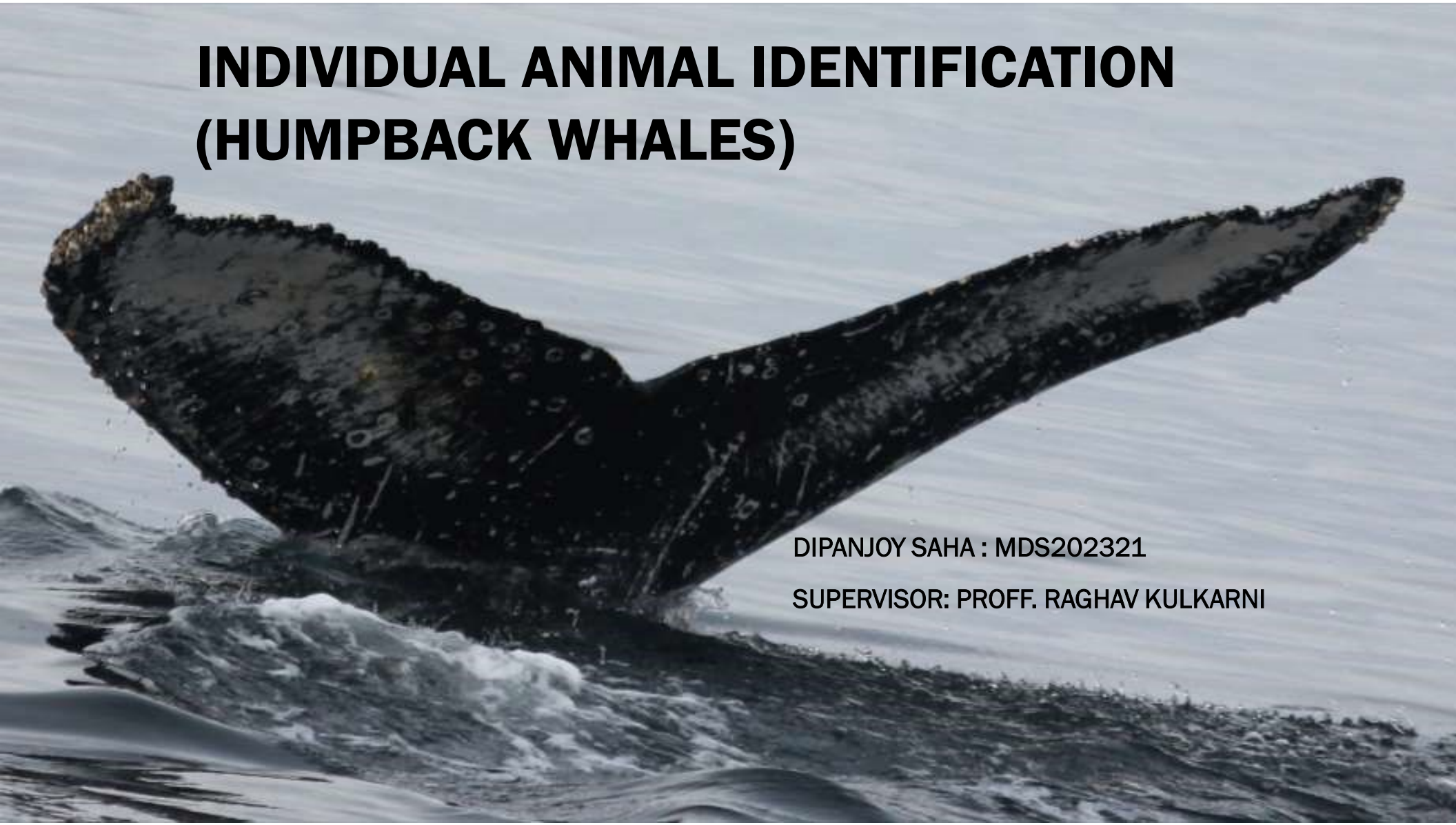
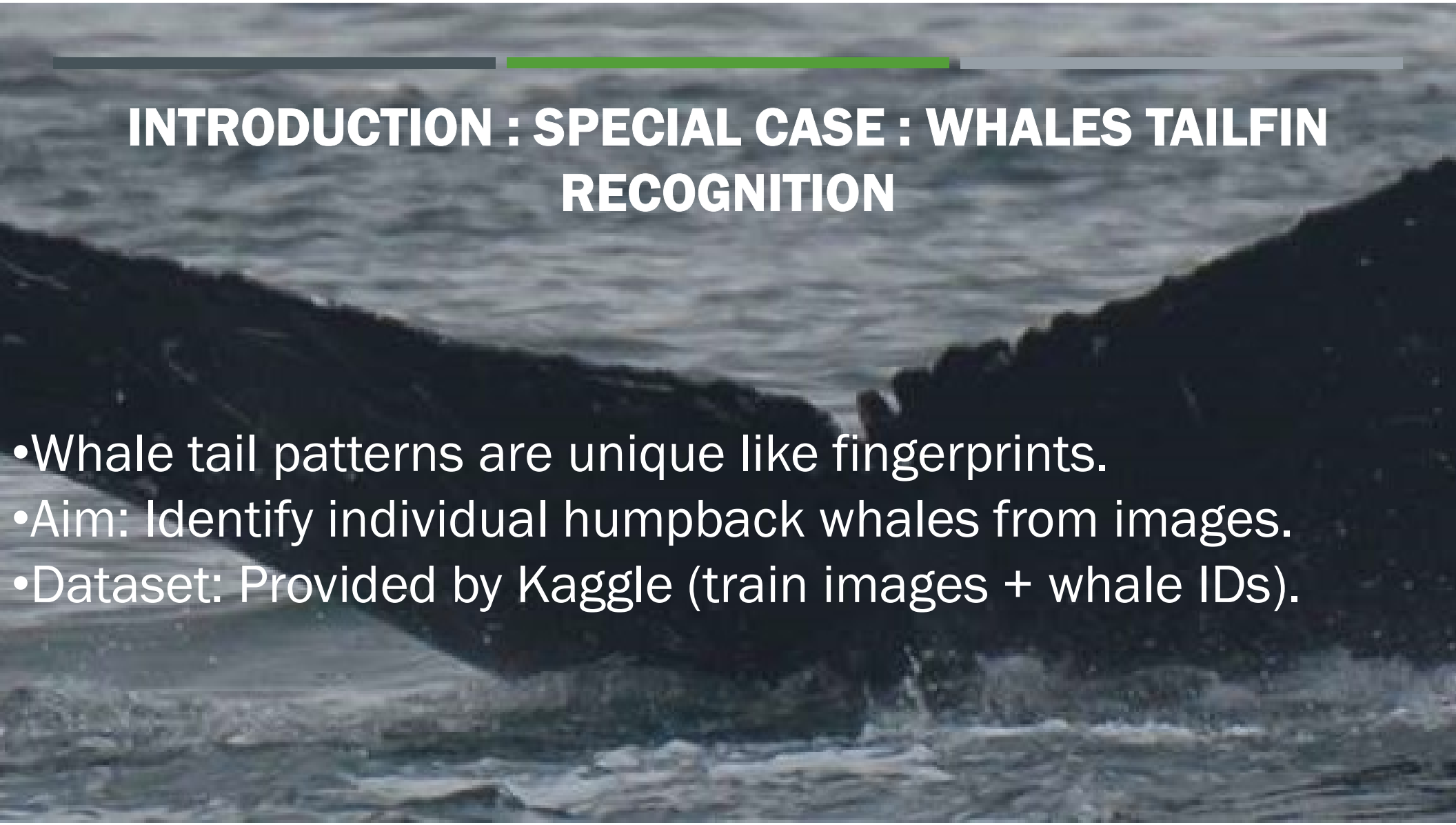
INDIVIDUAL ANIMAL IDENTIFICATION (HUMPBACK WHALES)



