J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr
Copyright ⓒ 2018 KSII

Iceberg-Ship Classification in SAR Images Using Convolutional Neural Network with Transfer Learning

최 정 환[1*]
Jeongwhan Choi

ABSTRACT

Monitoring through Synthesis Aperture Radar (SAR) is responsible for marine safety from floating icebergs. However, there are limits to distinguishing between icebergs and ships in SAR images. Convolutional Neural Network (CNN) is used to distinguish the iceberg from the ship. The goal of this paper is to increase the accuracy of identifying icebergs from SAR images. The metrics for performance evaluation uses the log loss. The two-layer CNN model proposed in research of C.Bentes *et al.*(1) is used as a benchmark model and compared with the four-layer CNN model using data augmentation. Finally, the performance of the final CNN model using the VGG-16 pre-trained model is compared with the previous model. This paper shows how to improve the benchmark model and propose the final CNN model.

☞ keyword : Convolutional Neural Network, Deep Learning, Transfer Learning, VGG-16, Pooling Layer, Adam Optimizer, Data Augmentation

1. Introduction

Drifting icebergs present threats to navigation and activities in areas such as offshore of the Arctic Ocean. Icebergs similar in shape to ships can be hard to distinguish, which can give a great impact to the passing ship.

Many organizations and companies are currently monitoring, but remote monitoring is not possible in bad weather. Therefore, there is no way to monitor from satellites. SAR from satellite is important for marine monitoring. The difference in response to backscattering is stronger than in seawater, which can distinguish between ship and iceberg. But it is not easy to distinguish between ship and iceberg.

In this paper, Convolutional Neural Network (CNN) is used to identify these. CNN, which imitates human visual cognitive processes, is, of course, used in the field of computer vision [2]. Convolution technology, which shows excellent performance in extracting desired features from various types of data, is used in various fields such as image recognition and speech recognition. The reason for using this technique in image and speech recognition is to separate and extract features contained in signals such as original image or sound wave.

The CNN model learns flattened image data as input and then determines whether the image is an iceberg or a ship. The result is shown in the range of 0 and 1, and if it is close to 1, it is judged to be iceberg. And since there are not many 1604 data sets, transfer learning, data augmentation, and k-folds are used to solve the problem with insufficient training data. Finally, the improved CNN model is compared with other CNN models.

Those techniques are used in the process of improving from the benchmark model to the final model. In Section 4, the process and the experimental results are shown. Section 3 also introduces those techniques for the final model and mentions preprocessing techniques for the dataset. The dataset provided by Statoil / C-Core Iceberg Classifier of Kaggle competition is used in this paper [3].

1 Dept. of Software Engineering, Chonbuk National University, Jeonju, 54896, Republic of Korea
* Corresponding author(jeongwhan.choi@jbnu.ac.kr)

[Received 15 May 2018, Reviewed 27 May 2018(R2 18 July 2018), Accepted 24 July 2018]

2. Related Works

In this research of C.Bentes *et al.*[1], it is best to use the CNN model as a way to distinguish icebergs in SAR images. This research shows that the CNN model has the highest accuracy by comparing SVM, SVM + PCA, and CNN models. However, there was no accurate information about the method used by the CNN model. They used two convolutionlayers and softmax but did not reveal which pooling technique was used.

I set the model used in the above research as a benchmark model, and compare it with the improved models with better performance.

Some of the SAR images may include tough conditions such as target translation, random speckle noise, and missing pose. According to J. Ding *et al.*, these problems have shown that data augmentation can improve target recognition accuracy [4]. While their approach solves the worst-case conditions of the SAR image, my paper also uses this technique to supplement training data that is lacking. Data augmentation is actually used to deal with fewer training data in the paper of A.Krizhevsky *et al.*[5]. This method is also mentioned with dropout as a way to prevent overfitting [6], [7].

However, in addition to data augmentation, transfer learning can be used to solve lower performance due to insufficient training data [8] - [10].

Therefore, this paper shows how data augmentation, and transfer learning can improve the CNN model. In other words, I compared the two-layers CNN model used in the research of C. Bentes *et al.*[1] with the improved models with higher accuracy.

