REPORT

Class Name : Artificial Intelligence Project

Subject   : Final report

Professor   : 이 지 형

김 동 원(전자전기공학부, 2018312292)

임 승 재(소프트웨어학과, 2018310773)

Submission Date : 2021.12.11

Contents

Table Contents

Figure Contents

그림 목차 항목을 찾을 수 없습니다.

Summary of task, goal, analysis of input data, and metrics

In order to collect marine debris, it is intended to conduct a design survey more quickly and efficiently. To this end, we intend to design an artificial intelligence model that can classify multi-labeled underwater photographed images.

Classifying models are going to be trained and validated by images taken underwatered. Trained models including VGGNet, ResNet, DenseNet, and DarkNet will be compared. The loss of each model is going to be analyzed by using LIME which is a kind of XAI(eXplainable Artificial Intelligence). By focusing on the result of LIME, the strengths of weaknesses of models can be compared. The analyzed result can be used for further research for constructing proper model for marine debris classification.

We will use '해양 침적 쓰레기 이미지' dataset from AIHub. The dataset provides training data and validation data, and we intend to train the model using training data and test the model using validation data. In particular, only underwater photographed images, excluding sonar image data, will be used. The reason is that in the case of sonar images, it is not easy to classify them into models we implement because they are images taken using special equipment related to sound waves and are different from general images. In addition, since the data provided by the dataset is image data of different sizes, preprocessing such as adjusting the size is required. Images in 'root/data' and 'root/test_data' of the submitted task are the results of preprocessing. The code used for preprocessing can be found in 'root/src/utils/AI_preprocessing.py'.

We will use the accuracy of the model as the metrics for evaluating the model

Data Source

We are using "해양 침적 쓰레기 이미지" [1] data from https://aihub.or.kr/. The exact address of the data is https://aihub.or.kr/aidata/30754.

Explanation for our model

All models in the project are implemented by ourselves referring the original papers of each model.

1. ResNet

   ResNet is the model presented in "Deep Residual Learning for Image Recognition" [2]. It is characterized in that the feature of the image is extracted using only the convolution layer of size 3x3. In addition, the paper introduced the concept of residual block and designed short cuts of input values. However, in our project, the form of ResNet-18 was used among several forms introduced in the paper to use the most basic form of ResNet.

2. VGGNet

   VGGNet is called as starting point of deep neural network. Before the model, models who won ImageNet Large Scale Visual Recognition Challenge used at most 8 layers. However, after VGGNet which used 19 layers, lots of models had deep networks. Since considering that other models are inspired by VGGNet, using VGGNet for comparison becomes meaningful. VGGNet use 3x3 kernel most often. Among VGGNet, VGG-16(D type in the paper) is used for the project since it is original mentioned model in the official paper. [3]

3. DenseNet

   DenseNet has similar structure with ResNet but tries to get better

performance than ResNet. There are some features in the model. In DenseNet, it uses two kinds of block which are bottleneck block and transition block. Bottleneck block instead of normal dense block to reduce feature map size. Bottleneck block contains 1 x 1 convolution layer before 3 x 3 convolution layer. Therefore, the block can reduce the feature map size and increase computational efficiency. Transition block is used for compressing data. It is done by adding 2 x 2 average pooling layer at the end of the block. In DenseNet, the number of dense block changes as 6, 12, 24, and 16 (In the project, DenseNet-121 is used). The model uses growth_rate and set proper input channel size for each block. There are some variations for the DenseNet such as 12, 24, 40 etc. according to the paper [4].

4. DarkNet

DarkNet is a backbone network of well-known image detection model, YOLO series. Especially from YOLOv2, DarkNet has used. Darknet is a kind of variation of GoogleNet but uses 1 x 1 convolution layer to reduce feature map size as DenseNet did. Among Darknet, Darknet-53 is used for this project since it is a backbone network of YOLOv3. The goal of this project is classifying marine debris images not detecting them, only backbone network is used. In the model, there are two kinds of block which are convolution block and residual block. Convolution block is consisted of 3 x 3 convolution layer, batchnormalization, LeakyReLU. Residual block is used in the model to get more meaningful semantic information form upsampled feature and finer-grained information from earlier feature map. In the residual block, x 2 upsampling is done and the result feature map is concatenated with 2 step earlier feature map. [5]

Explanation of experiments

1. Training process

   BCEWithLogits was used as a criterion in the process of training the model. It was judged to be the most suitable criterion for implementing the multi-label classification model. In addition, SGD was used as an optimizer and CosineAnnealingLR was used as a scheduler. The model is basically supposed to learn a total of 100 epochs, but early stopping is applied to terminate the learning if there is no improvement in the valuation loss during 10 epochs. In this case, the model recording the lowest validation loss is stored in the 'root/model' path.