DIPANJOY SAHA : MDS202321

SUPERVISOR: PROFF. RAGHAV KULKARNI



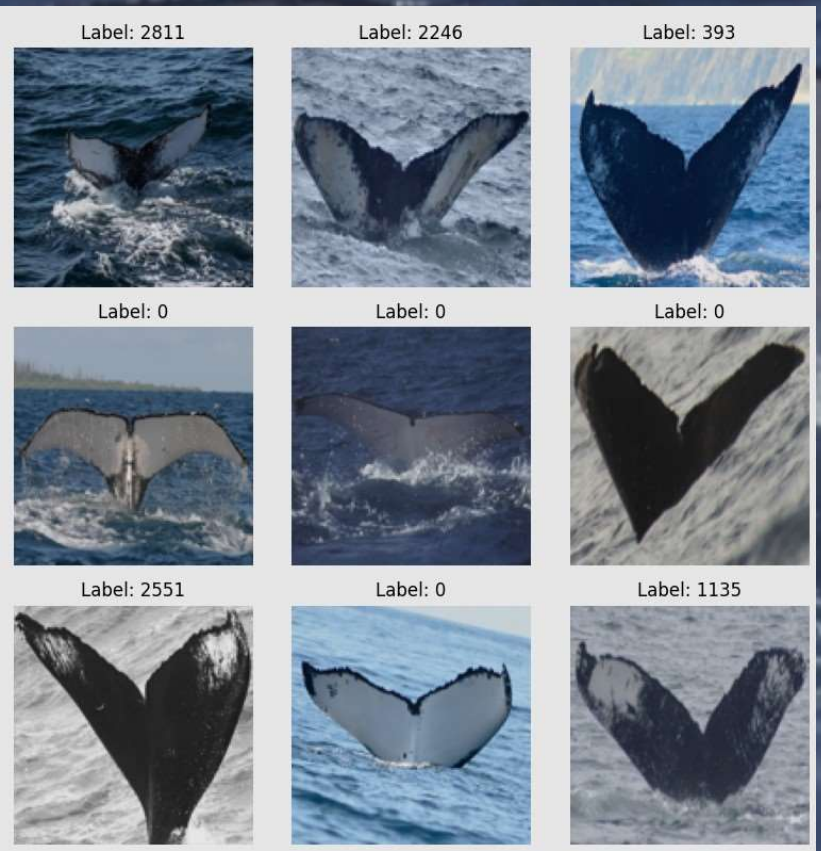
INTRODUCTION : SPECIAL CASE : WHALES TAILFIN RECOGNITION

- Whale tail patterns are unique like fingerprints.
- Aim: Identify individual humpback whales from images.
- Dataset: Provided by Kaggle (train images + whale IDs).