3. Methodology

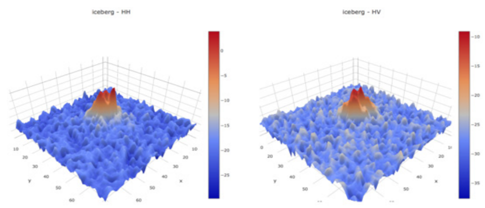
3.1 Data Exploration

This dataset is in JSON format. This file contains the following fields: ‘band_1’, ‘band_2’, ‘id’, ‘inc_angle’, ‘is_iceberg’.

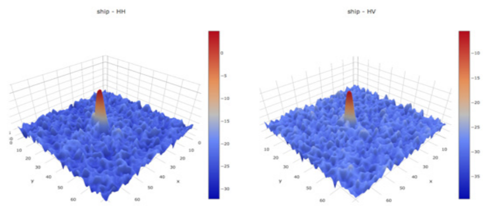
The ’band_1’ and the ’band_2’ are signals characterized by radar backscatter produced from different polarizations at a particular incidence angle [11]. The ’band_1’ is a 75x75 pixels flattened grayscale image taken by satellite that transmit and receive horizontally. Conversely, the ’band_2’ is the image received by the satellite both horizontally and vertically. The ’angle’ is the angle from which the image was received, and the ’is_iceberg’ is the result that determines whether the image is the iceberg. This target variable is set to 1 if it is an iceberg and 0 if it is a ship.

This data has a total of 1604 data and has 53.05% of ships and 46.94% of icebergs. These are close to balanced.

The figures below show the HH (transmit/receive horizontally) and HV (transmit horizontally and receive vertically) of the sample image data in 3D. Since these are not an actual images, it may contain scattered from the radar, which will cause peaks and distortions in the shape.



(Figure 1) Sample 3D images with a target variable of 1 (HH data on the left, HV data on the right).



(Figure 2) Sample 3D images with a target variable of 0 (HH data on the left, HV data on the right).

The shape of the iceberg in the radar data may look like a mountain as shown here. Conversely, the shape of the ship will be like a point, it can be like elongated point. Thus there is a structural difference, and CNN can be used to take advantage of these differences.

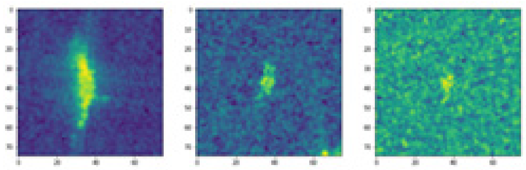
3.2 Data Preprocessing

3.2.1 Image Processing

The VGG-16 architecture requires images to be color images [12]. That is, the image has three channels. For example, an image 224x224x3 in size would mean an image with a height of 224 pixels, a weight of 224 pixels, and 3 RGB color information. Hoewever, 'band_1' and 'band_2' are both grayscale images and are 75x75x1 in size. In other words, it does not have three channels. So I need to find a way to format 75x75x3 images.

To satisfy this, 'band_1' was used as the first color, 'band_2' as the second color, and the third color was created as the average of two colors. Therefore, images with a 75x75x3 format can be used.

The following is sample images after image processing.


(Figure 3) Sample images after image processing.

3.2.2 Angle Processing

Since 133 angle values in the dataset do not appear as 'na', a method should be used to meet these missing values. The 'na' values were filled using the fillna of pandas [13], a way to fill in the missing values in the forward direction.

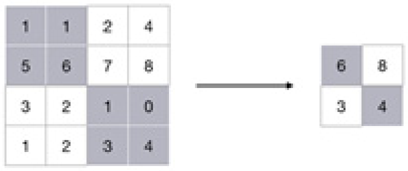
3.2.3 Data Augmentation

There is a way to increase the number of training data for better learning and prediction. To do this, the number of images must be increased by flipping or rotating them. The ImageDataGenerator from Keras is used to transform the images and augment data. In this paper, the images were flipped vertically and horizontally with a 10 degree range and a zoom range of 0.8 to 1.2.

3.3 Algorithms and Techniques

3.3.1 Max Pooling Layer

Pooling is an operation that reduces space in the vertical and horizontal directions. For example, as shown in the figure below, the 2x2 area is aggregated into one element to reduce the space size.



