Machine Learning Engineer Nanodegree

Capstone Proposal

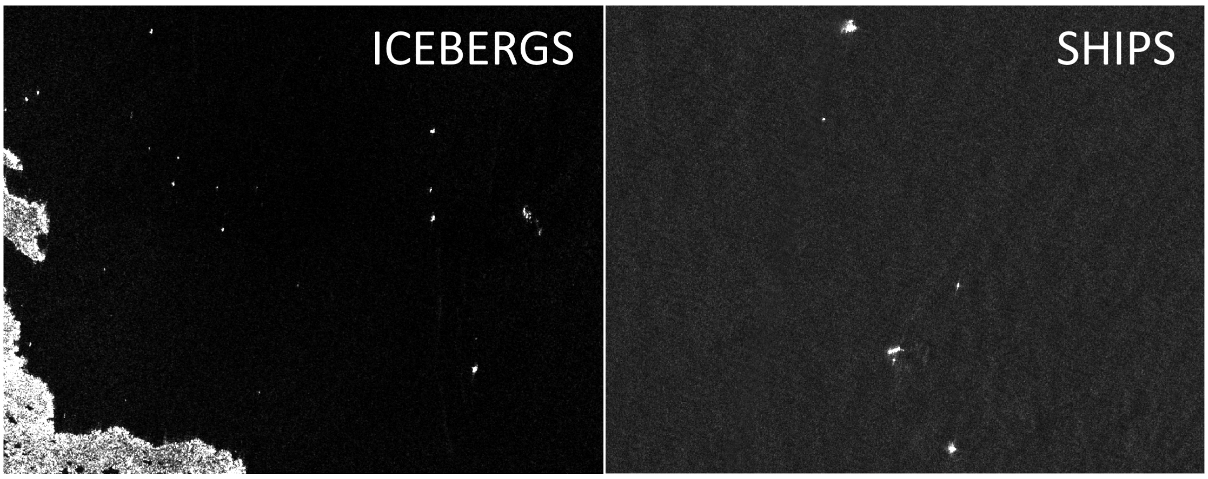
Satyanarayan Bhanja
October 28th, 2017

Proposal

Domain Background

The remote sensing systems used to detect icebergs are housed on satellites over 600 kilometers above the Earth. The Sentinel-1 satellite constellation is used to monitor Land and Ocean. Orbiting 14 times a day, the satellite captures images of the Earth's surface at a given location, at a given instant in time. The C-Band radar operates at a frequency that "sees" through darkness, rain, cloud and even fog. Since it emits it's own energy source it can capture images day or night.
Satellite radar works in much the same way as blips on a ship or aircraft radar. It bounces a signal off an object and records the echo, then that data is translated into an image. An object will appear as a bright spot because it reflects more radar energy than its surroundings, but strong echoes can come from anything solid - land, islands, sea ice, as well as icebergs and ships. The energy reflected back to the radar is referred to as backscatter.

When the radar detects a object, it can't tell an iceberg from a ship or any other solid object. The object needs to be analyzed for certain characteristics - shape, size and brightness - to find that out. The area surrounding the object, in this case ocean, can also be analyzed or modeled. Many things affect the backscatter of the ocean or background area. High winds will generate a brighter background. Conversely, low winds will generate a darker background.



Academic paper where machine learning is applied to this problem:

http://elib.dlr.de/99079/2/2016_BENTES_Frost_Velotto_Tings_EUSAR_FP.pdf
(http://elib.dlr.de/99079/2/2016_BENTES_Frost_Velotto_Tings_EUSAR_FP.pdf)
The researchers from German aerospace center(The authors of this paper) got an f1 score of 98% using a CNN model.

Kaggle Datasets link:
https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/train.json.7z
(https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/train.json.7z)
https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/test.json.7z
(https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/test.json.7z)

Problem Statement

Drifting icebergs present threats to navigation and activities in areas such as offshore of the East Coast of Canada.
Currently, many institutions and companies use aerial reconnaissance and shore-based support to monitor environmental conditions and assess risks from icebergs. However, in remote areas with particularly harsh weather, these methods are not feasible, and the only viable monitoring option is via satellite.

The problem is to predict whether an image contains a ship or an iceberg. This is a binary classification challenge.
The output is to predict the probability of an iceberg in the images.

Datasets and Inputs

Kaggle Datasets link:
https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/train.json.7z
(https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/train.json.7z)
https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/test.json.7z
(https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/download/test.json.7z)

The fields of training data as below,

- id - the id of the image
- band_1, band_2 - the flattened image data. Each band has 75x75 pixel values in the list, so the list has 5625 elements.
- inc_angle - the incidence angle of which the image was taken.
- is_iceberg - the target variable, set to 1 if it is an iceberg, and 0 if it is a ship.

