WhaleMatch

A Hack-for-the-Sea project
September 2018

Rich Bean, Eric and Rhianon Brown,
Ted Greene, Isaac Vandor, Chris Zadra

The Problem

- The “SnotBot” is a drone that follows whales and collects samples from the blowhole
- Whale identification is traditionally done visually using the tail, but not all whales show their tails when surfacing
- The “SnotBot” should be able to identify whales in order to more easily match DNA data with visual imagery

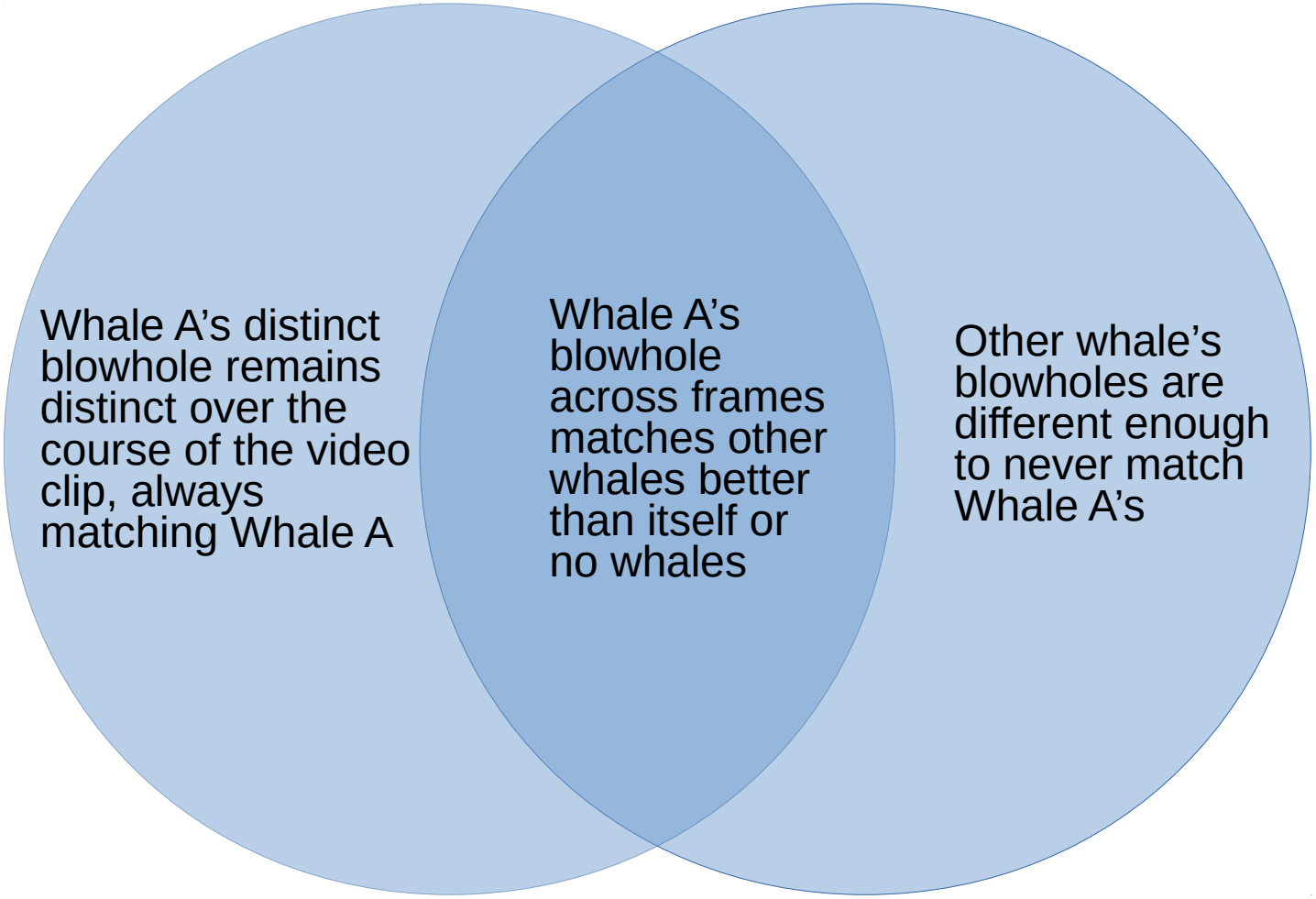
Possible Solution

- Whales have to show their blowholes
- The SnotBot collects video directly above these blowholes
- Is it possible to use this for identification?
  - It’s like identifying a person using only nostrils