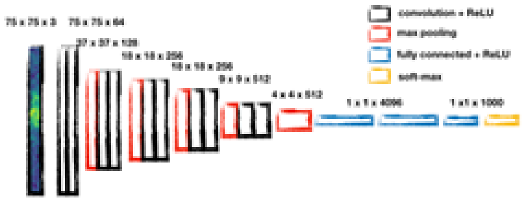
(Figure 4) Max pooling with 2x2 filters and stride 2.

The max pooling is the operation that takes the maximum value in the target area, while the average pooling computes the average of the target area. Since the max pooling is mainly used in the filed ofimage recognition, the max pooling is similarly used in this paper.

3.3.2 VGG-16

VGG is a 'basic' CNN consisting of a convolition layer and a pooling layer [14]. As shown in

the figure, both the convolution layer and the fully collected layer are deepened to 16 layers.



(Figure 5) VGG-16 architecture.

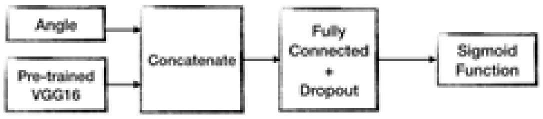
The point to note in VGG-16 is that the convolution layer using 3x3 small filter runs continuously. The pooling layer is repeated or four times in succession to reduce the size to half. At the end, it passes through the fully connected layer and outputs the results.

In the VGG-16, the ReLU activation function is used for these convolution and fully-connected layers, and this function is defined as following:

$$f(x) = \max(0, x)$$

This is a transform that removes the negative part of the input and solves the vanishing gradient problem that appears in sigmoid or tanh [15]. Also, because of the computational efficiency, faster training is possible.

3.3.3 Transfer Learning



(Figure 6) The architecture used in the final model. The fully connected and dropout layers are simply represented as one box.

Transfer learning is useful when threre are few training dataset. It uses the weight values learned from the huge dataset provided by ImageNet for the dataset of this paper. Weights copied from the ImageNet are copied to another neural network, and fine tuning is performed in that state.

The neural network with the same configuration as the VGG is prepared, the previously learned weight is set as the initial value, and fine tuning is performed on the new dataset [9]. Therefore, this is a useful method when it is judged that training data is insufficient.

3.3.4 Early Stopping

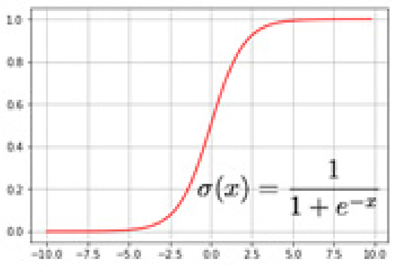
Early stopping is a rule to prevent overfitting [16]. This rule provides guidance on how many iterations can be performed before the learner is overfitting. A validation error is used to determine when overfitting has begun. This method is most commonly used for neural network training. Early stopping is also determined by a parameter called patience. The model in this paper sets the default patience to 10, which stops learning if the validation loss is no longer improved during 10 epochs.

3.3.5 Adam

When calculating the loss function, using the entire train set requires too much computation, so Stochastic Gradient Descent (SGD) is used to avoid this. This method computes the loss function only for a batch of small data(mini-batch) instead of whole data(batch). However, according to S. Ruder’s study, the performance of SGD is significantly lower than that of other optimizers [17]. Thus, instead of SGD, Adaptive Moment Estimation (Adam) was used in this paper as an optimizer for CNN model. Adam is an algorithm that combines RMSProp and Momentum [18].In this method, similar to the Momentum method, the exponential average of the gradient calculated up to now is stored, and the exponential average of the squared value of the gradient is stored similarly to RMSProp. In other words, this is to go down in inertia direction to grasp previous context and reduce step size.

3.3.5 Sigmoid Function

This model uses the sigmoid layer as the last layer, and it is appropriate to use the sigmoid layer because it is a binary classification that identifies wheter the input image is a ship or an iceberg.



(Figure 7) The sigmoid function and graph.

The sigmoid function above has the output range 0 to 1. As with the target variable, the result of the prediction must be between 0 and 1. Thus, to make this prediction, use the sigmoid function so that the result is not less than 0 or greater than 1.

