Classifying Marine Mammals Using Convolutional Neural Networks

Aditya Katre[1] and Arpit Jasapara[2]

[1]Del Norte High, San Diego, California
[2]University of California Los Angeles, Los Angeles, California

December 2024

Abstract

¿¿¿This is a change to test if tracking changes works...¡¡¡Deep learning continues to rapidly advance image recognition capabilities, providing cutting-edge support for wildlife conservation initiatives. In this study, we develop a Convolutional Neural Network (CNN) framework to distinguish individual whales and dolphins extracted from the Happywhale dataset, demonstrating high precision in marine mammal identification. Our model achieves robust feature extraction and enhanced class separability by leveraging an EfficientNetB5 backbone with an Elastic ArcFace loss function. Multiple data augmentation techniques, including random cropping, grayscale conversion, and color manipulations, are employed to further strengthen the model's adaptability across varied imaging conditions. Additionally, a k-nearest neighbors (KNN) algorithm is incorporated at the inference stage to refine predictions, especially for previously unseen images. Through these combined strategies, our final model significantly boosts classification accuracy, achieving 88% accuracy and showcasing the efficacy of deep learning solutions for fine-grained identification tasks in marine mammal conservation. The promising results highlight the model's potential for streamlining research workflows and informing long-term conservation strategies for whales, dolphins, and other marine species.

Keywords: Marine Mammal Identification, Convolutional Neural Network (CNN), Deep Learning, EfficientNetB5, ArcFace, Data Augmentation, K-Nearest Neighbors (KNN), Wildlife Conservation, MAP@5, Population Tracking

1 Introduction

Identifying and classifying individual marine mammals is critical for research on marine mammal conservation efforts and population tracking. Traditional identification methods rely heavily on manual photo matching, which is time-consuming and prone to human error. Artificial intelligence and machine learning offer robust ways to automate the identification of marine mammals with high accuracy[1].sl

This research leverages the Happywhale dataset, which includes tens of thousands of labeled images representing fifteen thousand unique marine mammal individuals across 30 distinct species. The goal is to create a model capable of effectively and reliably distinguishing individual animals based on physical features such as fluke shape, skin patterns, and body size. This study leverages a pre-trained EfficientNetB5 backbone as the feature extractor and integrates an Elastic ArcFace loss layer to enhance model performance by maximizing the separation between classes (i.e., individuals).

Additionally, this research explores various image argumentation techniques to improve the model's generalization ability. This study demonstrates the potential of deep learning for wildlife monitoring and establishes a practical and scalable framework for similar applications where technology meets biology.

2 Background

The Happywhale data set is a whale and dolphin identification challenge consisting of over 51,000 labeled images representing 15,587 unique marine mammals over 30 different species. These images were collected through contributions from marine researchers and photographers supporting the identification, tracking, and conservation of marine

mammal populations around the world. Whales and dolphins are typically recognized by their distinct features such as fluke shape, dorsal fin markings, skin patterns, and shape[2]. However, the difficulty of manually classifying individual images of marine mammals has led to the need for artificial intelligence recognition systems to automate this task.

Identifying individual whales and dolphins is essential to understanding their populations, social dynamics, and migration routes. These marine mammals exhibit unique physical features, like markings or scars, which can serve as natural identifiers. Traditionally, scientists have collected photographs of these markings and manually matched them to existing records. Although this manual photoidentification method has proven to be effective over the years, it is often time consuming and expensive. As researchers gather larger datasets, the potential for human error increases, making it difficult to maintain accuracy. Additionally, this manual process can create significant delays in data analysis, which can be critical when monitoring the health of marine ecosystems and the effect of environmental changes[3]. Identifying individuals is challenging due to subtle variations in markings, environmental factors, and image quality, making machine learning models essential for use in the real world.

The primary objective of the study is to develop a model that can accurately and reliably classify individual whales, dolphins, and other marine animals within the Happywhale dataset. This high-performance identification model will support marine biologists and conservationists in tracking marine mammal populations. The approach utilizes modern machine learning techniques, such as transfer learning and data argumentation. These methods allow the model to handle the complexity and unpredictability of real-world images.