2. Hyperparameters

   ResNet: ResNet uses three hyperparameters (block, layers, num_classes).

   a) Block determines which block ResNet will use, BasicBlock or Bottleneck.

   b) The layers determine how the layers of ResNet will be configured. In our project, we used [2, 2, 2, 2] to use the most basic form of ResNet.

   c) num_classes must convey how many classes the dataset consists of.

   VGGNet: VGGNet uses only one hyperparameter(num_classes).

   a) You must deliver how many classes the dataset consists of through num_classes.

   DenseNet: DenseNet has four hyperparameters and default values are shown as followed: (droprate=0.2, block=BottleneckBlock, growth_rate=12, num_classes).

   a) 'droprate' is for dropout layer to prevent overfitting even though in the original paper of the model did not mention about dropout.

   b) 'block' is for declaring dense block used for the model. Generally, normal block can be used for DenseNet. However, in the project, to

reduce the number of parameters in the model and avoid overfitting, use BottleneckBlock as dense block. Bottleneck block has 1 x 1 convolution layer.

c) 'growth_rate' is used for declaring rate of channel growth in the model. Referring the DenseNet paper, 12 is used for the project.

d) 'num_classes' is the number of classes in the dataset.

DarkNet: Darknet has three hyperparameters (num_classes, block, droprate).

a) num_classes: the number of classes in the dataset

b) block: block to be used for the residual block in the model. Default is ResidualBlock defined in the same file.

c) droprate: droprate for Dropout layer. Although in the original paper of the model did not mention about dropout, in the project, for flexibility and prevent overfitting, dropout layer is used. Default value is 0.2

3. Validation

We removed some of the train data and used it for validation. The important thing in the validation stage is that since it is multi-label classification, the method of simply selecting the largest value from the model output cannot be used to find the accuracy of the model. Instead, a method of selecting values greater than a certain threshold after taking sigmoid to the model output was used. The value of threshold seems to be the most appropriate at 0.4 for ResNet and 0.3 for the remaining models.

Explanation for our test data

As mentioned above, the model will be tested using validation data provided by the "Marine Deposition Waste Image" dataset.

1. File formatter: jpg(3 channel RGB)

2. Number of classes: 13

3. Class per image: more than 1(represented as one-hot encoding)

4. Total number of images used: 9024

5. Preprocessing

   a) Unzip the data

   b) Resize the data to 64 x 64

   c) Tokenize file names

   d) Make class for classes (like Vocabulary in image captioning, store result as json file debris.json)

   e) Store result as json file(imgclass_map.json, imgname_map.json)

   f) Make Dataset

Analysis of our outcome

The results of calculating the accuracy of our four models using the test data are as follows.

Table 1 Test accuracies

ResNet	DarkNet	VGGNet	DenseNet
55%	21%	50%	43%

It can be seen that DenseNet, which has higher model complexity, is rather less accurate than VGGNet and ResNet. In particular, DarkNet, which has the highest model complexity, shows a sharp decline in accuracy to 21%. This is because our interpretation about the task was somewhat correct. We initially speculated that a high level of object detection model such as YOLO would not be required because this task is not so complex. Therefore, we approached this task as a multi-label classification problem using 4 models. We believe that the complexity of DenseNet and DarkNet are too high to perform this task, resulting in poor accuracy as overfitting occurs in training data.

However, it is regrettable that the highest accuracy among our models is 55%. I think the limitation of computing power in the development environment may have affected by a resize of 64*64, and the fact that the size of the learning dataset is not so large.

References

[1] 씨에스피아이, "해양 침적 쓰레기 이미지," 씨에스피아이, 18 6 2021. [온라인]. Available: https://aihub.or.kr/aidata/30754. [액세스: 20 11 2021].

[2] X. Z. S. R. J. S. Kaiming He, "Deep Residual Learning for Image Recognition," %1 *Microsoft Research*, 10 Dec 2015.

[3] A. Z. Karen Simonyan, "Very Deep Convolutional Networks For Large-scale Image Recognition," University of Oxford , 2015.

[4] Z. L. L. v. d. M. K. Q. W. Gao Huang, "Densely Connected Convolutional Networks," Cornell University, Tsinghua University, Facebook AI Research, 2018.

[5] A. F. Joseph Redmon, "YOLOv3: An Incremental Improvement," University of Washington.