Humpback Whale Identification

Identifying a whale by its tail

Vitalii Duma
Data Science Master Programme
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine

Serhii Tiutiunnyk
Data Science Master Programme
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine

***Abstract*—This document is a Machine Learning project report dedicated to the humpback whale identification problem which was set as a `kaggle` competition "Humpback Whale Identification". Our approach used for solving the problem is training a Siamese Neural Network [1] with triplet loss [2], proper data cleaning, augmentation and different branch model architectures.**

***Index Terms*—Siamese Neural Network, image embeddings, triplet loss, cosine similarity, metric learning, PyTorch, image identification, whales identification**

I. Introduction

A. Importance of the problem

The problem of recovering whale populations is very actual nowadays due to the adapting to warming oceans, intense whaling during the last several centuries and competition with the fishing industry for food [3].

Nature scientists use photos from surveillance system to register whales activity and migration [4]. They use a shape of the whales' tails and special marks there to identify species of a whale.

There are many public solutions of this problem [6] with good results. So, the goal of this project is to compare different branch model architectures for SNN and investigate dataset to get the most accurate model. Along with it, one of the goals of the project is to learn how to implement SNN architecture from scratch and train branch models using `PyTorch` [7] framework to retrieve image embeddings.

The goal of solution this problem is automation manual work for whale identification which will improve the monitoring process, help to get rid of routine job and increase the scientists' performance.

B. Potential impact

Solution of this problem can be applied for migration monitoring of other animals which might help scientists to take care about endangered species.

Along with it, some new heuristics and approaches applied to Siamese Neural Network (SNN) can improve existed solutions for other problems, such as:

- One-Shot Image Recognition [1]. It is very similar to this case dataset as the vast majority of classes have only one example. Since siamese networks for the first time study discriminatory functions for a large concrete data set, they can be used to summarize this knowledge and for completely new classes and distributions.
- Pedestrian tracking for CCTV [9]. In this project SNN is being used together with size and position features of images to detect several persons in the camera view area. SNN learns associations between several frames and trajectories.
- Matching resumes to jobs [10]. In this use case, SNN tries to match job offers and candidate's resumes. Here can be applied natural language processes (NLP) to retrieve deep contextual information from offer description and resumes, compare its embedding and force to increase distance between unsuitable pairs.

Siamese Neural Networks are widely used for solving problems in different ares. So comparison different architectures for branch models gives the baseline models for content-based image retrieval tasks with high similarity between images.

II. Data collection and preparation

A. Exploratory Data Analysis

The data for the whale identification task was provided by the organizers on the competition platform `kaggle.com` [11]. The dataset contains thousands of images of humpback whale flukes. Individual whales have been identified by researchers and given an `Id`. The target is to predict the `id` of images from the test set which will be used for evaluation. What makes this task such a challenge is that there are only a few examples per instance of whale `id` (`3,000+` whale `Ids`).

At first, take a look at random examples of whale's tail images [1].

After observing whales, we started to detect images replicas. There were two kinds of replicas found, complete copies and the same photos but with different brightness or contrast [2]. Replicas were found using perceptive hash algorithm [8]. If perceptive hash of two images differs no more than 6 bits, images have the same shape and mean squared error of their pixelwise difference is less than a threshold. Thus we found 4547 replicas in both train and test sets. Then we take only one image per replicas pair, preferably image with higher resolution and three color channels instead of one.