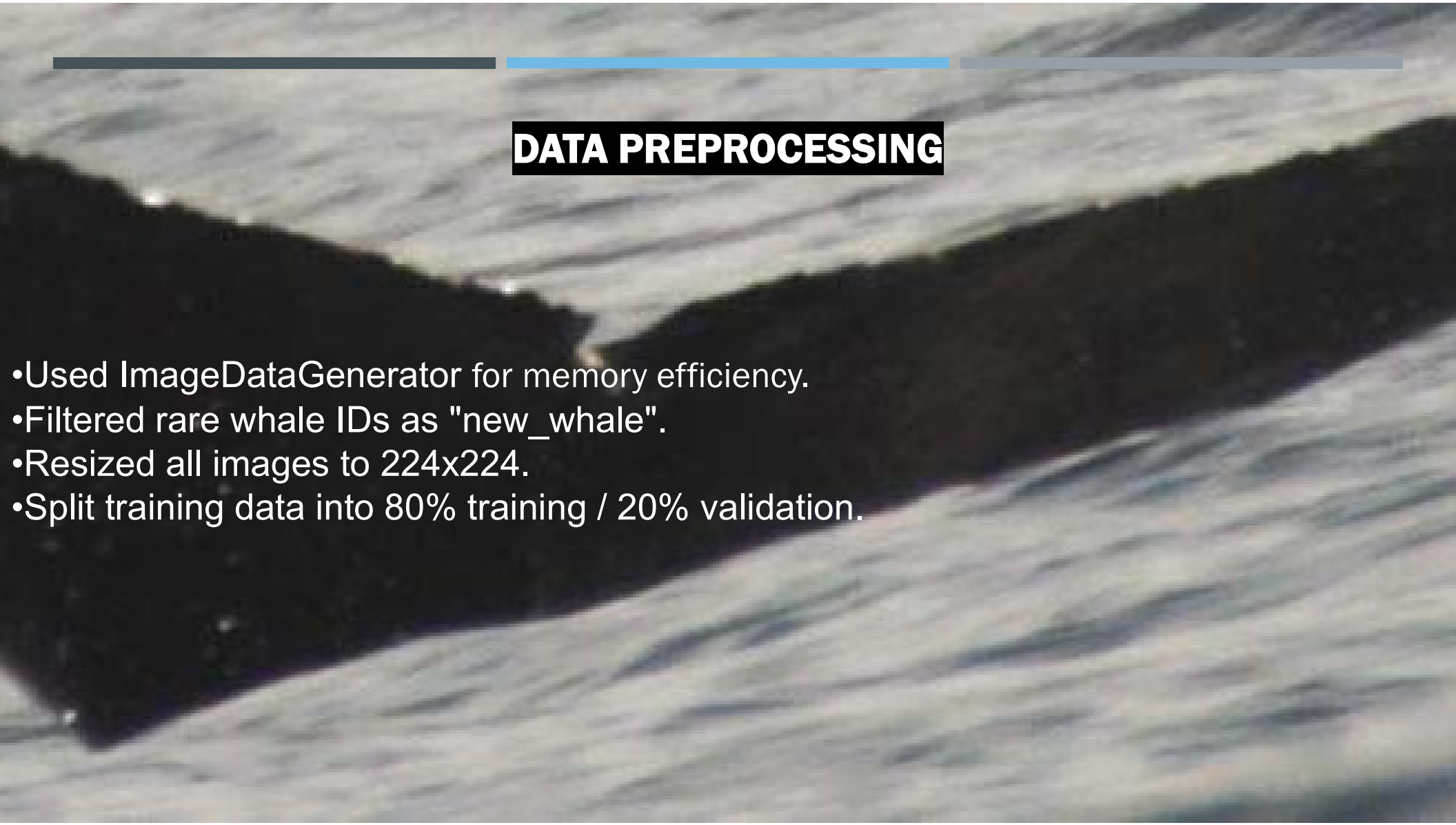
PROBLEM STATEMENT

- Classification problem with highly imbalanced classes.
- Some whale IDs appear only once or twice.
- New/unseen whales also appear during testing.