This research fills a critical gap in marine animal conservation by developing an automated and scalable approach to identify individuals. The methods used in this study are designed to achieve high accuracy and provide a practical tool for conservationists, directly contributing to the sustainability and conservation efforts of marine mammals worldwide.

3 Methodology

The methodology of this study is to develop the best possible deep learning model to classify whale and dolphin images into specific categories. The model's design and training processes were optimized to enhance both accuracy and generalizability, leveraging transfer learning, loss functions, and various techniques for data augmentation. Here are the key components used to set up the model:

The model uses EfficientNetB5 as its backbone. EfficientNetB5 is a convolutional neural network (CNN) that strikes a balance between accuracy and efficiency, ensuring the model runs promptly while minimally sacrificing accuracy[4]. The EfficientNetB5 model efficiently scales the neural network across three dimensions:

1. **Depth Scaling:** This increases the depth of training data, allowing the model to learn subtle patterns in images. This enables the model to learn from more complex, layered data. However, overusing this method may lead to higher computational demands and a decline in efficiency.

2. **Width Scaling:** This method of scaling adds more neurons to each layer of the model, expanding its width. As a result, the CNN can learn more nuanced patterns in data. Depth and width scaling work in harmony to effectively extract features from data.

3. **Resolution Scaling:** The EfficientNetB5 model scales images in a balanced manner, ensuring that images are high-quality and that nuanced features can be extracted while maintaining a decent level of computational demand.

The Elastic ArcFace loss layer is introduced as the classification head for this model. ArcFace increases the separation between different classes, improving the model's ability to distinguish images. It achieves this by introducing Additive Angular Margin Loss to the feature vectors. This additional margin in ArcFace creates a more robust distinction between image classes. This enhanced the model's discriminative learning and accuracy in identifying images across a large subset of classes. The loss layer operates by normalizing each feature and then using a cosine similarity metric for each feature. Then, ArcFace makes it easier to distinguish between classes by adding a fixed angle, or angular margin, between the feature vector and target class. This contributes to making class boundaries more distinct, enforcing a distinct separation between classes. The Elastic variation of ArcFace introduces elasticity, or randomness, to the angular margin, causing the model to perform better

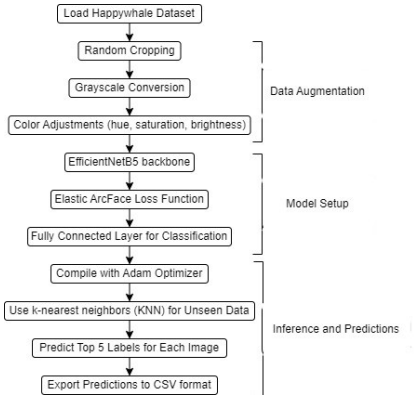

Figure 1: A visual representation of the third model.

against slight, almost random variations in images that are part of the same category[5].

3.1 Data Preparation and Augmentation

Kaggle's Happywhale dataset is split into training and testing sets. A CSV file containing the filename for each image and the corresponding label for the image is included. Some labels include: *bottlenose_dolphin*, *beluga*, *killer_whale*, *blue_whale*, and many others. Each image in the training set undergoes a series of transformations to improve the model's capability to process and develop an understanding of features in the images. The key transformations that the images undergo include:

- **Random Cropping:** Images are cropped randomly using full-body, YOLOv5, Detic, or Vision Transformer (ViT). This allows the model to focus on specific body parts of the whale or dolphin.
- **Color Adjustments:** Hue, saturation, contrast, and brightness are randomly altered to allow the model to handle various lighting and color conditions effectively.
- **Grayscale Conversion:** Converting images to grayscale enhances the model's ability to recognize patterns based solely on pixel intensity values.

3.2 Training Strategy

Training occurred on a Tensor Processing Unit (TPU) on Google Cloud, significantly speeding up training times for this model and using fewer local resources. The batch size is set to 16 * the number of TPU replicas, leveraging Google Cloud's TPUs. Some methods used to make training more efficient include:

- **Multiple-Sample Dropout:** To improve generalization, multiple-sample dropout (MS-Dropout) is used with different dropout rates on feature embeddings. The outputs are averaged to reduce overfitting while retaining rich feature information[6].