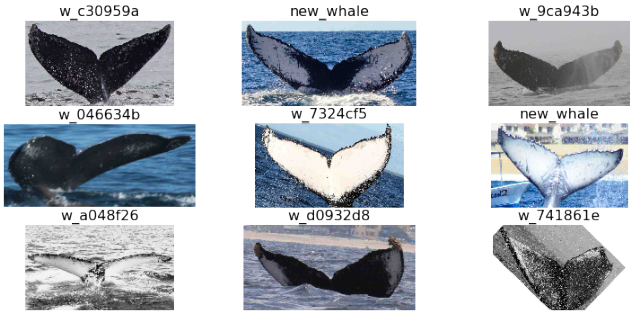

Fig. 1. Whale's tail examples

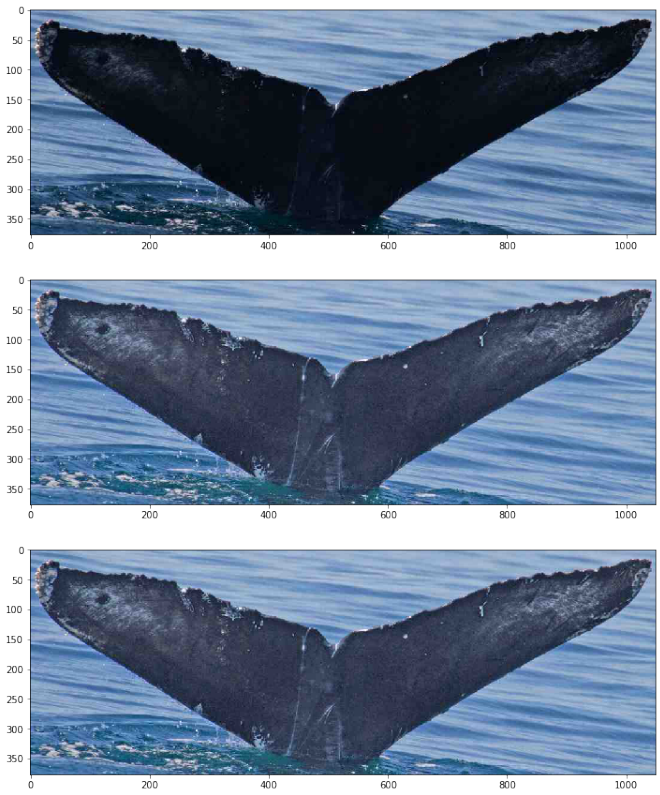

Fig. 2. Whales replica example

Moreover, the dataset is imbalanced in terms of the number of images per category. Let's draw the plot of the number of categories which have corresponded number of images. [3] shows the bar plot of the number of categories which have corresponded number of images. It means that vast majority of categories have only one or two examples.

Also there were found images of whales when whale's tail is upside down or the whole image is rotated. Kaggle user [**?**] prepared file with 10 rotated image names. We reverted images back. In the [4] shown an example of rotated whale's tail image.

It was noticed that there are images unhelpful or even harmful to training. These are images with two or more whale's tails per image or images which displays whale's body

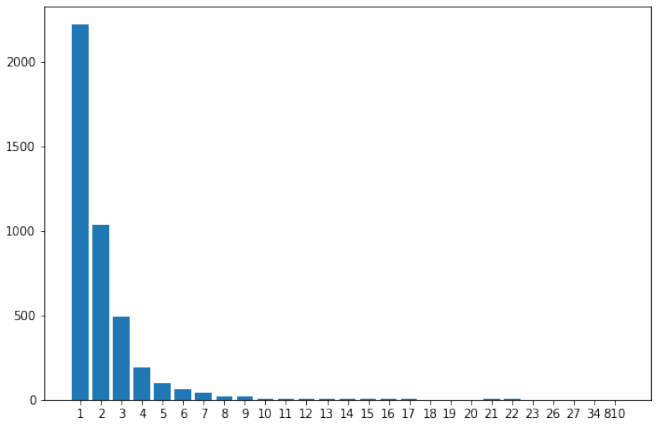

Fig. 3. Images per category

Fig. 4. Rotated image example

in addition to the whale's tail. So, such images were marked manually. In the [5] shown images excluded from the training set.