APPROACH OVERVIEW

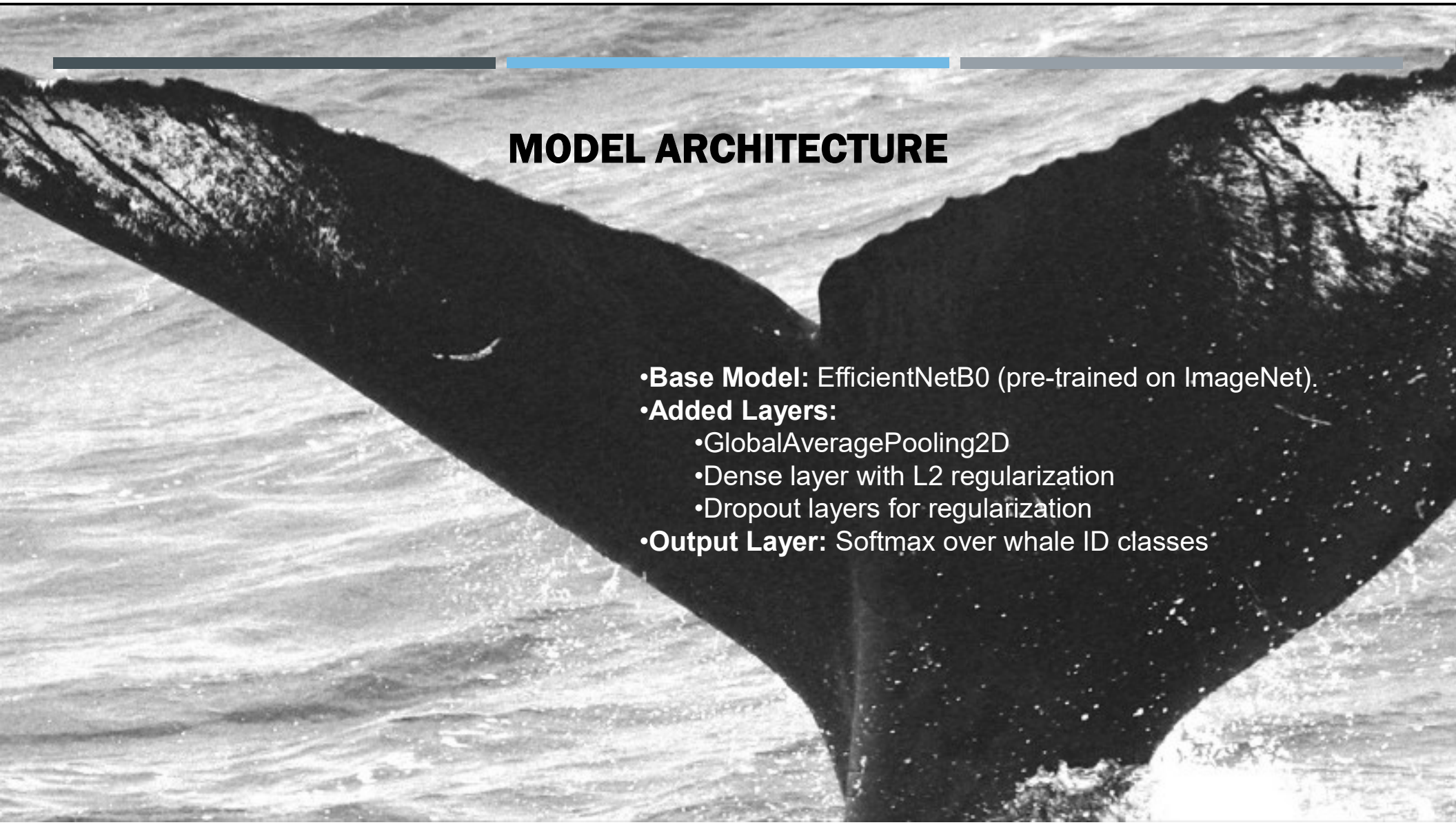


1. Preprocess image data.
2. Handle label encoding and rare classes.
3. Use transfer learning (EfficientNetB0).
4. Apply image augmentation and regularization.
5. Fine-tune the model and predict on test images.