3.3 Inference and Threshold Tuning

When inferencing, the model uses k-nearest neighbors to match each image in the testing set to the closest image it trained with using cosine similarity. This step allows the model to make accurate predictions by matching features in training images with testing images[7].

A thresholding strategy is utilized to refine the weightage between known, identifiable labels and "*new_individual*" (unknown) labels. This threshold is enhanced using the validation set to maximize precision and distinguish known and new images of animals. If a higher threshold is selected, the model will be more conservative in predicting "*new_individual*", whereas a low threshold is more inclusive to assigning to labeling certain classes.

3.4 Ensemble and Blending

The final model predictions are generated through a blending strategy that combines multiple model performances that range between several different epochs and crop augmentations. Additionally, snapshots were taken during various stages of training of the model. These checkpoints were blended using optimized weights to improve the performance of the model. Optimized weights were also used for each model in the ensemble to ensure that the accuracy of the ensemble is maximized[8]. This is represented by the following equation:

$$P_{\text{final}} = w_1 P_1 + w_2 P_2 + \cdots + w_n P_n$$

where w_i represents the weight assigned to model i and P_i is its prediction.

3.5 Evaluation Metrics

The primary evaluation metric employed in the Happywhale competition on Kaggle is the Mean Average Precision at 5 (MAP@5). This metric is

specifically designed to measure the quality of a model's top-five predictions for each image in the test set, thereby assessing how effectively the model prioritizes correct labels at the highest ranks. By limiting the evaluation to the top five predictions per image, MAP@5 focuses on the model's ability to quickly and accurately pinpoint the correct individual from a shortlist of candidates, rather than requiring it to produce a perfectly ordered, full-length ranking. Such a constraint reflects practical scenarios in which only the top few suggestions matter most, such as identifying a specific whale or dolphin from a large database.

Additionally, MAP@5 is applied consistently to both validation and test sets, ensuring that the metric used during model tuning and selection is the same as that used in the final evaluation. This consistency fosters reliable model comparison and supports a more robust understanding of how well the model can generalize to new, unseen images.

To define MAP@5 more concretely, let U be the total number of images in the test (or validation) set, and for each image, assume the model produces n predicted labels ranked from most to least likely. The value k represents the position (or cutoff) in the ranked list, with $k \in \{1, 2, 3, 4, 5\}$, since we only consider the top five predictions. For each image $u \in \{1, 2, \ldots, U\}$, we denote its correct label as a "relevant" item. The model's task is to place this relevant label as high as possible among its predicted labels.

We define a relevance function $\text{rel}(k)$, which equals 1 if the correct label is found at rank k and 0 otherwise. The precision at cutoff k, denoted $P(k)$, is computed as the fraction of correct labels among the top k predictions:

$$P(k) = \frac{\text{Number of relevant labels in top } k}{k}.$$

Since only one correct label exists per image (under the assumption that each image corresponds to a single individual), any rank k beyond the first occurrence of the correct label does not contribute additional precision improvements.

The MAP@5 metric is then calculated by summing the precision values at each cutoff k where the correct label is found and then averaging this quantity across all images. Formally, if we let n be the number of predictions for each image, and consider only up to $\min(n, 5)$ predictions for MAP@5, the formula can be expressed as:

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

In this formula, U represents the number of images, $P(k)$ represents the precision at cutoff k, n is the number of predictions per image, and $rel(k)$ indicates whether the item at rank k is a correct label. If the item is an incorrect label, the $rel(k)$ is set to zero.

Once an image is labeled correctly, it's marked as "found" and is no longer considered in the evaluation. On the other hand, if the model predicts the correct label multiple times in a row for a given image, only the first predictions counts for the $MAP@5$ evaluation, and the rest are dropped.