There is a special class `new_whale` which corresponds to unlabeled images. It is helpful to know how this class is located regarding to the other big classes and excluded

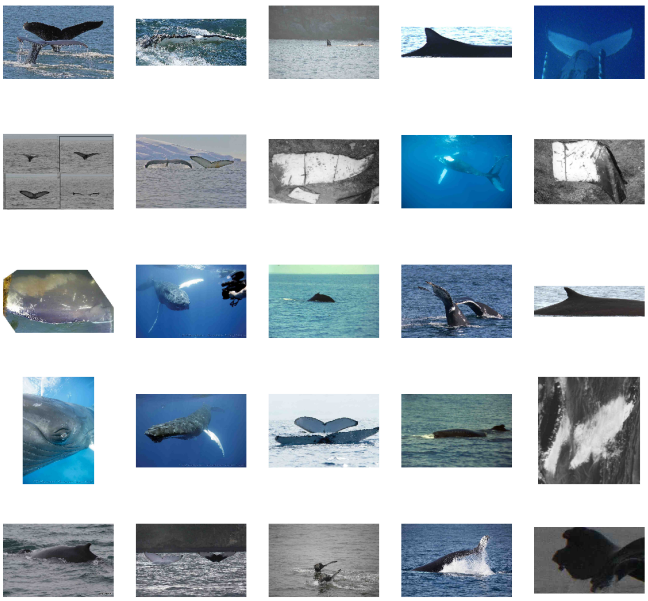

Fig. 5. Manually excluded images

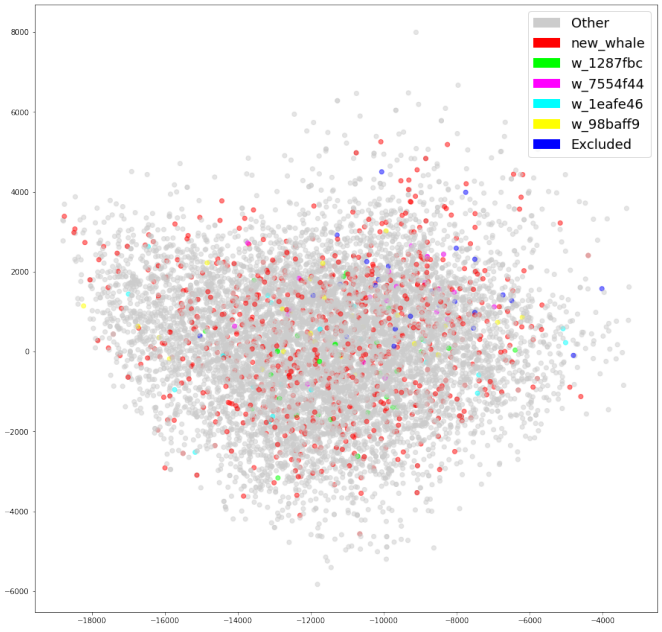

Fig. 6. Image data visualization

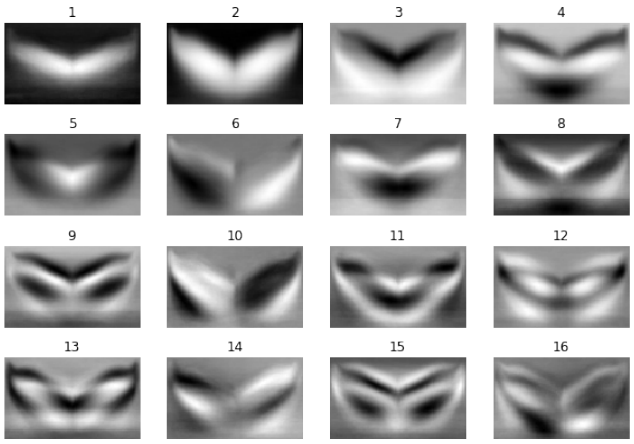

Fig. 7. First 16 principal components

images. We applied Principal Component Analysis [5] method to visualize images on the scatter plot. Fig. [7] shows the first 16 principal components. We can see that the first two components describe only the shape of whale's tail. In [6] we can see that unlabeled images are distributed very similar to the whole dataset. Also we can notice that excluded images are mostly concentrated in the top right area. Also we added classes with the biggest number of examples to the plot. This classes also can not be splitted into separated clusters. So, the variance of the data in each class is pretty high.