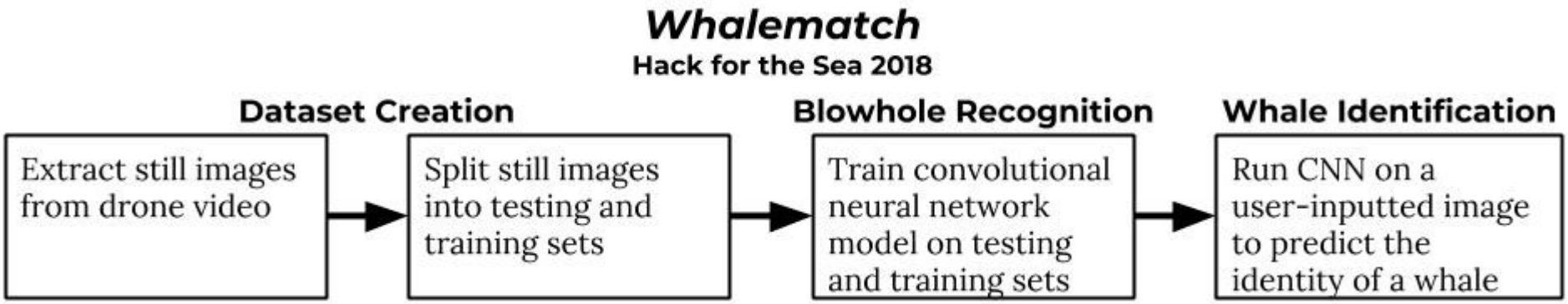
Challenges

- Whales are moving targets
- Spray from blowholes kills visibility
- Blowholes open and close (to keep water out)
- Skin color very similar to surrounding ocean
- Self and cross-subject variability of blowhole shape

Varying Shape Causes Errors



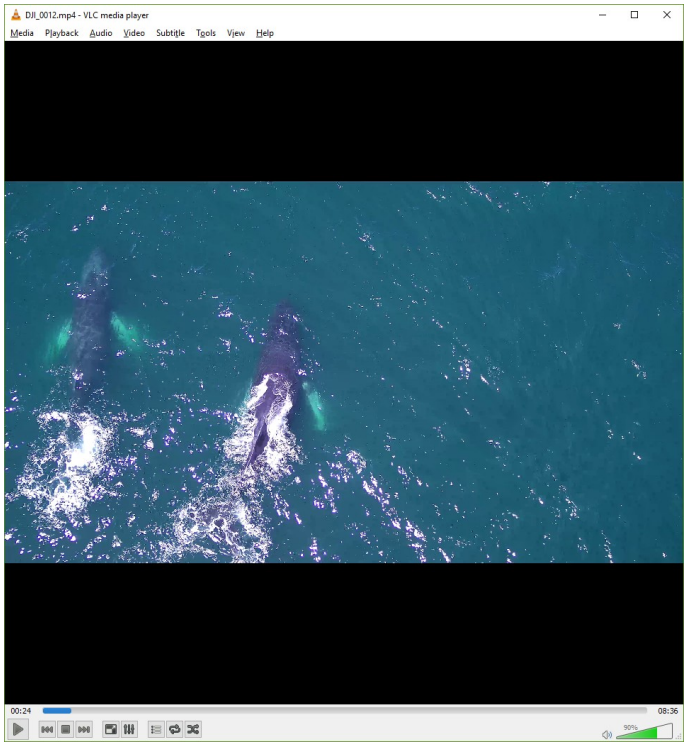
Whalematch Block Diagram



Video Processing

- 3-8 minute long HD videos
- Extract and convert frames

```
 49
-- 50 def process_video(video_stream, prefix="whale"):
 51     """
 52     Process the input video stream, saving frames.
 53
 54     Uses the OpenCV video capture functionality to step through frames of the
 55     input video and save them to disk.
 56
 57     Args:
 58         video_stream:   The video stream, as produced by OpenCV VideoCapture.
 59         prefix:         A string prefix to use for filenames for stills.
 60
 61     Returns:
 62         None.
 63     """
 64     count = 0
 65     success, image = video_stream.read()
 66     while success:
-- 67         imwrite("{}{:05d}.jpg".format(prefix, count), image)
 68         success, image = video_stream.read()
 69         if success:
 70             print('Got frame: {}'.format(count))
 71         count += 1
 72
 73
 74 if __name__ == '__main__':
-- 75     input_filename = parse_command_line()
-- 76     video_stream = VideoCapture(input_filename)
 77     process_video(video_stream)
NORM ~\hfts\whalematch\grabimage.py
```