4. Experiment

4.1 Experimental Environment

All experiments were carried out on an Amazon EC2 p2.xlarge GPU compute instance. A detail of the instance is below Table 1. Using this reduces the model training time. The based operating system of the instance is Ubuntu 16.04. The model is implemented in Keras 2.2.0 running on top of TensorFlow 1.8.

(Table 1) The detail of p2.xlarge instance type.

Type	GPUs	vCPU	RAM(GB)	GPU Memory(GB)
p2.xlarge	1	4	61	12

4.2 Metrics

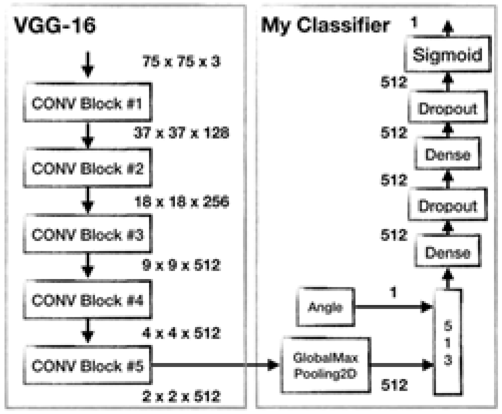
All Log-Loss measures the performance of a classification model where the predicted input is a probability value between 0 and 1. The goal of the CNN model is to minimize this value. The perfect model will have zero log loss. This value increases as the predicted probability deviates from the actual label.

$$logloss = \frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1-y_i)\log(1-p_i))$$

On the other hand, accuracy is the number of predictions whose predicted value is equal to the actual value. Accuracy is not always a good indicator because of the yes or no nature. Because Log-Loss considers the uncertainty of the prediction depending on how far it is from the actual label, this paper uses Log-Loss when evaluating the model.

4.3 Transfer Learning

In the final model, transfer learning is performed using the VGG-16 pre-trained model. As shown in the Figure 8 below, 75x75x3 images are input and pass through a total of 5 convolutional blocks.



(Figure 8) Architecture of the final CNN model.

The box on the left represents the VGG-16 model, and the box on the right represents the classifier used in the final model. In VGG-16, the output of 2x2x512 is reduced in dimension by the global max pooling layer and concatenated with the angle data.It then reaches the final layer via two fully-connected and dropout layers. As shown in the figure above, the sigmoid layer is placed on the last layer to identify whether it is iceberg or ship.

(Table 2) The layers of each convolution block.

Block1	Block2	Block3	Block4	Block5
CONV Layer	CONV Layer	CONV Layer	CONV Layer	CONV Layer
CONV Layer	CONV Layer	CONV Layer	CONV Layer	CONV Layer
Max Pooling	Max Pooling	CONV Layer	CONV Layer	CONV Layer
		Max Pooling	Max Pooling	Max Pooling

The above table provides more information on the convolutional blocks of the VGG-16 model. There are two convolutional layers in the first two blocks and three convolutional layers in the last three layers. Each block contains a MaxPooling layer.

4.4 Experimental Results

4.4.1 Benchmark two-layers CNN Model

The two-layers CNN model used inthe research of C.Bentes *et al.* is used as a benchmark model [1]. In their paper, it was not specified which pooling technique the model used, and softmax was used as the last layer. However, the model in this paper is not learned using softmax as the last layer. The softmax function is not suitable for use for multi-classification. Therefore, a sigmoid funcion for binary classification should be used. It is appropriate to use the sigmoid function because the goal in this paper is to refine a binary classification model that distinguishes whether it is a ship or an iceberg.

(Table 3) Benchmark model summary.

Layer	Layer Type	Output Size
0	Input	75 x 75 x 2
1	CONV Layer	73 x 73 x 32
2	MaxPooling	36 x 36 x 32
3	CONV Layer	34 x 34 x 64
4	GlobalMaxPooling	64
5	Dropout	64
6	Fully Connected	1
7	Sigmoid	1

The Table 4 below shows the results based on the above model. Comparing the model with the max pooling layer to the model with the average pooling layer, it is better to use the max pooling layer because the latter validation loss is 0.18 lower. That is, the max pooling technique is mainly used as a powerful classifier tool in image recognition.