B. Data preparation

First of all, whole data set should be leaded to the same size. As it is shown on the [8], most images in dataset have high resolution. We transformed every image to the 224×224

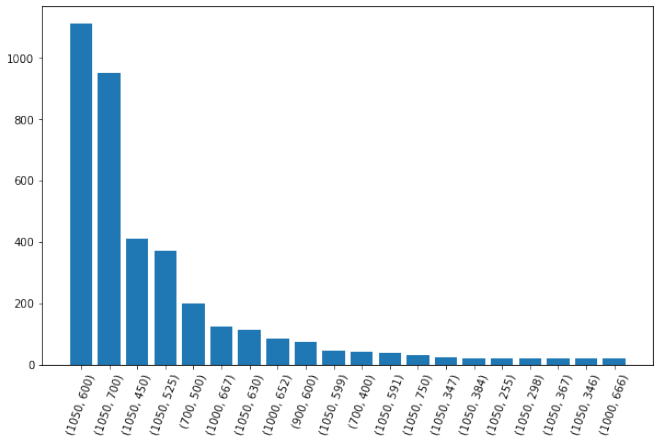

Fig. 8. Image size bar plot

dimension by affine transformation [14] using PIL library [17]. Affine transformation maps a rectangular image dimension to a square.

The big challenge in solving this problem was to find the method of generating appropriate pairs of images for the training SNN. Each element of batch is so called triplet. Triplet of images consists of anchor image, positive image which is whales tail's photo of the whale from the same category and negative example, which is tail's photo from another category. To train a good siamese model every triplet should have two images of whales of the same category so that their distance is big in sense of cosine similarity and also two whales from different categories but close in embedding space. The similar approach is widely used in adversarial training [12]. There are several approaches how to evaluate the difference (distance) between the images. The linear sum assignment [13] method was used to find the most difficult pairs of whales images for matching.

At first, as it was mentioned above, there are identical images in the training and test set. Some of them are pixel to pixel identical and the others differ by brightness background color, contrast, size etc. On the step of data preparing all such images were detected and cleaned.

At second, some images can contain more than one whale, the whole whale, or beach. These images were deleted from training dataset. There were also a few images of rotated upside-down whale flukes. They were accordingly rotated back. All images were cropped due to bounding-boxes data found in a Martin Piottes kernel [15]. Bounding boxes were not very accurate, so a cropping margin was introduced to save important parts of flukes.

Having one or two images per category is not enough. Thus we implemented data augmentation pipeline. This pipeline was created by using torchvision [18] package. On the first step we used color jittering, with 5% both contrast and saturation change. Any change in brightness did't improve the model. The next step in pipeline was random horizontal flipping, transforming images to tensors and image normalization. For

Fig. 9. Image augmentation example

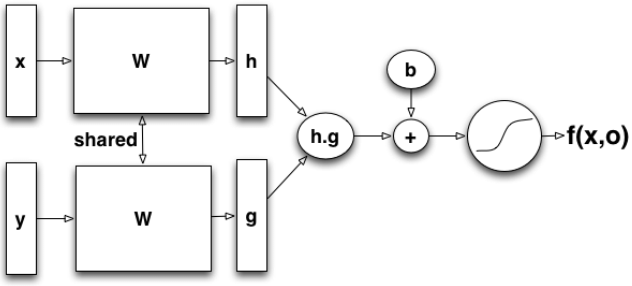

Fig. 10. SNN architecture

image normalization we took the same means and standard deviation which were used for training PyTorch pretrained models [19]. On the [9] shown an image augmentation example.

III. Modeling

Siamese neural network consists of two identical Convolutional Neural Networks (CNN) which transform the whale images to the embeddings vector. The general architecture of SNN is represented on [10]. The figure is published here [16]. The input of the model is two images of whales. The CNN is trained to increase the distance between outcome vectors if the input images have different `Ids` i.e. belong to different whales. In the other words, the same CNN with the same weights is being used for both input images.

By comparing each image from the test set with each image from the training set it is possible to determine the most likely whales by sorting them by the probability of a match.

