Iceberg Detection using Sentinel-1 Satellite Images

Matthew Lee

December 28, 2017

1 Domain Background

Icebergs present a significant danger to shipping, especially in adverse weather conditions. Identifying Icebergs and distinguishing them from ships is therefore vital for safe sea transport. This project aims to improve the detection of icebergs using satellite imagery. This project is based on the Statoil/C-CORE Iceberg Classifier Challenge, posed on kaggle.com [4]. To mitigate risks caused by icebergs, many operators use shore based techniques and aerial reconnaissance. However, in remote locations or during harsh weather conditions the icebergs may not be visible for aerial reconnaissance or there may not be an airfield nearby to provide coverage. This is where satellite imagery can help. The Sentinel-1 satellites use imaging techniques that are able to penetrate through clouds and can operate at night as they are their own lightsource. However, telling ships apart from icebergs is difficult as the images do not have colour and have low resolution. The current C-Core system used is the proprietary Iceberg Detection Software (IDS). As it is proprietary I was not able to find out how the existing software works, but there are many other examples of satellite-based iceberg detection in the literature [1, 2].

The radar takes two images, with different polarisations, Horizontal-Horizontal(HH) and Horizontal-Vertical(HV). For HH the image is formed by transmitting light with a horizontal polarization and receiving light with a horizontal polarisation. For HV the light is transmitted with horizontal polarisation and received on vertical polarisation. The polarisation of light backscattered to the satellite depends on the material, polarisation of transmitted light and the angle of the reflecting surface. This imaging technique is referred to as dual-polarisation. Dual-polarisation has proven to improve the classification of ice Vs water, therefore I expect it will improve classification of iceberg vs ship [3, 9].

I was interested in this project because of my previous experience with polarisation and satellite imagery from my Physics degree. I'm really interested in the application of satellites and am excited to see what can be achieved by combining machine learning with satellite technology. I think improvements to data analysis from satellites could be very valuable given the considerable investment involved to transport a satellite into space and our inability to upgrade existing satellite hardware once transported. This means that work to upgrade the software, to extract more value from these satellites is especially important.

2 Problem Statement

The problem to be solved is one of binary image classification. Does a given Sentinel-1 image contain an iceberg? The model that solves this problem must have a high accuracy in order to be trusted to handle the important task of guiding ships. The metric used for this is discussed in the Evaluation Metrics section. The problem is repeatable because there is constant shipping across the atlantic, with over 10 billion tonnes being shipped globally each year. [8]

3 Datasets and Inputs

The dataset is provided in a JSON list with the following fields for each entry in the list:

- ID: The ID of the image

- band_1: The flattened image data in a list. Images are 75x75 so the flattened list has 5625 elements. band_1 is for the HH polarisation.

- band_2: The same as band_1 except this image is of the HV polarisation.

- inc_angle: The incidence angle that the satellite was at when the images were taken. 133 of the images have NA for incidence angle so some pre-processing may be needed.

- is_iceberg: This field only exists for the training set and indicates whether the object is an iceberg.

There are 1604 samples in the training set and 8424 in the test set. The test set includes some machine generated images to prevent hand scoring.

Each band provides an image of the object, interpretation of this data will form the main process of the classification model. It has been found that for the purposes of ice identification, using a single band rather than interpretting both, results in much poorer results [3,9]. Therefore it is likely that the strongest models will use both bands for classification.

The incidence angle describes the position of the satellite when the image was taken. The angle is defined as the angle between the the position of the satellite and the vector perpendicular to the receiving surface. i.e. if the incidence angle is 0^0 then the satellite is directly overhead. The angle affects the backscattering behaviour of the target. The change in backscattering as angle changes is different on the different polarisations, therefore it is essential that the model takes this into account when interpretting the bands. For example: Larger incidence angles reduce backscattering from the ocean clutter and have increased the probability of iceberg detection for previous models [3]

4 Solution Statement

I intend to use a convolutional neural network to solve this problem. As I do not have access to vast computing resources, will use transfer learning models. I will compare the performance of the VGG16, ResNet50 and Inception-ResNet V2 models. After measuring their initial performance I will commit to one model and tune it to improve performance.

In addition to the pre-train convolutions, I will need to pass the incidence angle to the fully connected layers. As the relationship between backscattering and incidence angle is trigonometric, I will take the sine of the angle.

In order to provide sufficient training time and computing resource, I will run the model on an AWS EC2 server. I will use a callback function to save the

weights and structure of the model, so I can experiment without losing progress towards higher accuracy.

5 Benchmark model

At the time of writing the scores on the leaderboard show a top score log loss of 0.0958. To be realistic, I am not expecting my model to outperform this as the leaderboard will constantly improve and there may be several people collaborating in each Kaggle team. A more realistic benchmark would be the median of all the entries which has a log loss of 0.2128 [4]. The closest benchmark available outside of Kaggle are the results of the German Aerospace Center [1] who performed ship-iceberg discrimination using high resolution images from the TerraSAR-X satellite. This achieved recall of 100% of icebergs with an F1-score of 98%. However this was using high resolution images on a different satellite so is not an exact equivalent. The lower resolution of the Sentinel-1 images means we can expect this project to have poorer results than the TerraSAR-X benchmark.