(Table 4) Model performance comparison for average and max pooling.

Pooling	Validation Accuracy	Validation Loss
Average	60.21 %	0.5916807956046
Max	76.96 %	0.4186338916495

4.4.2 Deeper CNN Model with Data Augmentation

The table below shows a model with four convolution layers and three fully-connected layers. This is a model with a deeper layer than the benchmark model. The max pooling layers are located after each convolution layer. Of the last three output layers, the first two layers use the ReLu activation function and the last layer uses the sigmoid activation function.

(Table 5) Deeper CNN model summary.

Layer	Layer Type	Output Size
0	Input	75 x 75 x 3
1	CONV Layer	73 x 73 x 64
2	MaxPooling	36 x 36 x 64
3	Dropout	36 x 36 x 64
4	CONV Layer	34 x 34 x 128
5	MaxPooling	17 x 17 x 128
6	Dropout	17 x 17 x 128
7	CONV Layer	15 x 15 x 128
8	MaxPooling	7 x 7 x 128
9	Dropout	7 x 7 x 128
10	CONV Layer	5 x 5 x 64
11	MaxPooling	2 x 2 x 64
12	Dropout	2 x 2 x 64
13	Flatten	256
14	Fully Connected	512
15	ReLU	512
16	Dropout	512
17	Fully Connected	256
18	ReLU	256
19	Dropout	256
20	Fully Connected	1
21	Sigmoid	1

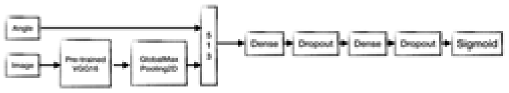
In addition, data augmentation techniques are used to solve the problem of insufficient data.The following table compares the results with and without data augmentation.

(Table 6) Model performance comparison for using data augmentation techniques and not using.

Data Augmentation	Validation Accuracy	Validation Loss
No	87.03 %	0.274543564664
Yes	90.76 %	0.207176460736

Using the data augmentation technique, the validation loss is reduced by 0.07 and the validation accuracy is increased by 3.73. and can be refined to a better performance model. This means that the data augmentation technique can improve the CNN model with better performance.

4.4.3 Final CNN Model with Transfer Learning



(Figure 9) The same final model as Figure 6. It is compressed into 512 one-dimensional information by Global MaxPooling2D, and is connected to angle information, and 513 are input to a fully-connected layer.

The final CNN model is shown in the Figure 6 above. It improves accuracy by about 92% over previous models. As before, the Adam optimizer is used as the gradient descent optimizer algorithm of the final model. In addition, resampling technique is used through 10-folds cross validation. This not only improves the statistical reliability of the classifier performance measurements through resampling, but is also useful when the amount of data is not sufficient.

(Table 7) Comparison of model performance according to each k-fold cross validation.

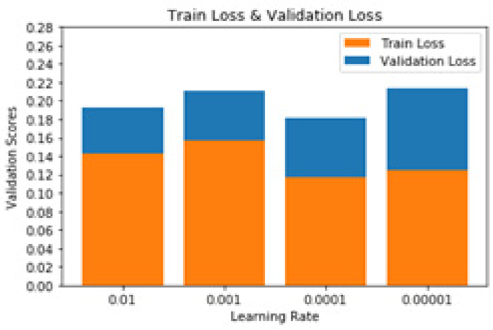
K	Validation Accuracy	Validation Loss
3	91.64	0.208624429906
5	91.82	0.196969180077
7	91.31	0.204074223575
10	92.39	0.181701044989

The above Table 7 shows the cross-validation results for each k value. The learning rate in this experiment is 0.0001. In order to select the most suitable k for this model, the k value is changed as above and the comparison is made. When k is 3, 5, and 7, the loss is close to 0.2 and the accuracy remains at 91%, but when k is 10, the loss is reduced to about 0.18 and the accuracy is also improved to 92%.

(Table 8) Comparison of model performance according to learning rate.