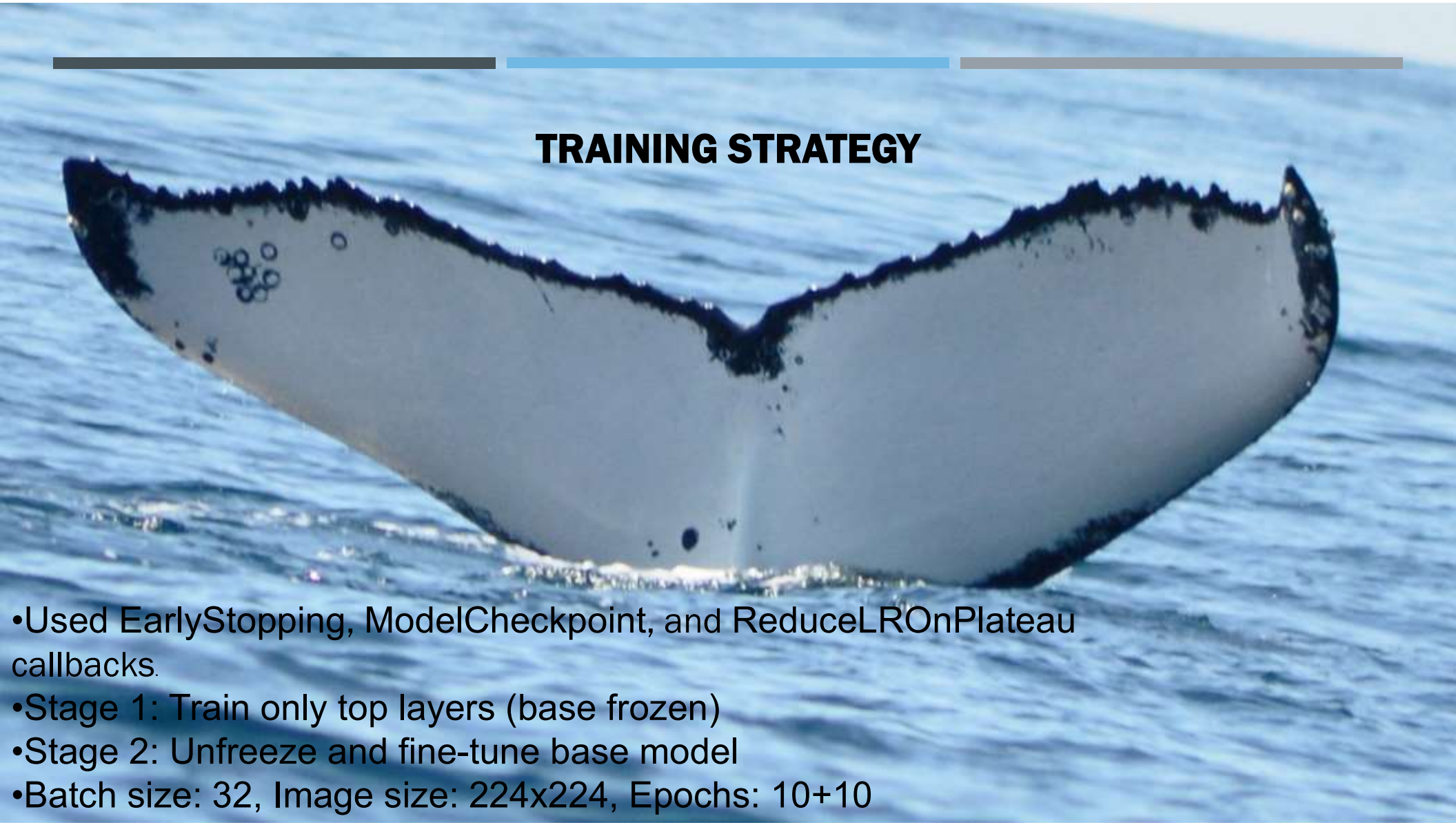
DATA PREPROCESSING

- Used ImageDataGenerator for memory efficiency.
- Filtered rare whale IDs as "new_whale".
- Resized all images to 224x224.
- Split training data into 80% training / 20% validation.



