Machine Learning Engineering Nanodegree
Capstone Proposal

Russell Razo

April 8th, 2017

Domain Background

Computer vision is an interdisciplinary field that focuses on making computers able to "describe the world that we see in one or more images and to reconstruct its properties, such as shape, illumination, and color distribution" (Szeliski, 2010, p. 5). It started in the early 1970s by some of the early artificial intelligence and robotics pioneers as an attempt to make a computer describe what it saw from a camera linked to it (Boden, 2006).

In the last few years the field of Machine Learning has made tremendous progress on one of the classical problems in computer vision: image recognition; which consists on determining whether an image contains some specific object or feature. In particular, "a new kind of algorithm, convolutional neural network, can achieve reasonable performance on hard visual recognition tasks -- matching or exceeding human performance in some domains"[1].

Nowadays, the applications of computer vision range a wide number of fields and disciplines. A few of the more prominent examples are: in medicine, where it aids medical diagnosis through medical image processing; in the automotive industry, for autonomous driving cars; on manufacturing, where products are automatically inspected for quality control; and in military, to improve battlefield awareness.[2]

Personal Motivation

My personal motivation for working on a project in this area comes from one of my most recent hobbies: photography. The technological advances from recent years on digital cameras and other technologies like smartphones and drones, have made digital photography very accessible and available to the masses, and empowered people to gather visual data in ways and quantities that were not possible before. As a result, I think, digital images have become one of the most rich and important media of the present, not just because they can convey art, but also because they hold information of everyday life.

---

[1] Image Recognition. (2017, March 8). Retrieved from https://www.tensorflow.org/tutorials/image_recognition

[2] Computer vision. (2017, April 7). In *Wikipedia, The Free Encyclopedia*. Retrieved 06:26, April 9, 2017, from https://en.wikipedia.org/w/index.php?title=Computer_vision&oldid=774274838

Therefore, this is a perfect opportunity for me to learn more about machine learning, and an initial step towards bringing it together with one of my passions.

Problem Statement

In this project, I will tackle a problem presented by the National Oceanic and Atmospheric Administration through the Kaggle competition *NOAA Fisheries Steller Sea Lion Population Count*, which can be found at this link:

https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count.

Description

Steller sea lions in the western Aleutian Islands have declined 94 percent in the last 30 years. The endangered western population, found in the North Pacific, are the focus of conservation efforts which require annual population counts. Currently, it takes biologists up to four months to count sea lions from the thousands of images NOAA Fisheries collects each year. Once individual counts are conducted, the tallies must be reconciled to confirm their reliability. The results of these counts are time-sensitive.[3]

Problem assessment

Having understood the challenge posed by the competition, and after taking a glance of some sample images of the dataset, I can identify two major tasks:

1. Recognizing instances of sea lions on a given image
2. Classifying a single instance into one of the five possible categories: adult males, subadult males, adult females, juveniles and pups

As such, this problem is a perfect candidate to be solved by using computer vision techniques and deep learning models.

Datasets and Inputs

The dataset I will be using is the dataset provided for the competition, which can be found at this url: https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count/data

The dataset consists of approximately 96 gigabytes of aerial photographs, all of them in JPG format and RGB color space, but with different sizes and different dimensions. The dataset has already been split into the following items:

---

[3] NOAA Fisheries Steller Sea Lion Population Count. (2017, April 9). Retrieved from: https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count#description

- Train/train.csv: a list of ground-truth counts for each train_id
- Train/*.jpg: a set of training images, with each filename corresponding to a train_id
- TrainDotted/*.jpg: copies of the images in the Train folder, but with colored dots placed over the animals. The color scheme of the dots is:
    - red: adult males
    - magenta: subadult males
    - brown: adult females
    - blue: juveniles
    - green: pups
- Test/*.jpg: a set of test images, with each filename corresponding to a test_id

While the TrainDotted folder has been provided to assist in locating individual sea lions, both the Train and TrainDotted folders will be used to train the models, and the Test folder to validate and evaluate the model performance.

Scaled down non-dotted training image

Solution Statement

As briefly mentioned in the problem statement section, this problem is an special case of image classification and segmentation: my algorithm must be able to find, on each image, specific regions where there is potentially a sea lion, and then tell what type of sea lion it is with some degree of confidence or probability.

For this reason, I intend to use deep learning to solve the task of image classification, more precisely a convolutional neural network, as it has proven to yield satisfactory results in this kind of problem. On the other hand, I have very limited knowledge on how to approach the task of finding portions of an image that potentially contain an object of interest, in this case a sea lion; but the research I've done so far has shown me it is possible to do it by a combination of a visual description algorithm (like Histogram of Oriented Gradients) and Support Vector Machines, or through convolutional neural networks as well.