Learning Rate	Validation Accuracy	Validation Loss
0.00001	91.64	0.213980056827
0.0001	92.39	0.181701044989
0.001	91.39	0.211348751952
0.01	92.14	0.20165806206

The model, which is improved to 92.39%, uses 0.0001 as the learning rate. The Table 8 shows the loss and accuracy results for each learning rate. The validity of this value can be confirmed by the Figure 10. If the learning rate is 0.01 and 0.001, the test log loss is larger than the remaining 0.0001. Both 0.0001 and 0.00001 have low train log loss, but suitability is determined by the difference from the test log loss. Since the test log loss of this learning rate is not significantly different from the train log loss, the best learning rate is 0.0001.



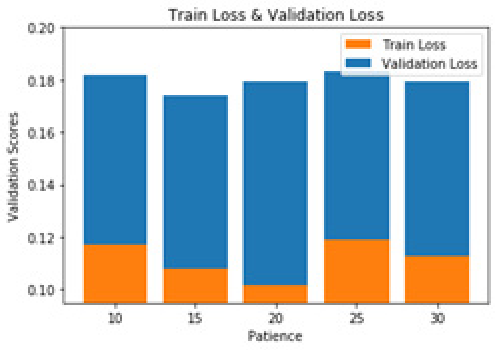
(Figure 10) Difference between train loss and validation loss of each learning rate.

So far, CNN models have been subject to the patience of 10 epochs with early stopping applied. However, before determining the final model, it should be ensured that the number of patience properly prevents overfitting. If patience is too large, it will not prevent overfitting, but if it is too small, it will terminate too early. Therefore, patience experiments from 10 to 30 showed the most appropriate early stopping rule, and it is the most appropriate learning end when patience is 15.

(Table 9) Comparison of model performance according to patience.

Patience	Validation Accuracy	Validation Loss
P = 10	92.39	0.181701044989
P = 15	92.89	0.174257478678
P = 20	92.70	0.179240771208
P = 25	92.52	0.183048155108
P = 30	92.39	0.179382493428

In the above Table 9, the lowest validation loss value is when patience is 15. This validation loss value is also the most appropriate because the difference from the train loss value is smallest compared to other patience values. The patience is 20, but the difference between the train loss and the validation loss is bigger than the others, so it can not be selected as the early stopping rule of the final model.



(Figure 11) Difference between train loss and validation loss of each patience.

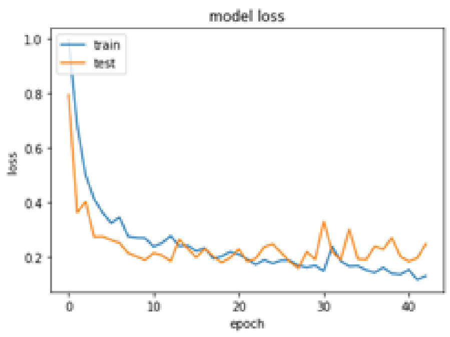
4.5 Final Model and Discussion

Again referring to the experimental results, the CNN model using data augmentation has lower validation loss by about 0.2 than the benchmark model. Finally, compared to the deeper CNN model using the data augmentation technique, the final CNN model has lower validation loss by about 0.033, improving the model performance. The final CNN model used data augmentation technique and transfer learning to solve the problem of insufficient training data, and improved the statistical reliability of the classifier performance measurement using k-fold cross validation. Also, overfitting was prevented by early stopping rule and dropout. Using these techniques, the final model yielded the following results.

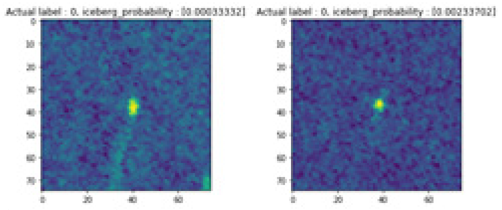
(Table 10) Performance result table of final model.

Final Model	
Training Loss	0.107999644928
Validation Loss	0.174257478678

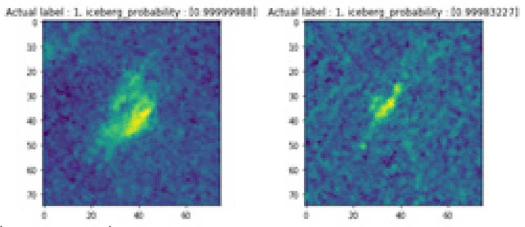
The Figure 12 below shows the train loss and validation loss of the final model. The loss value indicates how the performance of this model is per epoch. The train loss of this final model shows good loss function with optimal learning rate and early stopping rule is applied before validation loss and loss value are deteriorated.