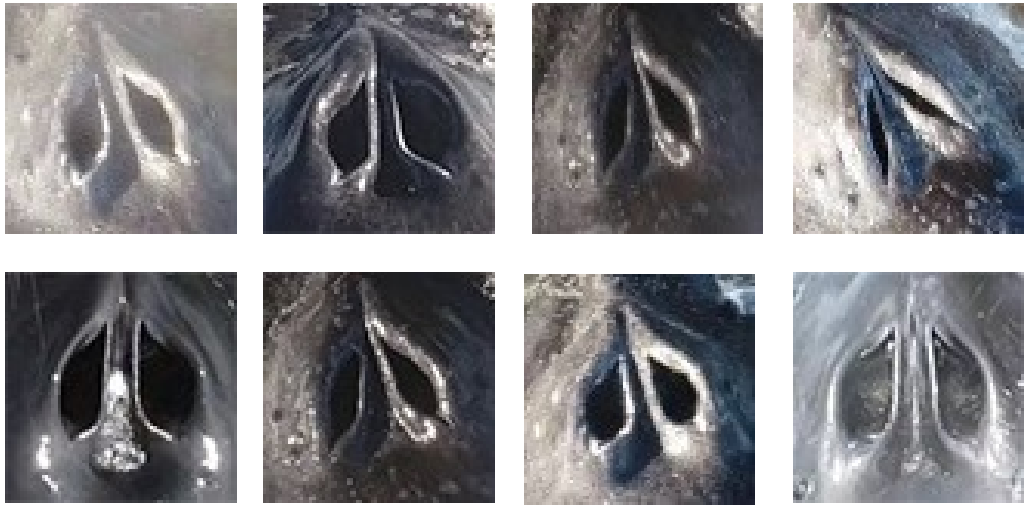
We got around 100K images!

Our (non-representative) Dataset

- Took random 200x200px samples from video stills, resized to 75x75
- Negative sample creation was semi-automated (hand reviewed)
- Blowholes were manually created
- Total Count: 727 negatives, 14 blowholes

Training Data (Negatives)

Testing Data (Blow Holes!)

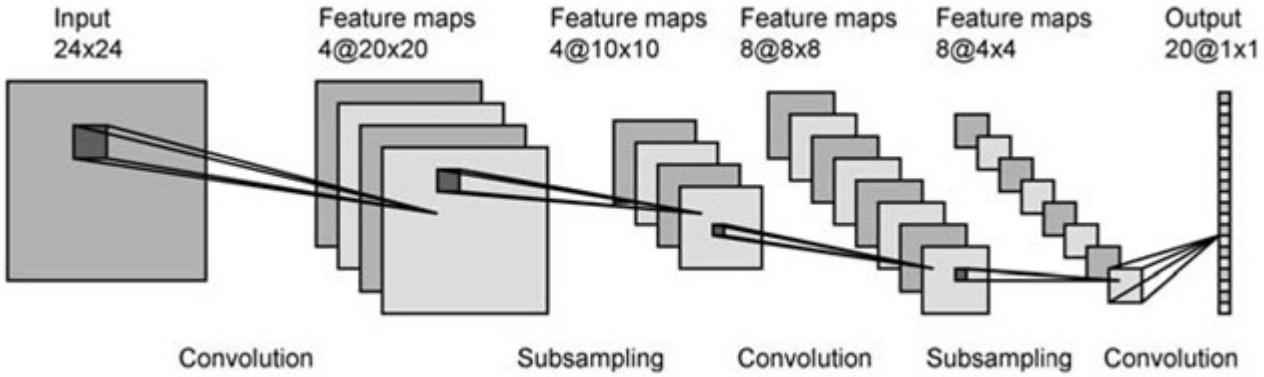


The Machine Learning Approach

- Without doing much pre-processing, can we train a machine learning model to recognize and classify blowholes?
- A similar, but harder, problem to common ML examples like Iris detection, cats vs. dogs detection etc.
- Decided on a convolutional neural network approach to recognize and classify images

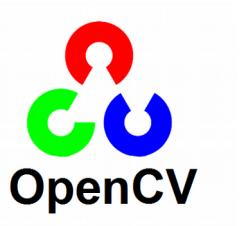
Convolutional Neural Networks

- Feature matching across a set of training data to inform recognition of images in the testing data