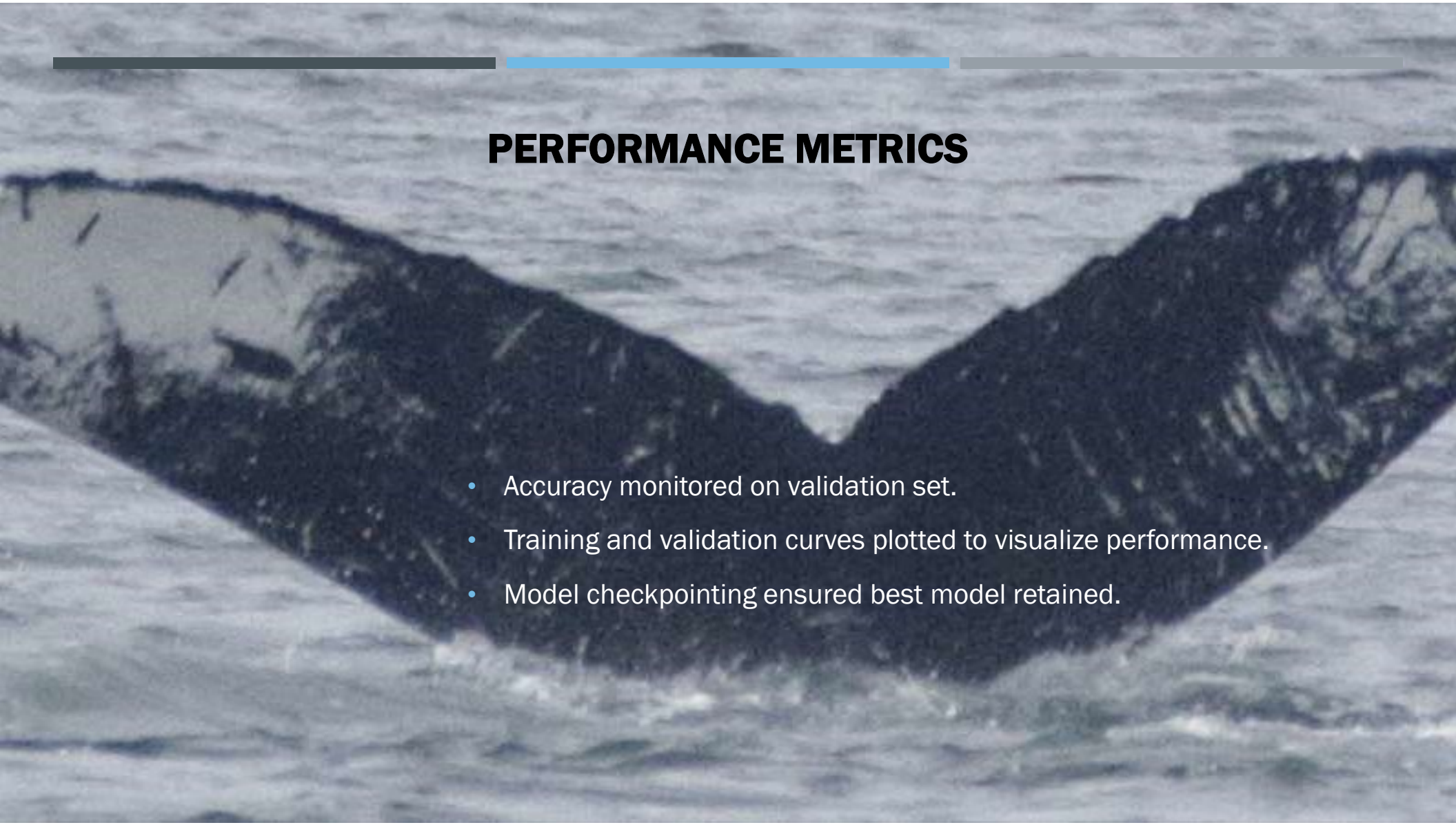
MODEL ARCHITECTURE

- **Base Model:** EfficientNetB0 (pre-trained on ImageNet).
- **Added Layers:**
  - GlobalAveragePooling2D
  - Dense layer with L2 regularization
  - Dropout layers for regularization
- **Output Layer:** Softmax over whale ID classes



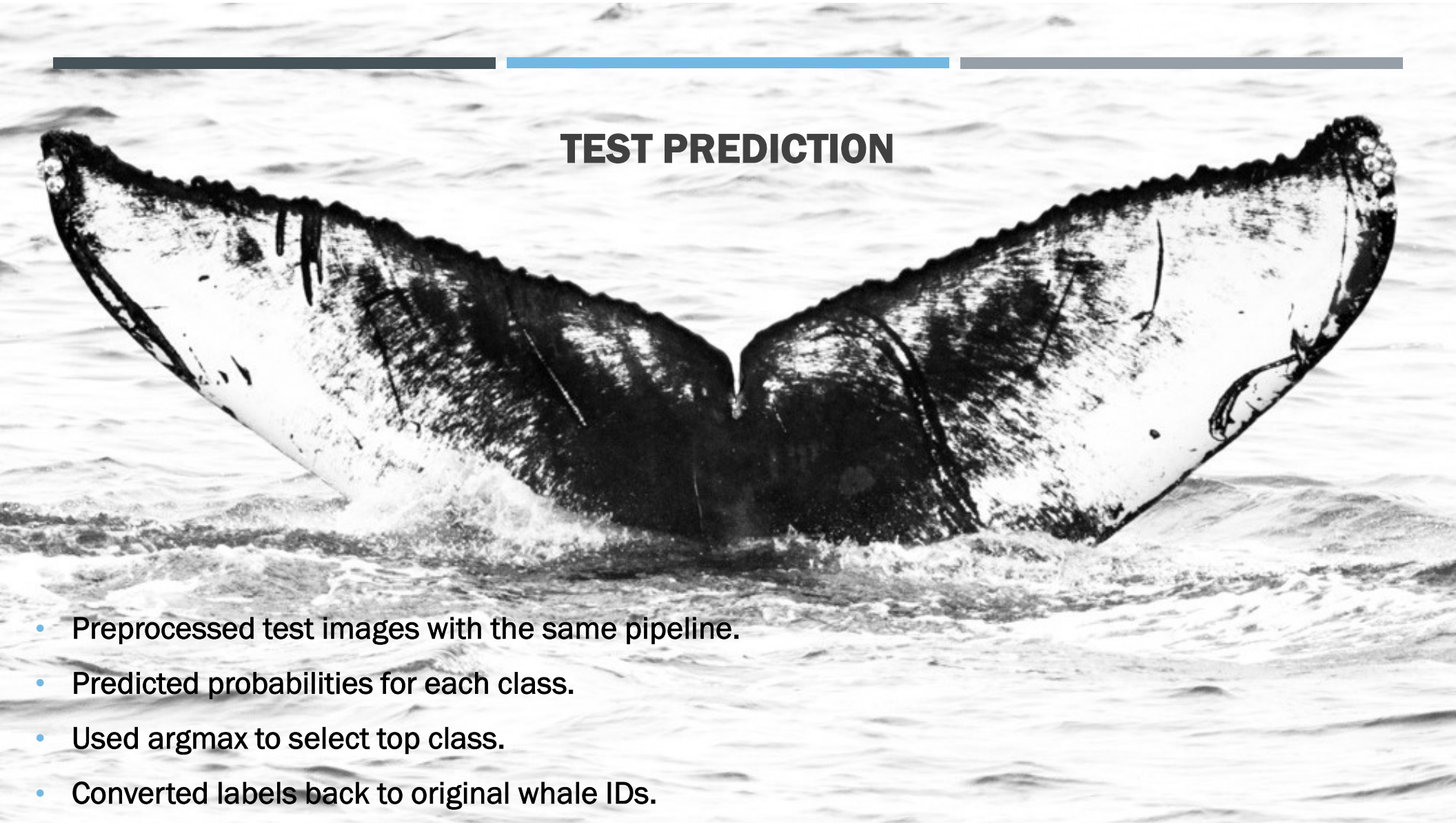
TRAINING STRATEGY

- Used EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau callbacks.
- Stage 1: Train only top layers (base frozen)
- Stage 2: Unfreeze and fine-tune base model
- Batch size: 32, Image size: 224x224, Epochs: 10+10



PERFORMANCE METRICS

- Accuracy monitored on validation set.
- Training and validation curves plotted to visualize performance.
- Model checkpointing ensured best model retained.