Benchmark Model

Although there are image recognition APIs open to the public, like Rekognition from Amazon and Cloud Vision API from Google, the requirements of this problem are different than those of determining the class of an image or finding the most suitable labels from the contents in the image, therefore, the benchmarks or reference points I will use will be intrinsic to the problem and it's resources:

- **train.csv file**: For the task of doing image segmentation and finding individual sea lions on a whole image, the ground-truth counts for each train_id, that are provided on this file as part of the data resources, will be used.
- **Random agent**: This is the most basic benchmark for the task of classifying individual sea lions. Assuming image segments of individual sea lions are given to a model, it must be able to do better than guessing their respective class, that is, the model should be able to tell with more than 16% accuracy what kind of sea lion it is, or whether it is not a sea lion at all.
- **Competition submission**: Another possible benchmark for the whole solution can be through submitting my results to the competition platform. Besides evaluating the performance of my solution, the platform can tell me how well is my solution doing compared to other competitors.

Evaluation Metrics

- **Prediction accuracy**: I will use accuracy as the main metric to evaluate the model that classifies the sea lions.
- **F1 and F2 scores**: I will employ F1 and F2 scores as well, since they can help me identify what kind of sea lion does the model struggle to predict the most, along with a confusion matrix.
- **Root mean squared error (RMSE)**: I will evaluate the performance of the whole model (count sea-lions by type) by the RMSE from the human-labelled ground truth, averaged over the columns. One approach I can follow is to split the training set into training and testing subsets in order to take advantage of the file *train.csv* that is provided along with the dataset. Another option is to submit my results over the original testing set to the competition platform, which will also give me the RMSE score.

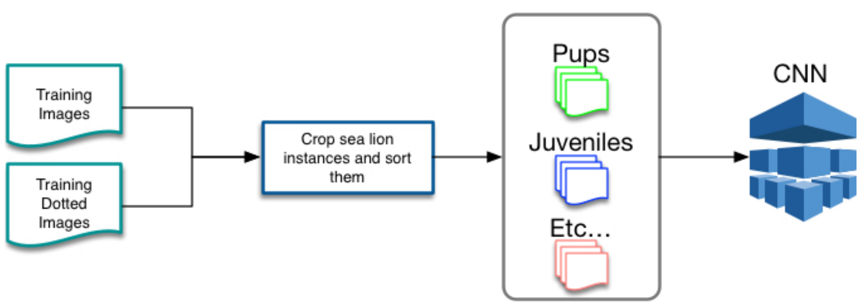
Project Design

The first step in my project will be to extract smaller images with individual sea lions from the images in the training set, for which the dotted version of the images will be very valuable. The most basic approach I can think of to achieve this is by creating an estimated area around each dot in the dotted image and "cut" it from the original image; another way would be to use a visual description algorithm around each dot, although I definitely need to do more research to figure out the best method. Regardless, the goal of this task is to generate single instances of sea lion images and separate them into their corresponding class. Also, it's important to mention a few challenges that become apparent at first glance:

1. Sea lions are of different sizes
2. Aerial photos were not taken from the same distance
3. Due to the proximity of the sea lions, they can potentially overlap after cropping a rectangular or square area around a dot

Fragment of a dotted image. Areas with single sea lions are created around each dot

Once I have extracted all the possible images of single sea lions from each photo in the training set, and separated them by type, I can train a convolutional neural network model to be able to classify sea lion instances.



Steps to create a convolutional neural network classifier for the different sea lion classes

As the last major step, I will create the image segmentation component, taking advantage of the previously created CNN classifier. Although I must perform more extensive research, I've already identified a few algorithms and ways to carry out this task, one of the most attractive techniques I've found is described on the next paragraph:

> "We can do segmentation using an existing Convolutional Neural Network by applying it in a Fully Convolutional manner. This is done by casting the Fully Connected Layers of a network into Convolutional – this way we can input image of any size and get segmentation of lower resolution due to max-pooling layers that are used in network." (Pakhomov, 2016)

Another algorithms to consider are color-based image segmentation and thresholding. Both could be useful to subtract the background of the images, in this case the sea, and probably take advantage of the difference in colors of the sea lions and the rest of the elements in the images.

Programming Language and Libraries

These are some of the main libraries I intend to use:

- Python 2.7
- scikit-learn: Open source machine learning library
- scikit-image: Open source image processing library
- OpenCV: Open source computer vision library
- Pandas: Library for for data manipulation and analysis
- TensorFlow: Open source software libraries for deep learning
- TF-Slim: Lightweight wrapper library around TensorFlow

References

Boden, M. A. (2006). *Mind as machine: A history of cognitive science*. Clarendon Press.

Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2014). Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv: 1412.7062*.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3431-3440).

Nielsen, M. A. (2015). *Neural networks and deep learning*. Determination Press.

Pakhomov, Daniil. (2016). *Image Classification and Segmentation with Tensorflow and TF-Slim*. Retrieved Apr. 10, 2017 from http://warmspringwinds.github.io/tensorflow/tf-slim/2016/10/30/image-classification-and-segmentation-using-tensorflow-and-tf-slim/

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., ... & Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1386-1393).

Image Recognition. (2017, March 8). Retrieved from https://www.tensorflow.org/tutorials/image_recognition

Computer vision. (2017, April 7). In Wikipedia, The Free Encyclopedia. Retrieved 06:26, April 9, 2017, from https://en.wikipedia.org/w/index.php?title=Computer_vision&oldid=774274838