To illustrate this with a simple example, consider a single image with the correct label "A." Suppose the model's top five predictions are [A, B, C, D, E]. The table below demonstrates how $rel(k)$ and $P(k)$ are computed:

Rank (k)	**Prediction**	$rel(k)$	$P(k)$ **(Precision at** k**)**
1	A	1	$\frac{1}{1} = 1.0$
2	B	0	$\frac{1}{2} = 0.5$
3	C	0	$\frac{1}{3} \approx 0.333$
4	D	0	$\frac{1}{4} = 0.25$
5	E	0	$\frac{1}{5} = 0.2$

Table 1: Example computation of $rel(k)$ and $P(k)$ for an image with the correct label "A." The first prediction is correct, so $rel(k)$ is 1 for $k = 1$, and subsequent predictions contribute 0.

In this scenario, the correct label "A" appears at the first position $k = 1$, so only $P(1) = 1.0$ contributes to the average precision for this image. Consequently, the Average Precision for this single image is 1.0. If we had multiple images, we would compute each one's Average Precision similarly and then average these values to arrive at MAP@5.

By relying on MAP@5, the evaluation ensures that the model does not simply identify the correct individual somewhere down a long list of predictions but ranks it highly within the top five guesses. This is especially important in applications such as whale and dolphin identification, where researchers and conservationists need rapid and reliable identification to inform population tracking, behavioral studies, and conservation strategies. The MAP@5 metric thus provides a practical and meaningful measure of model performance that aligns well with real-world requirements.

Combining the EfficientNetB5 backbone, Elastic ArcFace facial discrimination capabilities and other classification methods results in an effective and comprehensive approach to identifying individ-

ual whales and dolphins. The result of this use of creating iterations of models for the Happywhale dataset is a model capable of achieving high accuracy with fine-grained mammal identification tasks.

4 Results

The experiments for this project were conducted on four distinct deep learning models that identified individual marine animals using various conditions. The Mean Average Precision at 5 was employed as an evaluation metric. Below, we present the outcomes of these models, along with several diagrams, tables, and figures to visualize the architecture of these models and their performances.

Four distinct models, each developed using various architectures and data augmentation techniques, were evaluated:

Model	Accuracy (%)
Basic CNN	0.10
ResNet and ArcMargin	0.48
Baseline Model 1	0.77
Baseline Model 2	0.79
EffNetB5 and ElasticArcFace	0.86
Blend of Models	0.88

Table 2: Accuracy results for different models.

The models' performance, as measured by Mean Average Precision at 5, shows that the blend of models and the third, advanced model demonstrate significantly greater amounts of discriminative power within the top five predictions. The results of these models show the effectiveness of the advanced architectures employed in the third and fourth models and their ability to handle the complexities of the Happywhale dataset and individuals that appear to be visually similar.

4.1 Model Architectures and Code Excerpts

To ensure reproducibility and demonstrate the complexity of the models, Figures 1–4 show snippets of the TensorFlow/Keras code and Matplotlib graphs used for training each model, embedded as code listings. These snippets provide an overview of the layers, data preprocessing steps, and training loops employed.

```
import tensorflow as tf
from tensorflow.keras import layers, models

model = models.Sequential([
    layers.Conv2D(32, (3,3), activation='relu',
        input_shape=(128,128,3)),
    layers.MaxPooling2D((2, 2)),
    ...
    layers.Flatten(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.45),
    layers.Dense(len(label_encoder.classes_),
        activation='softmax')
])
model.compile(optimizer='adam',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])
```

Figure 2: Code snippet for Basic CNN model.