There are many fields of experiments. For example, using different loss functions, head models on top of branch models, methods for finding closest classes by embeddings, embeddings size and margin threshold.

We started from tuning the number of the dimensions in embeddings space. For this experiment we used cosine similarity as a measure of distance between outputs.We tested 300-dimensional and 500 dimensional embeddings on ResNet18 and ResNet34. We have not tried using bigger embeddings on these resnets, because their last fully connected layers have only 512 features. Embeddings sizes 500 and 1000 were tested on ResNet50. 300-dimensional embeddings size showed worse

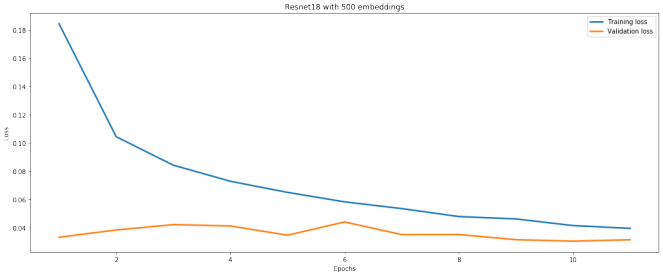

Fig. 11. ResNet18 with embedding vector size 500

result after 12 epochs than 500-dimensional on both ResNets. In average 500-dimensional embeddings showed better result than 300-dimensional embeddings from 4% to 6%. Along with it, we did not notice significant improve after setting embeddings size 1000 on ResNet50, that is why we decided to use 500 as optimal size.

Te results of experiments with 4 different branch models are shown here (`ResNet18` with embedding vector dimension `500` [11], `ResNet34` [12] with embedding vector dimension `500`, `ResNet50` [13] with embedding vector dimension `500` and `VGG16` [14] with embedding vector dimension `500`).

Then we started testing cosine similarity and logistic regression to find the best approach of comparing branch model output embeddings.

The first approach is cosine distance measure with threshold and a loss function. There are pros and cons of applying the distance measure.

First of all, the distance as a measure of vectors similarity is intuitively understandable. Moreover, swapping the order of input images does not change the result which is also makes sense.

Along with it, there some drawback of applying the distance measure.

- The distance measure will recognize two objects with a zero values as a perfect match of two images.
- At the same time, two vectors with large but a little bit different values will be considered as good match but not perfect as they are not equal.
- Also distance measure makes applying ReLU (Rectified Linear Unit) activation inappropriate as it zeros negative values which leads to loosing some part of information and force network to work only in the positive part of the space.

The second approach is just simple logistic regression model which takes two embeddings and predicts if they are similar or not. Each time cosine distance measure showed significantly better result than logistic regression.

When the best branch model and distance measure were chosen (`ResNet34`) we conducted an experiment with loss function margin. As we used cosine similarity as measure of distance, our margin doesn't depend on the number of features and their magnitude. If margin will be chosen too big, training will be slow and hard, because it would make model only care

