode them. The contrastive loss is performed on the embedding vectors.

5.2 Results

Table 1 lists the result of each model. Training ResNet101 from scratch takes a huge amount of time but still does not achieve a descent result. Instead, we use pre-trained ResNet101 model on ImageNet and fine-tune the last fully-connected layer, which leads to an efficient and high MAP@5 (0.415). This is because training on the large image classification dataset allows the model to learn a good feature extraction CNN with the good generalization ability. Siamese Network achieves the highest Id recognition performance (0.678 MAP@5). Siamese Network does not learn a classifier directly but optimizes the embedding network to encode the whale tails such that the different whale pairs have low similarity and the same whale pairs have high similarity score. This is rather appropriate to the special dataset where there are a large number of classes (whale Ids) and each class only has a few images.

Models	MAP@5	Epoch
ResNet (training from scratch)	0.182	70
ResNet (transfer learning)	0.415	20
Siamese Network	**0.678**	100

Table 1: The comparison of models trained in the project.

6 Conclusions

By doing EDA on the dataset, we get some main features. The Happywhale's database provides totally 25, 361 images with 5, 005 unique classes in training dataset and there are 7, 960 images in test dataset. It is an imbalanced dataset, about 38% of images belong to "new whale" without a specific Ids. There are over 2, 000 whales only having 1 image sample, and about 32% of images coming from whales with 5 or less images. The color distribution in the dataset is another main feature. There are 7, 053 different types of resolutions. Around 30% of images are grey-scale, the others are reddish or bluish or greenish. The imbalance distribution of resolution cause the increased difficulty in resolution.

In this project, we implement two models. The ResNet101 with transfer learning and Siamese Network. ResNet, a typical CNN architecture is implemented for this task. To achieve superior performance and faster training speed, we unitize pre-trained ResNet101 model on ImageNet and fine-tune it on Happywhale dataset. We implement a Siamese Network from scratch which is particularly designed for recognition tasks. To keep consistency with the Kaggle competition, MAP@5 is our evaluation metric. ResNet101 trained 100 epochs from scratch achieves 0.182 MAP@5. When we use the pre-trained ResNet101 on ImageNet, the MAP@5 reaches 0.415 after the model is trained for only 25 epochs. The Siamese Network performs the best. The model trained after 100 epochs can achieve 0.678.

References

[1] Yoshua Bengio et al. "Learning deep architectures for AI". In: *Foundations and trends® in Machine Learning* 2.1 (2009), pp. 1–127.

[2] Anna Bosch, Andrew Zisserman, and Xavier Munoz. "Image Classification using Random Forests and Ferns". In: *IEEE 11th International Conference on Computer Vision* 1 (2007), pp. 1–8. URL: `https://ieeexplore.ieee.org/abstract/document/440906`.

[3] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[4] Raia Hadsell, Sumit Chopra, and Yann LeCun. "Dimensionality reduction by learning an invariant mapping". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 1735–1742.

[5] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: `1512.03385`. URL: `http://arxiv.org/abs/1512.03385`.

[6] Michal Irani, Oren Boiman, and Eli Shechtman. "In defense of Nearest-Neighbor based image classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1 (2008), pp. 1–8. URL: `https://www.computer.org/csdl/proceedings-article/cvpr/2008/04587598/12OmNyoiYTb`.

[7] Gregory Koch. "Siamese neural networks for one-shot image recognition". In: 2015.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`.

[9] Y. LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE*. Nov. 1998, 86(11):2278–2324.

[10] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: *CoRR* abs/1503.03832 (2015). arXiv: `1503.03832`. URL: `http://arxiv.org/abs/1503.03832`.

[11] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014). arXiv: `1409.1556`. URL: `http://arxiv.org/abs/1409.1556`.

[12] Y. Taigman et al. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. June 2014, pp. 1701–1708. DOI: `10.1109/CVPR.2014.220`.

[13] Lisa Torrey and Jude Shavlik. "Transfer learning". In: *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*. IGI Global, 2010, pp. 242–264.

[14] Chong Ho Yu. "Exploratory data analysis". In: *Methods* 2 (1977), pp. 131–160.

[15] Kai Yu et al. "Large-scale image classification: Fast feature extraction and SVM training". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1 (2011), pp. 1689–1696. URL: `https://www.computer.org/csdl/proceedings-article/cvpr/2011/05995477/12OmNx6PiF9`.

7 Contribution

Every member joined each part of the project and made an agreeable decision together after discussion. To make implementation efficient, each member is mainly responsible for one or two task(s). The exploratory data analysis was done by Siwei Wang, and Jiwei Yu pre-processed data based on the result of data analysis. Then Siqing Geng and Xingyu Jin implemented the transfer learning with ResNet101 and Siamese Network with ResNet50 respectively. We evaluate our models together and everyone completes the corresponding parts of the report.