6 Evaluation Metrics

The Kaggle competition measures the performance of a given model using log loss. As there are only two classifications for an image; has an iceberg or does not have an iceberg, the classification is binary and the log loss equation is simple. The log loss equation: [5]

$$logloss = -\frac{1}{N} \sum_{i=1}^{N} y_{i1} \ln(p_{i1}) + y_{i2} \ln(p_{i2})$$

Where N is the number of images classified, y_{i1} is 1 if image contains an iceberg and 0 otherwise, y_{i2} is 0 if image contains an iceberg and 1 otherwise, p_{i1} is the predicted probability of image i containing an iceberg, p_{i2} is the predicted probability of image i not containing an iceberg.

This equation sums the natural logarithms of the models output probabilities for incorrect predictions and divides by the total number of predictions. This means that the log loss is reduced by improving accuracy. A perfect model will have a log loss of 0. This is interesting as I would have expected the competition to have valued recall most, as a ship wrongly classified as an iceberg poses no danger to another ship, where as an iceberg wrongly classified as a ship could be extremely dangerous. For the purposes of this project I will use Kaggle's log loss equation as the main metric, however I will still measure recall as it will be an important metric in real world scenarios.

7   Project Design

To start with the data will need to cleansed. Fortunately, as this is a Kaggle competition the data is mostly in a sanitized form. The notable exception is that there are 133 training data that have 'na' for their incidence angle. If there were more training data, I would consider deleting these data entirely, however with only 1604 images to train the model I will avoid throwing away 8% of the data. To handle these, I will replace the 'na' with the average incidence angle. I expect the image data to be more important for classification than the angle data and so the data is still valuable as a way to train the neural network's interpretation of the images.

Convolutional neural networks have proven effective in image classification problems [7]. However, they can require long training times, which increase with width and depth of the neural network. The training time can be reduced by using transfer learning.

I will create a convolutional neural network, using VGG16 to help train it. For the fully connected layer I will start with a single hidden layer of 32 neurons as this proved effective when classifying dog breeds in a previous Udacity project. If that proves to be too simple then I will increase the depth and width of the hidden layers. Initially, I expect the accuracy to be very low and so I anticipate having to create a complex neural network. Later I intend to replace VGG16 with other transfered models, but I will not be able to measure the performance change if the accuracy of the model is already as low as guessing. I will continue to increase the complexity of the hidden layers until the model reaches an accuracy that would allow me to distinguish between the performance of different transfer learning training models.

Eventually I will experiment using different transfer-learning models. I will assess the performance of the model when trained using VGG16, ResNet50 and Inception-ResNet V2. Once I have compared the performance of these, I will choose one model and perform my final tuning on it.

In addition to the images, I will need to pass in the incidence angle to the hidden layers of the model. To do this I will need to create an image generator that includes the incidence angle as well as the image. I will also need to use keras' concatenation to add an additional input. The code snippet below shows how this can be done [6].

```
#Create the input for the angles
angles_input = Sequential()
angles_input.add(Dense(1,input_shape=(1,),activation='sigmoid'))

#create the vgg16 model to take in images as input
vgg16_model = .....

combined_model = keras.layers.Concatenate([vgg16_model,angles_input])
```

This approach will allow the neural network to use data from the images as well as from the incidence angle, without requiring any manual intervention.

To estimate the log loss of the model, I will use 10-fold cross validation.

References

[1] C. A. Bentes da Silva, A. Frost, D. Velotto, and B. Tings. Ship-iceberg discrimination with convolutional neural networks in high resolution sar images. In *11th European Conference on Synthetic Aperture Radar Electronic Proceedings, 06-09 June, 2016, Hamburg, Germany*, pages 491–494. VDE Verlag GmbH, 2016.

[2] C-Core. `"https://www.c-core.ca/IcebergSurveillance"`. Accessed: 28/12/2017.

[3] A. et al. Radarsat-2 mode selection for maritime surveillance. pages 38–70, 2011.

[4] Kaggle.com. `"https://www.kaggle.com/c/statoil-iceberg-classifier-challenge"`. Accessed: 23/12/2017.

[5] Kaggle.com. `"https://www.kaggle.com/wiki/LogLoss"`. Accessed: 23/12/2017.

[6] Keras. `"https://keras.io/layers/merge/"`. Accessed: 23/12/2017.

[7] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[8] UNCTAD. Review of marittime transport 2016. page x, 2016.

[9] P. Yu, A. K. Qin, and D. A. Clausi. Feature extraction of dual-pol sar imagery for sea ice image segmentation. *Canadian Journal of Remote Sensing*, 38(3):352–366, 2012.