- Very useful for image recognition since you can do multiple types of recognition as layers

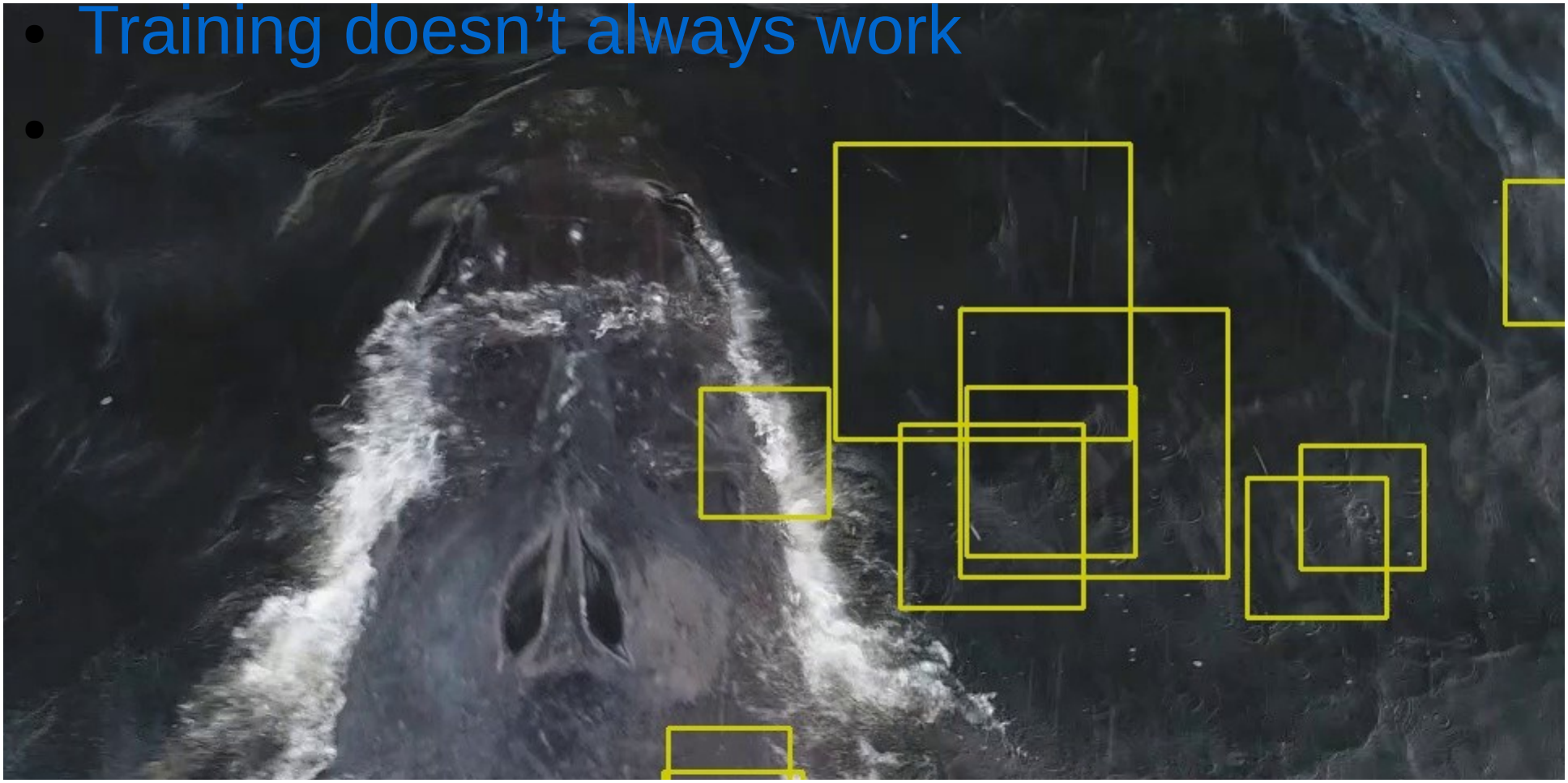
Training

- We tried three different techniques for detection
  - Two using OpenCV 

  - One using Keras 

OpenCV False Positives

- Training doesn’t always work
-



But Not Unreasonable Either!

- Close ups of the failures help explain them

Keras Results

Whalematch: Example Output

Hack for the Sea 2018

Input Image

Output Blowhole

Classified as 'Whale C'

The Road Ahead

- A larger dataset of blowhole images and non-blowhole images
- Tuning the ML model
- A database mapping blowhole images to whale names