This first model's simplistic architecture and relatively short training regime resulted in the lowest accuracy among the four tested models for the Happywhale dataset.

```
def get_model():

    EMB_DIM = 512
    N_CLASSES_MODEL = N_CLASSES

    with strategy.scope():
        inp = tf.keras.layers.Input(shape=(*IMAGE_SIZE,
              3), name="inp1")
        label = tf.keras.layers.Input(shape=(), name="
             inp2")
        model_feat = SwinTransformer('swin_large_384',
             num_classes=N_CLASSES_MODEL, include_top=
             False, pretrained=True, use_tpu=True)
        embed = model_feat(inp)
        embed = tf.keras.layers.BatchNormalization()(
             embed) # batch norm or L2
        embed = tf.keras.layers.Dropout(0.2)(embed)
        embed = tf.keras.layers.Dense(EMB_DIM, name="
             dense_before_arcface", kernel_initializer=
             "he_normal")(embed)
    ...
        model = tf.keras.Model(inputs=[inp, label],
             outputs=[output])
        embed_model = tf.keras.Model(inputs = inp,
             outputs = embed)

        model.compile(
            optimizer=tf.keras.optimizers.Adam(
                 learning_rate=1e-3, epsilon=1e-5),
            loss = [ tf.keras.losses.
                 SparseCategoricalCrossentropy()],
            metrics = [tf.keras.metrics.
                 SparseCategoricalAccuracy(),
                      tf.keras.metrics.
                           SparseTopKCategoricalAccuracy
                           (k=5)]
        )
        model.summary()

        return model,embed_model
```

Figure 3: Function to create the EfficientNetB3 model.

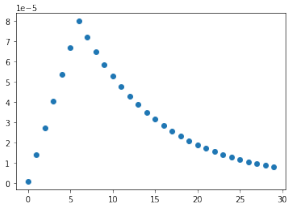

Figure 4: Learning rate schedule used during training, demonstrating an initial warm-up phase where the learning rate rapidly increases, followed by a peak and a gradual decay over subsequent epochs.

The graph illustrates a learning rate schedule employed during the training process of the model with an EfficientNetB5 backbone. The sharp increase then gradual decrease represents the dynamic adjustment of the learning of the model over time. The learning rate (y-axis) initially follows a warm-up phase, which is quickly followed by a peak. As the number of epochs (x-axis) increases, the learning rate gradually decays, which results in the optimization of the model.

The blend of models, illustrated by the integrated blending and augmentation strategies in the code snippet below (Figure 4), reached the highest accuracy at 0.88. The difference in accuracy using MAP@5 is subtle but significant due to the refinement of inference strategies and data handling.

```
test['predictions'] = test.apply(lambda row:
     blender([...], [...]), axis=1)
submission = pd.DataFrame({'image': test_generator.
     filenames, 'predictions': test['predictions'
     ]})
submission.to_csv('submission.csv', index=False)
```

Figure 5: Code snippet showcasing the blending of multiple independent models, achieving an accuracy of 0.88.

The blending approach shown in Figure 4 uses predictions from various models to produce a more well-rounded and robust final result. This demonstrates how several models working in combination can result in increased overall production quality.

In summary, the four models showcased progressed from a relatively simple CNN with a score of 0.10 to a high-performing, blended solution that achieved a score of 0.88, which is higher than other baselines showcased in Table 2. These results represent the outcome of creating several iterations and improvements in making predictions in challenging, fine-grained datasets.

5 Discussion

The performance of the different models tested in this work provides a series of lessons learned about both the challenges and solutions proposed for the individual identification of whales and dolphins. Analyzing the architecture, optimization strategies, and data augmentation techniques used by the various models allows us to identify which factors contribute most to their relative success.

5.1 Basic CNN Model

The Basic CNN Model was implemented using a barebones Convolutional Neural Network (CNN) that served as a baseline model to achieve a score

of 0.10. Its limited predictive capabilities can be attributed to several shortcomings apparent in its design and training. Firstly, the architecture of the model required more depth and complexity to increase the model's discriminative capabilities. The model contained only four convolutional layers and a small amount of image adjustments, both leading to low accuracy. On average, this model struggled with capturing intricate patterns present in the marine mammals in this dataset, such as unique pigmentation, scars, and fin shapes that may be used to distinguish among individuals. In addition, the input images were trimmed, resulting in a loss of fine-grained details necessary for classification.

Additionally, the data augmentation techniques utilized in the Basic CNN Model—such as rotation, width/height shifts, and zoom—fell short in replicating the varied viewing angles, lighting conditions, and occlusions typical of real-world images. The implementation of a dropout layer at a 50% rate to prevent overfitting might have unintentionally restricted the model's capacity to capture subtle features. Furthermore, sparse categorical cross-entropy was adopted as the loss function, yet without incorporating more sophisticated methods like margin-based loss, which constrained the model's effectiveness in differentiating between visually similar individuals effectively.