TEST PREDICTION

- Preprocessed test images with the same pipeline.
- Predicted probabilities for each class.
- Used argmax to select top class.
- Converted labels back to original whale IDs.

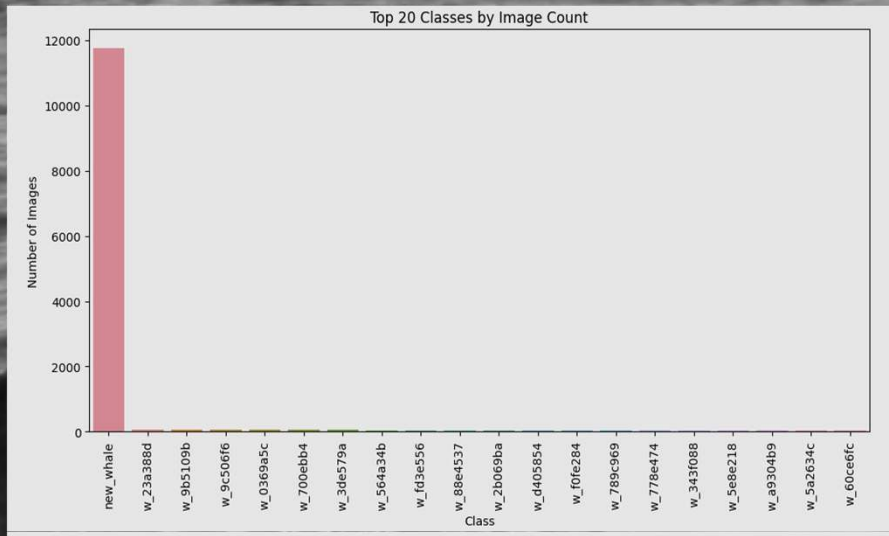
SUBMISSION

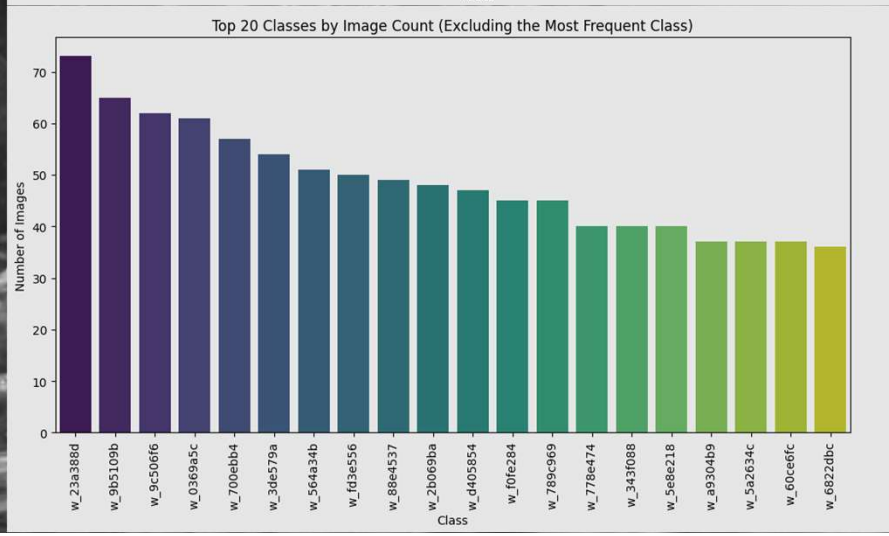
- Final predictions saved in submission.csv.
- Format:

```
Image,Id
img1.jpg,w_1234
img2.jpg,w_5678.
```

RESULTS & REFLECTION

- EfficientNetB0 provided strong feature representations.
- GlobalAveragePooling2D reduced overfitting.
- L2 and dropout improved generalization.
- Pipeline is scalable and GPU-compatible.
- Still accuracy is almost 48% due to technical support.