(Figure 12) The validation loss graph of the final model.


Actual label : 0. iceberg_probability : [0.00033332]
Actual label : 0. iceberg_probability : [0.00233702]

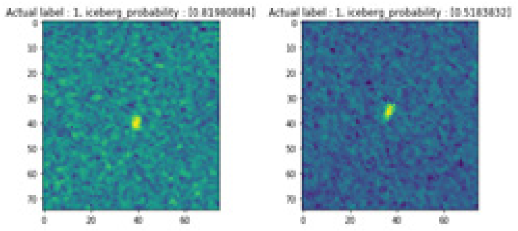
(Figure 13) Correct prediction results among the ship samples in the test set. The probability of iceberg on the left is 0.00033332, the right is 0.00233702.


Actual label : 1. iceberg_probability : [0.99999988]
Actual label : 1. iceberg_probability : [0.99983227]

(Figure 14) Correct prediction results among the iceberg samples in the test set. The probability of iceberg on the left is 0.99999988, the right is 0.99983227.

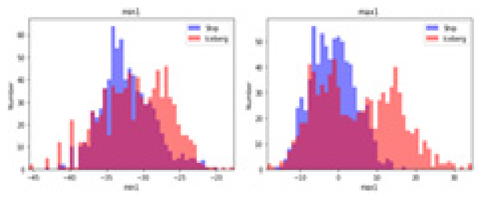
The above Figure 13 and 14 are the predicted result samples of the final model. The corresponding value of 'iceberg_probability' is 1 for iceberg and 0 for ship. Those figures are good examples of iceberg and ship classification, but the Figure 15 below is hard to discern as they are small-sized iceberg images. As shown below, the left image has a good classification of about 0.8, but there is still a tricky image to classify as the right image.

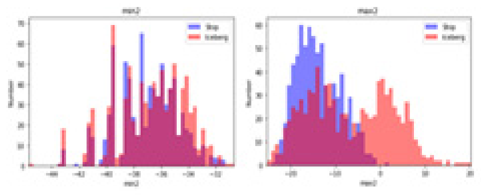
This means that it is difficult to recognize iceberg images that are smaller than ship size. In practice, the maximum and minimum values of the training data set are as shown in the Figure 16 and 17 below. In the case of the graph showing the minimum value of the data, there are some values smaller than ships. This dataset can be said to be mainly occupied by more pixels than icebergs, but it can be difficult to predict because of exceptions as shown in the Figure 15 below.


Actual label : 1. iceberg_probability : [0.81980884]
Actual label : 1. iceberg_probability : [0.5183832]

(Figure 15) The prediction result of a small iceberg image in the test set. The probability of iceberg on the left is 0.81980884, the right is 0.5183832.

The problem is that performance is no longer improved with a validation loss of 0.17425, which should be further refined in the final model. Although the performance of the benchmark model is superior to that of the benchmark model, the validation loss of the final model is near 1.7. This value decreases to less than 1.0, and model improvement is needed to show better performance.


(Figure 16) The distribution of the minimum and maximum values for the 'band_1' (HH) fields of the dataset.


(Figure 17) The distribution of the minimum and maximum values for the 'band_2' (HV) fields of the dataset.

6. Conclusion

The goal in this paper is to enhance the performance of the CNN model to distinguish between ship and iceberg. The performance of the CNN model is evaluated as validation loss and a total of 1604 data with 851 times and 753 iceberg values are used. After training with the training data divided by 10 fold cross validation, the validity of the final model is checked with the test set corresponding to the fold. First, the two-layer CNN model used in the research of Bentes *et al.*[1], the benchmark model, was used to evaluate the model performance. Second, the deeper layer CNN model using data augmentation technology is used to improve the validation loss to 0.2. Finally, the final model was transferred learning to the VGG-16 pretrained model to improve the validation loss to 0.1725.

Therefore, this paper proposes a model using transfer learning, data augmentation, and K-fold cross validation as CNN models that identify ship and iceberg in SAR images.