5.2 ResNet and ArcMargin Model

The ResNet and ArcMargin Model introduced significant improvements in accuracy using a ResNet-based architecture and incorporating the ArcMargin loss function, resulting in a significant increase in accuracy to 0.48. The ResNet backbone for this model enabled the model to recognize more nuanced patterns in images compared to the simple CNN. The ResNet and ArcMargin model also mitigated the vanishing gradient problem, reducing the likelihood of training rates stalling or stopping.

The use of the ArcMargin loss function increased the discriminative power of the models by adding an angular margin to the classification boundary between image labels. This resulted in increased discriminative capabilities for the model, and a higher score using the MAP@5 metric.

However, the model's performance plateaued due to certain restrictions on the training strategy and data. In this model, the ArcMargin loss function improved the separation between classes, but it was not an optimal solution for every class in the Happywhale dataset, which contained an imbalance across various species of marine mammals. Additionally, the data processing techniques in this model were relatively basic, resulting in the model's inability to adapt to the real-world variances presented in the dataset.

5.3 EfficientNetB5 and Elastic ArcFace Model

The EfficientNetB5 and Elastic ArcFace Model demonstrated a significant leap in terms of performance compared to models discussed in Sections 5.1 and 5.2, and it earned an accuracy of 0.86. The introduction of the Elastic ArcFace function resulted in increased distinctions between classes of marine mammals by adding randomness to the angular boundary between classes. This randomness allows the model to better handle slight variations in the appearance of an individual. This is especially crucial for Happywhale, where images of the same animal can vary due to lighting, pose, color, and environmental conditions.

The use of an EfficientNetB5 backbone also boosted the model's performance. The backbone's compound scaling strategy helped create a balance between depth, width, and resolution in the Convolutional Neural Network. During training, a higher-quality 380x380 resolution was used, allowing the model to capture and learn from more subtle and intricate details in images.

The data augmentation was also improved in the EfficientNetB5 and Elastic ArcFace Model, using grayscale transformations, color adjustments, and random cropping. The use of these techniques reduced the likelihood of overfitting. Additionally, employing k-fold cross-validation ensured that the model would generalize well against unseen data.

5.4 Blend of Models

The Blend of Models achieved the highest accuracy of 0.88 by combining the advancements made in the other models. This approach to the machine learning challenge combined predictions from multiple models that each contained varying data augmentation techniques.

Model snapshots were also impactful in obtaining a high score. Each snapshot represented the model at a different stage of training, capturing diverse perspectives on the data. This training strategy capitalized on the strengths of each model's checkpoints while minimizing the weaknesses. This allowed the model to generalize more effectively and achieve a higher score using the MAP@5 evaluation metric. The score of the blend was higher than that

of the standalone EfficientNetB5 backbone, showing how blending strategies can be more adept at categorizing these images.

The ensemble also benefits from thorough hyperparameter tuning. For example, the weights given to each snapshot in the blend were fine-tuned through cross-validation, which helped the ensemble achieve the highest overall accuracy. Moreover, employing Elastic ArcFace loss with adaptive margins improved feature discrimination, especially in difficult situations where several animals looked alike.

5.5 Key Learnings

1. Loss Functions: Transitioning from basic, cross-entropy loss functions to more advanced ArcMargin and Elastic ArcFace loss functions had a positive impact on the ability of the models to classify marine mammals. While cross-entropy loss functions are great for most tasks, they typically focus solely on minimizing the error percentage instead of enforcing the separation of different classes in the parameter space. However, the ArcMargin loss function introduced an angular margin to separate classes, forcing the model to place the predictions farther apart. This enhances the model's capability to make distinctions between visually similar species while maintaining its ability to effectively classify images. Finally, the Elastic ArcFace Loss function builds on ArcMargin by adding an element of randomness to the margin between categories. This allowed the model to be more robust to slight variations in images caused by things such as lighting or camera angles.