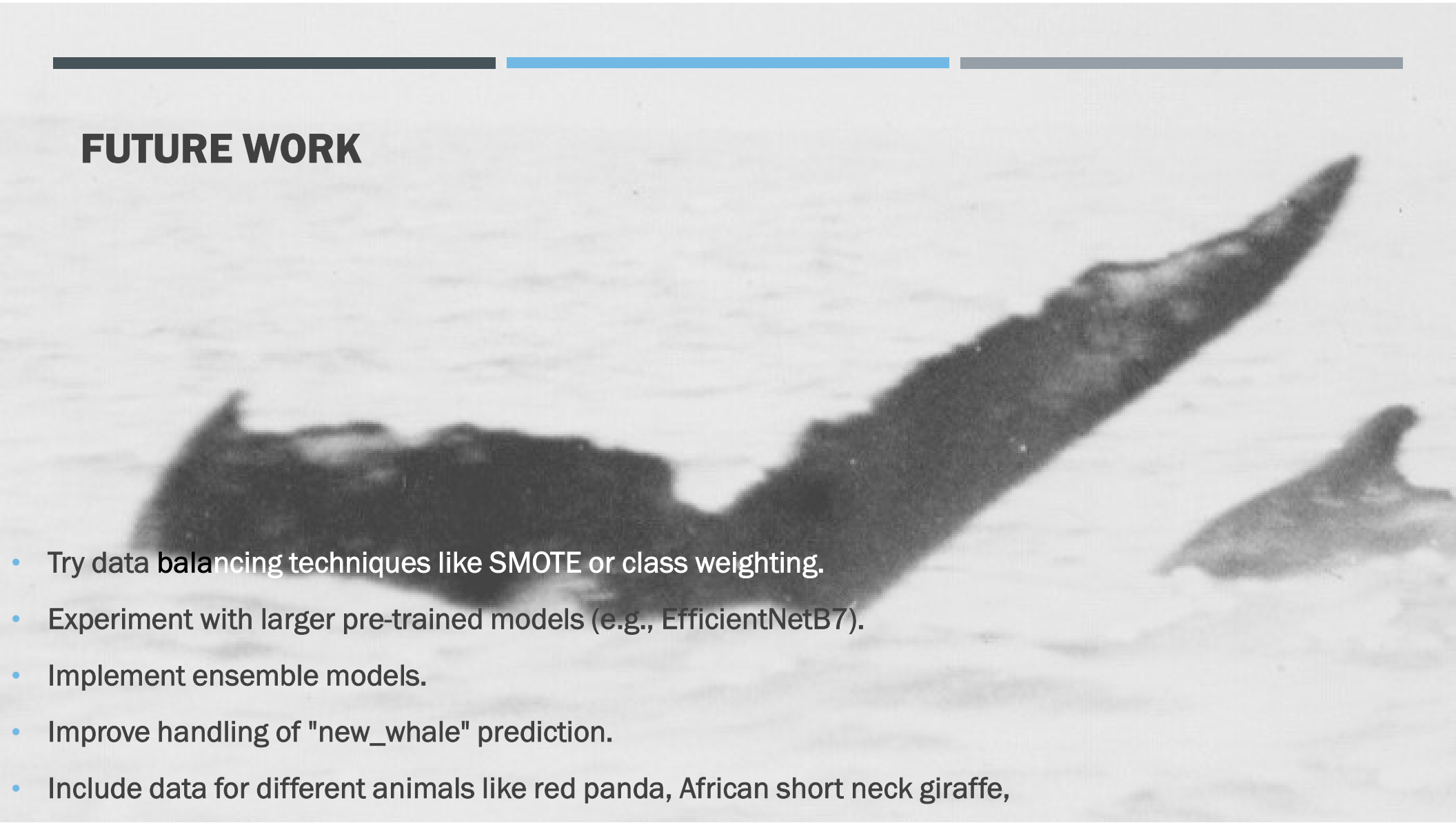


Demo example outcome of training

```
Epoch 2/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4423 -
Epoch 3/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4571 -
Epoch 4/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4642 -
Epoch 5/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4677 -
Epoch 6/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4590 -
Epoch 7/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4614 -
Epoch 8/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4593 -
Epoch 9/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4594 -
Epoch 10/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4626 -
Epoch 11/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4625 -
Epoch 12/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4636 -
Epoch 13/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4611 -
Epoch 14/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4618 -
Epoch 15/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4641 -
Epoch 16/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4637 -
Epoch 17/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4657 -
Epoch 18/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4651 -
Epoch 19/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 7s 22ms/step - accuracy: 0.4662 -
Epoch 20/50
317/317 ━━━━━━━━━━━━━━━━━━━━ 8s 26ms/step - accuracy: 0.4653 -
```

```
635/635 ━━━━━━━━━━━━━━━━━━━━ 286s 399ms/step - accuracy: 0.4538
Epoch 2/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 145s 226ms/step - accuracy: 0.4591
Epoch 3/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 137s 214ms/step - accuracy: 0.4644
Epoch 4/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 136s 213ms/step - accuracy: 0.4604
Epoch 5/10
634/635 ━━━━━━━━━━━━━━━━━━━━ 0s 170ms/step - accuracy: 0.4580 -
Epoch 5: ReduceLROnPlateau reducing learning rate to 0.000200000
635/635 ━━━━━━━━━━━━━━━━━━━━ 137s 213ms/step - accuracy: 0.4581
Epoch 6/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 136s 213ms/step - accuracy: 0.4560
Epoch 7/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 137s 214ms/step - accuracy: 0.4575
Epoch 8/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 140s 218ms/step - accuracy: 0.4603
Epoch 9/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 135s 210ms/step - accuracy: 0.4597
Epoch 10/10
635/635 ━━━━━━━━━━━━━━━━━━━━ 136s 212ms/step - accuracy: 0.4654
```

FUTURE WORK



- Try data balancing techniques like SMOTE or class weighting.
- Experiment with larger pre-trained models (e.g., EfficientNetB7).
- Implement ensemble models.
- Improve handling of "new_whale" prediction.
- Include data for different animals like red panda, African short neck giraffe,

THANK YOU

- GitHub / Notebook link: https://www.kaggle.com/code/dipanjoysaha/whale-identification
- Contact: dipanjoydipanjoysaha@gmail.com