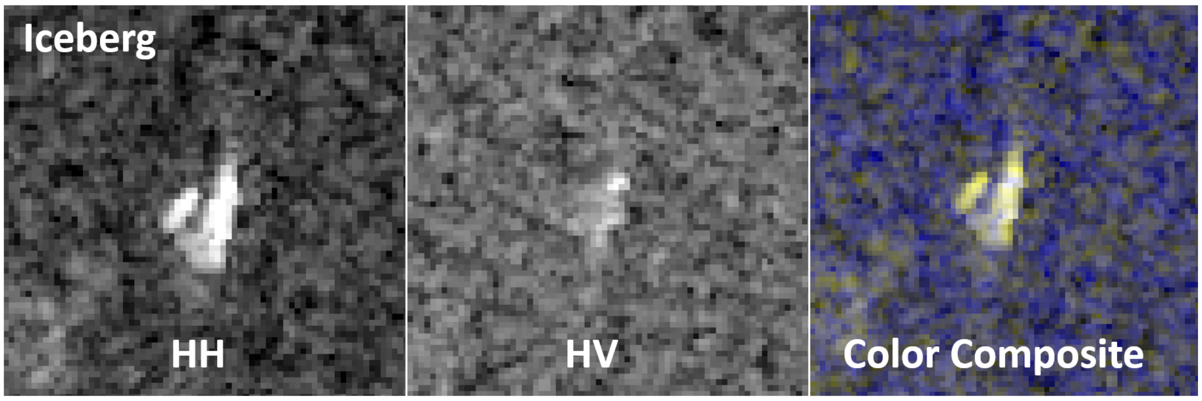
- The testdata has all the fields except is_iceberg column.
- There are total 1604 training data and 8424 test data.
- The training dataset has 53.05% "ships" and 46.94% "ice-bergs". The classes are close to balanced.
- The training dataset will be devided into 75% training and 25% testing data,

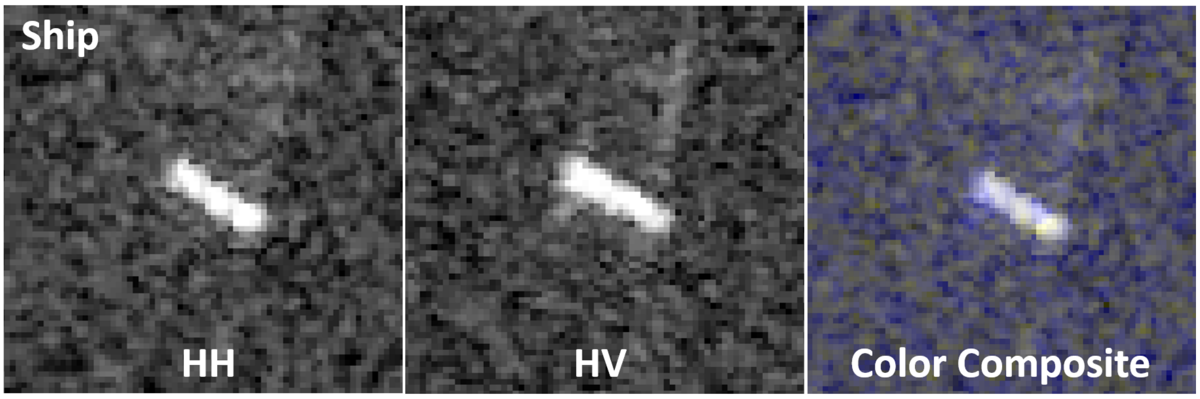
The data looks like as below,

	band_1	band_2	id	inc_angle	is_iceberg
0	[-27.878360999999998, -27.15416, -28.668615, -...	[-27.154118, -29.537888, -31.0306, -32.190483,...	dfd5f913	43.9239	0
1	[-12.242375, -14.920304999999999, -14.920363, ...	[-31.506321, -27.984554, -26.645678, -23.76760...	e25388fd	38.1562	0
2	[-24.603676, -24.603714, -24.871029, -23.15277...	[-24.870956, -24.092632, -20.653963, -19.41104...	58b2aaa0	45.2859	1
3	[-22.454607, -23.082819, -23.998013, -23.99805...	[-27.889421, -27.519794, -27.165262, -29.10350...	4cfc3a18	43.8306	0
4	[-26.006956, -23.164886, -23.164886, -26.89116...	[-27.206915, -30.259186, -30.259186, -23.16495...	271f93f4	35.6256	0

- The band_1 and band_2 are the HH and HV bands.
- HH (transmit/receive horizontally) and HV (transmit horizontally and receive vertically).

Sample iceberg and ship images are as in kaggle below,





Solution Statement

- Create a third band by averaging the first and second band. Now, the images are of size (75x75x3).
- The classification algorithm to be used is Convolutional neural netwroks.
- Stacked layers of conv, maxpooling, dropout to be used for pre-processing or feature generation.
- Then fit the model and validate the results.

Benchmark Model

The benchmark model will be a simple CNN classifier trained on the train data. Then I will try to improve the model performance by adding more layers and use this model as a benchmark to test performance of new model.

Evaluation Metrics

The avaluation metric to be used is **f1-score** to quantify the performance of both the benchmark model and the solution model.

TP = true positive
TN = true negative
FP = false positive
FN = false negative

$\text{precision} = TP/(TP + FP)$
$\text{recall} = TP/(TP+TN)$

$\text{f1\_score} = 2 * (\text{precision} * \text{recall})/(\text{precision} + \text{recall})$

Project Design

The project workflow as below,

- Read the datasets and remove NAs
- Create a third band by averaging the first and second band. Now, the images are of size (75x75x3).
- pre-process the images - Using image augmentation techniques.
- train-test split the data into 75:25 for validation.
- Use CNN with Keras to classify the images.
-
    - The layers I want to use many layers of (conv2d, maxpooling, droput),dense stack to get features from images.
- The features extracted from CNN will be passed to a fully connected layer for classification.
- train the model using keras.
- Then validate the model, on the validation set.
- Predict on the test data set and upload in kaggle to check public score.
- The solution file should be of format,
-
    - id - the id of the image
-
    - is_iceberg - your predicted probability that this image is iceberg.