2. Model Architectures: The use of model architectures was also critical to the success of more advanced models. The use of the EfficientNetB5 model architecture allowed the model to outperform the basic CNN architecture. This is attributed to this backbone's ability to effectively scale across various levels of depth, width, and resolutions. Selecting architectures that strike a balance between computational efficiency and representational power is important for this machine learning task.

3. Data Augmentation: Using data augmentation techniques enhances the robustness of the models through grayscale transformations and color adjustments. This shows the importance of data augmentation strategies in mimicking real-world variances in images and helping the model effectively generalize.

4. Ensembles: The ensemble approach to this Machine Learning challenge showed how combining multiple perspectives on the same data can yield significantly better results. Model four effectively took advantage of the various strengths in snapshots, leading to better performance in the dataset.

5. Handling Imbalanced Data: The Happywhale dataset has an unequal distribution of images with an unequal distribution of training samples for certain individuals. However, our models resolved this issue using margin loss and cross-validation to help the model generalize and become more robust to unseen data.

6 Conclusion

This study is an application of advanced deep learning methodologies to overcome the challenge of identifying marine mammals using the Happywhale dataset. The models leveraged modern convolutional architectures, loss functions, and data augmentation strategies. This research efficiently addresses the challenge of classifying fine-grained image data to advance ecological research. This study showcased models that progressed from a baseline Convolutional Neural Network CNN model with limited classification capabilities to high-performing models that incorporate advanced backbones and loss functions to effectively make predictions.

The iterative approach of progressing through four separate models shows that refining components ranging from network depth to augmentation methods can lead to breakthroughs in the performance of the machine learning model. Specifically, the k-nearest-neighbors algorithm proved to be critical in categorizing unseen images while maintaining a clear, defined boundary between marine mammal species. Using all the techniques described in this study, an identification system has been created to assist environmentalists in tracking marine mammalian populations without relying on labor-intensive manual matching.

Beyond its immediate application, this research contributes to the field of wildlife conservation by providing an efficient and scalable way to track species. The model's ability to identify marine

mammals is vital to monitor populations and protect marine ecosystems.

From a wider viewpoint, the findings in this study highlight the practicality and criticalness of implementing deep learning solutions to monitor wildlife and to make inferences on carrying capacity, food webs, and more. The use of machine learning also provides an instant mechanism to support ecological studies and inform conservation programs. This research shows how machine learning helps bridge the gap between technology and environmental science, and it proves to be a vital tool to preserve marine ecosystems for our posterity. The approaches utilized and the lessons learned during this study pave the way for broader adoption of Artificial Intelligence in ecological research.

7 Refrences

1. K.N. Shivaprakash, N. Swami, S. Mysorekar, R. Arora, A. Gangadharan, K. Vohra, M. Jadeyegowda, J.M. Kiesecker. Potential for artificial intelligence (AI) and machine learning (ML) applications in biodiversity conservation, managing forests, and related services in India. Sustainability. 14, 7154 (2022). https://doi.org/10.3390/su14127154

2. Kaggle. Happywhale - Whale and Dolphin Identification. https://www.kaggle.com/competitions/happy-whale-and-dolphin (2022).

3. NOAA Fisheries. Marine mammal photo-identification research in the Southeast. https://www.fisheries.noaa.gov/southeast/endangered-species-conservation/marine-mammal-photo-identification-research-southeast (2024).

4. M. Tan, Q. V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 (2019).

5. J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia, S. Zafeiriou. ArcFace: additive angular margin loss for deep face recognition. arXiv preprint, https://arxiv.org/abs/1801.07698 (2018).

6. H. Inoue. Multi-sample dropout for accelerated training and better generalization. arXiv preprint, https://arxiv.org/abs/1905.09788 (2019).

7. The scikit-learn developers. Nearest neighbors. https://scikit-learn.org/stable/modules/neighbors.html (2025).

8. M. Shahhosseini, G. Hu, H. Pham. Optimizing ensemble weights and hyperparameters of machine learning models for regression problems. arXiv preprint, https://arxiv.org/abs/1908.05287 (2019).