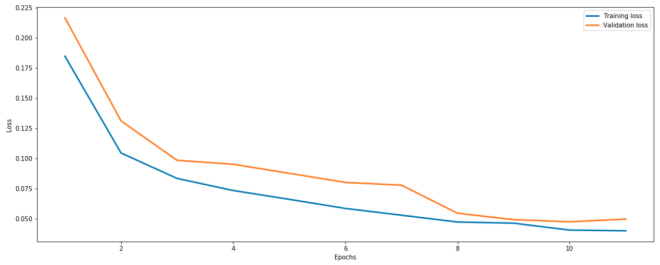

Fig. 12. ResNet34 with embedding vector size 500

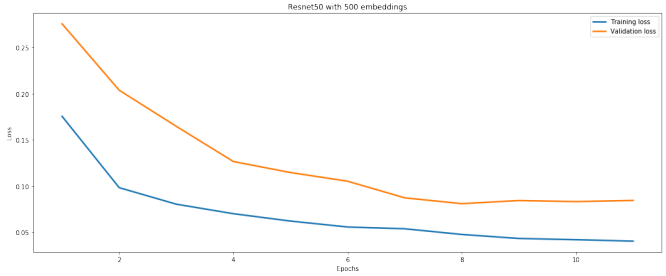

Fig. 13. ResNet50 with embedding vector size 500

about samples, which are too distant in embeddings space. On the other side, network with too low margin won't learn anything about distant examples. Thus, we were manually testing different margins. We checked paper [21] related to adjustment margin for distance measures. It suggests taking margin for cosine distance from 0.3 and up to 1. It would be more reliable to conduct more experiments with small step to choose optimal margin value but we took 4 values (0.3, 0.5, 0.7 and 0.9) with `ResNet34` as a branch model. It occured that margin values 0.7 showed the best result on validation set.

IV. Evaluation

There are two evaluation stages. The first one is `kaggle` competition evaluation and the second one is evaluation after the training to compare models before submitting to the `kaggle` checking.

A. Competition evaluation

Competition submissions are evaluated according to the Mean Average Precision 5. 5 is the number predictions per

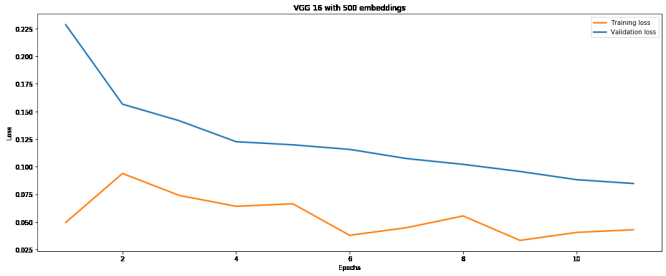

Fig. 14. VGG16 with embedding vector size 500

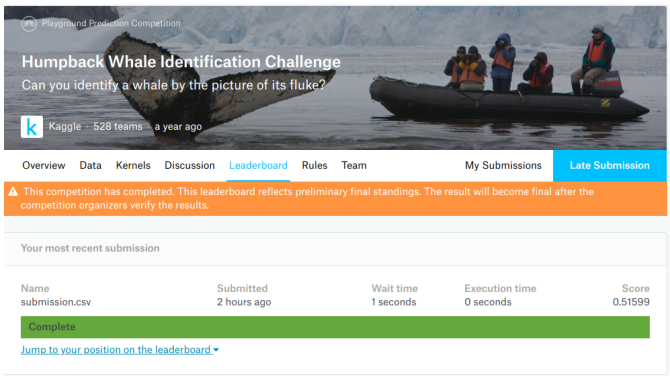

Fig. 15. Competition evaluation

image.

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

where U is the number of images, $P(k)$ is the precision at cutoff k, n is the number predictions per image. For each `Image` in the test set, model may predict up to 5 labels for the whale `Id`. Whales that are not predicted to be one of the labels in the training data should be labeled as `new_whale`.

`Kaggle` evaluation result is shown in the [15].

B. Model evaluation

The same evaluation measure is applied for local evaluation i.e. comparison models before submitting. As there is no labeled training set for developers, test set was taken from labeled training set. But there is an issue with splitting train and test sets as there are a lot of classes with only one image instance. Despite of having augmented images, it was decided not to take into account augmented images for evaluation.

Only classes which include more than 3 images were taken for model validation.

There can be applied some heuristics to calculation top 5 classes and including `new_while` class into the response. Top 5 classes are taken as the closest average cosine distance from the test image to the corresponded class.

Along with it, there is an unlabeled `new_whale` class and there are a lot of heuristics how to add this class to the top 5 list of response. The idea was to add `new_whale` to the response if the test image is approximately equidistant from the top 5 classes with some threshold.

The evaluation result of the best model (`Resnet34`) is shown in [12].

As an example, we can see top 5 closest images (including the original one) according to cosine distance and 500-dimensional embeddings [16]. All whale's tails look very similar.