Suggested future work includes a study that other pre-trained models also can be expected the performance improvement. In this paper, the VGG16 architecture was used as a pre-trained model as part of the improvement method, and it can be expected in other pre-trained models like AlexNet, GoogleNet, and ResNet. In addition, it is necessary to study using data pre-processing how to solve the problem using the data pre-processing that it is difficult to distinguish the iceberg image which is similar to or smaller than the size of the ship, which is the limit at the end of Section 5.

Reference

[1] C. Bentes, A. Frost, D. Velotto, and B. Tings, "Ship-iceberg discrimination with convolutional neural networks in high resolution SAR images," in *EUSAR 2016: 11th European Conference on Synthetic Aperture Radar, Proceedings of,* pp. 1–4, 2016.
https://doi.org/10.1109/joe.2017.2767106

[2] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, 2016.
https://doi.org/10.1007/bf00344251

[3] Kaggle.com, "Statoil/C-CORE Iceberg Classifier Challenge | Kaggle." [Online]. Available: https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/data. [Accessed: 24-Feb-2018].

[4] J. Ding, B. Chen, H. Liu, and M. Huang, "Convolutional Neural Network with Data Augmentation for SAR Target Recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 3, pp. 364–368, 2016.
https://doi.org/10.1109/lgrs.2015.2513754

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

[6] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "High-Perfor-

mance Neural Networks for Visual Object Classification,” 2011.
https://doi.org/10.1145/3065386

[7] D. Cireşan, U. Meier, and J. Schmidhuber, “Multi-column Deep Neural Networks for Image Classification,” *Int. Conf. Pattern Recognit.*, no. February, pp. 3642–3649, 2012.
https://doi.org/10.1109/cvpr.2012.6248110

[8] R. Wagner, M. Thom, R. Schweiger, G. Palm, and A. Rothermel, “Learning convolutional neural networks from few samples,” *Proc. Int. Jt. Conf. Neural Networks*, pp. 1884–1890, 2013.
https://doi.org/10.1109/ijcnn.2013.6706969

[9] W. Zhao, “Research on the deep learning of the small sample data based on transfer learning,” vol. 020018, p. 020018, 2017.
https://doi.org/10.1063/1.4992835

[10] N. . C. B. . P.-S. A. . C. A. . Chakraborty D.a b Kovvali, “Structural damage detection with insufficient data using transfer learning techniques,” *Proc. SPIE - Int. Soc. Opt. Eng.*, vol. 7981, no. May, pp. 1–9, 2011.
https://doi.org/10.1117/12.882025

[11] E. Attema *et al.*, “Sentinel-1 ESA’s new European SAR mission,” *Proc. SPIE*, vol. 6744, pp. 674403-674403–8, 2007.
https://doi.org/10.1117/12.747146

[12] Y. Z. Jeff Hwang, “Image Colorization with Deep Convolutional Neural Networks,” *Cs231N. Stanford.Edu*, 2016.
https://doi.org/10.1109/icmla.2016.0019

[13] W. McKinney, “Data Structures for Statistical Computing in Python,” *Proc. 9th Python Sci. Conf.*, vol. 1697900, no. Scipy, pp. 51–56, 2010.

[14] K. Simonyan and A. Zisserman, “Very Deep Networks for Large-Scale Image Recognition,”, pp. 1–14, 2014.
https://doi.org/10.1109/cvpr.2014.219

[15] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” *Proc. 27th Int. Conf. Mach. Learn.*, no. 3, pp. 807–814, 2010.
https://doi.org/10.1109/icassp.2010.5495651

[16] L. Prechelt, “Early stopping - But when?,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7700 LECTU, pp. 53–67, 2012.

[17] S. Ruder, “An overview of gradient descent optimization algorithms,”, pp. 1–14, 2016

[18] D. P. Kingma and J. L. Ba, “Adam: A Method for Stochastic Optimization,” *arXiv preprint arXiv:1412.6980*, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980.

◐ 저 자 소 개 ◑

최 정 환(Jeongwhan Choi)
2016년~현재 전북대학교 소프트웨어공학과 (학부과정)
관심분야 : 머신러닝, 컴퓨터 비전, 딥러닝
E-mail : jeongwhan.choi@jbnu.ac.kr