Also we tried to visualize image embeddings applying PCA to it and check visually the quality of embeddings. First of all, we should admit that first two principal components of embeddings represent only 15% of data variety which shows

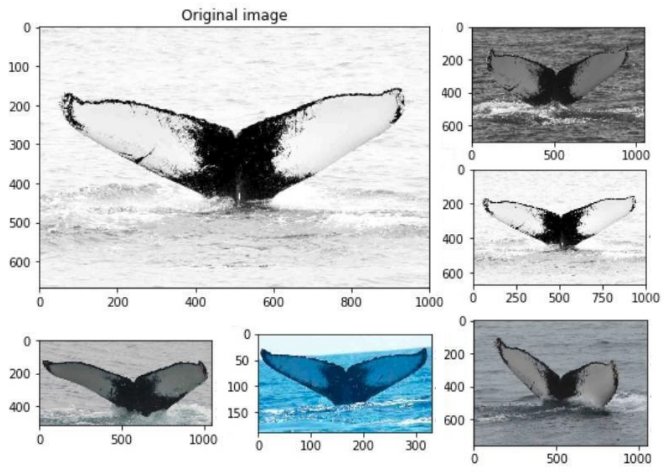

Fig. 16. Top 5 closest images

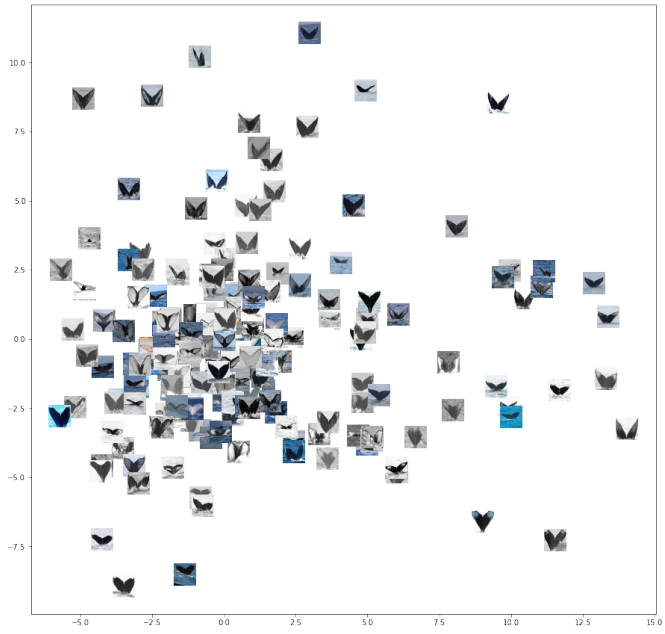

Fig. 17. PCA of embeddings. Images

a high quality of the data. As the coordinates of embedding vectors are more or less independent. In [17] we can see a disposition of images according to the first two principal components of its embeddings. What is even more interesting is representation the same classes projected on the principal components of embeddings. In [18] top 10 biggest classes are highlighted by color. We can see that images from the same class are situated closer to each other. Compering with PCA of pixels [6], where clusters of the same classes can not be detected, we can notice clusters of the same classes.

C. Discussion

`PyTorch` model, explonatory data analysis, dataset loader, data augmentation and other parts where developed from scratch. We received 0.51599 score, which would take us on

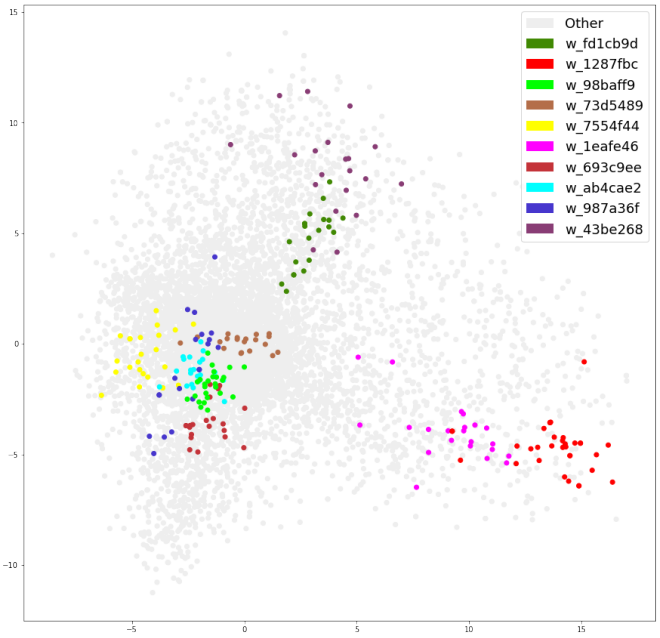

Fig. 18. PCA of embeddings. Classes

18th place in public leaderboard in competition [22]. Taking into account the fact that there were about four thousands categories with only one, two or three images, the result of the model categorization is good.

There is still room for different improvements.

D. Future improvements

There are some ideas worthy of our consideration. For example, we have not tried KNN classifier for measuring embeddings distance [20].

One more experiment is training branch models from scratch. Probably, this would lead to better results.

Another improvement could be testing other distance functions, for example euclidean distance.

We added new_whale category into each answer. This boosted our result a little bit. But there can be considered more complex heuristic for dealing with this category.

References

[1] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov. "Siamese neural networks for one-shot image recognition." ICML Deep Learning Workshop. Vol. 2. 2015.

[2] arXiv:1412.6622 [cs.LG]

[3] Endangered whales won't reach half of pre-hunting numbers by 2100 https://www.theguardian.com/environment/2017/aug/22/endangered-whales-wont-reach-half-of-pre-hunting-numbers-by-2011-study-says

[4] Whales are dying along East Coastand scientists are racing to understand why https://www.nationalgeographic.com/animals/2019/03/humpback-whales-unusual-mortality-event/

[5] Jolliffe I. Principal Component Analysis (2ed., Springer, 2002)

[6] Humpback Whale Identification Challenge Kernels https://www.kaggle.com/c/whale-categorization-playground/kernels

[7] PyTorch https://pytorch.org/

[8] The open source perceptual hash library https://www.phash.org/

[9] Laura Leal-Taixe, Cristian Canton-Ferrer, and Konrad Schindler. "Learning by tracking: Siamese CNN for robust target association." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2016.
[10] Maheshwary, Saket, Hemant Misra. "Matching Resumes to Jobs via Deep Siamese Network." Companion of the The Web Conference 2018 on The Web Conference 2018. International World Wide Web Conferences Steering Committee, 2018.
[11] Humpback Whale Identification competition dataset https://www.kaggle.com/c/humpback-whale-identification/data
[12] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae1,Shin Ishii "Distributional smoothing with virtual adversarial training" Graduate School of Informatics Kyoto University.
[13] Burkard R.E., Derigs U. (1980) The Linear Sum Assignment Problem. In: Assignment and Matching Problems: Solution Methods with FORTRAN-Programs. Lecture Notes in Economics and Mathematical Systems, vol 184. Springer, Berlin, Heidelberg
[14] Martin G.E. (1982) Affine Transformations. In: Transformation Geometry. Undergraduate Texts in Mathematics. Springer, New York, NY
[15] Martin Piotte's kaggle kernel model files https://www.kaggle.com/martinpiotte/humpback-whale-identification-model-files
[16] Gundogdu Batuhan, Keyword Search for Low Resource Languages, 2017
[17] Pillow https://pillow.readthedocs.io/en/stable/
[18] Torchvision package https://pytorch.org/docs/stable/torchvision/index.html
[19] PyTorch pretrained models normalization mean and std https://github.com/Cadene/pretrained-models.pytorch/blob/master/pretrainedmodels/models/torchvision_models.pyL61
[20] Transfer Learning part 2 https://medium.com/dataswati-garage/transfer-learning-part-2-zero-one-few-shot-learning-8d23d2f8583b
[21] CosFace: Large Margin Cosine Loss for Deep Face Recognition http://zpascal.net/cvpr2018/Wang_CosFace_Large_Margin_CVPR_2018_paper.pdf
[22] Humpback Whale Identification Challenge public leaderboard https://www.kaggle.com/c/whale-categorization-playground/leaderboard