Recognizing Birds by Image Recognition

Bochen Pan[a], Yujin Wang[b], Ruming Zhong[*c]
[a]Marshall College, The University of California, San Diego, La Jolla, California 92092, USA;
[b]National Pilot School of Software, Yunnan University, Kunming, 650504, China
[c]Computer Science, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada
*Corresponding author. Email: zhongroy189@gmail.com

ABSTRACT

In this work, three algorithms (HOG with SVM, VGG, ResNet) are chosen to perform better image recognition on different categories of birds. HOG with SVM performs worse than ResNetwith SVM because HOG is better at recognizing objects than identifying categories of objects. Hyper-parameters in HOG and SVM will affect the accuracy. (Bochen). To improve the accuracy of the image recognition, the VGG algorithm is adopted with different hyper-parameters. Yujin Wang employs “Deep residual learning for image recognition” from He K, Zhang X, Ren S, et al to show his understanding of the ResNet algorithm. He will test different hyperparameters in ResNet and show readers how they will affect the results.

Keywords: Image Recognition, Fine-grained, Accuracy, Hyperparameters

1. INTRODUCTION

Fine-grained Image recognition has become popular these years. People are trying to detect objects through cameras such as detection when a vehicle is coming to check whether the vehicle exceeds the speed limit. Most detections are based on recognizing different species. But what about detection on different kinds of one specific species such as different kinds of birds? People are not that familiar with specific categories of birds. So, in this study a dataset of bird-species images with annotations and attributes are chosen. To recognize each category of bird, this work will show fine-grained image recognition, which is a challenging task. By adopting different algorithms to recognize which category it is for birds by given images, accuracy of our algorithms will be presented as the performance of recognizing images.

2. DATA

Link: https://www.kaggle.com/veeralakrishna/200-bird-species-with-11788-images

Number of categories: 200

Number of images: 11,788

Annotations per image: 15 Part Locations, 312 Binary Attributes, 1 Bounding Box

Data Preprocessing: Since 10,000 data isn't enough to process recognition, data augmentation such as rotate the images or zoom out/in images would be a good option to increase data size. As a result, there will be 50000 images in the datasets.

3. FINE-GRAINED BIRD IMAGE RECOGNITION VIA SVM WITH HOG

3.1 Research Methods:

HOG, also called as histograms of oriented gradients, can be considered as a feature describer used in image processing for image detection, while SVM, also called as support-vector machines, are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. HOG and SVM are chosen because the article “Histograms of Oriented Gradients for Human Detection” by Navneet Dalal, Bill Triggs states about how people can use SVM with HOG to detect humans, which brings me inspiration [1]. HOG performs existing feature sets after “reviewing existing edge and gradient-based descriptors” [1]. The goal for part 3 is to test whether SVM with HOG is suitable for recognizing bird categories and how hyperparameters in HOG and SVM affect accuracy. With reference to codes in “Object-detection-via-HOG-SVM” [2] and “HOG-SVM-Python” from Jianlong Yuan [3], two githubs provide good examples for image recognition using HOG and SVM.

3.2 Evaluation

No normalizations are needed in this case because “these normalizations have only a modest effect on performance, perhaps because the subsequent descriptor normalization achieves similar results” [1]. Resizing all pictures into 125x125 size and getting 396 feature vectors by HOG. With more exploration, relationships are shown between SVM and HOG by different sizes of images. Greater image size will lead to more feature vectors. Although many feature vectors will perform more complete on SVM tests, they will lead to a more time-consuming waiting process when running codes. So, 125 x 125 image sizes is a good choice. With greater image size, test accuracy will become smaller. It is caused by greater size with more feature vectors, leading to more vectors to be tested and classified.

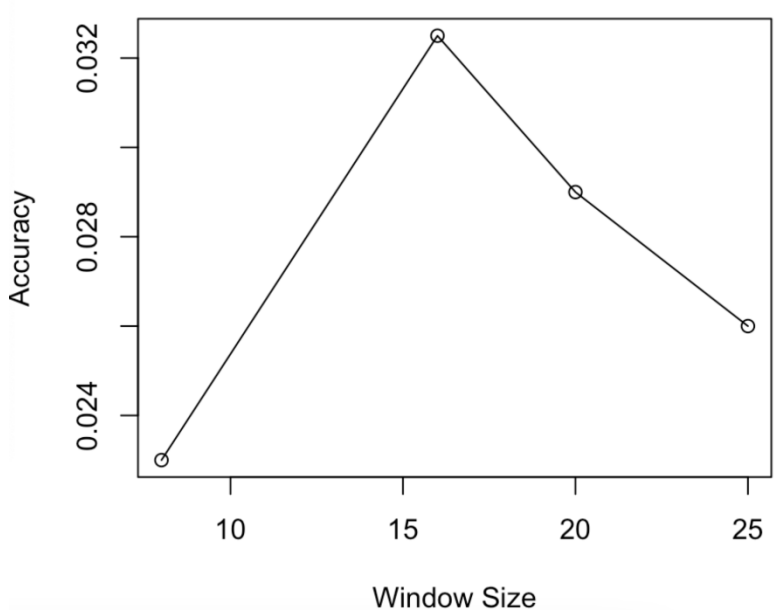
A. Creating dataframe with images’ vectors and categories to prepare fitting in SVM. Let each vector as columns and encode string labels into numerical values, which will be easier to fit in SVM.

B. Fit HOG feature vectors into SVM and test the effects of two hyperparameters of HOG on accuracy.

a. Window size: used for detecting how many pixels at once to recognize the image. (With 16 my default value)

b. Orientation: used to extract how many orientations for images. (4 as my default value)

c. Figure explanation: The first plot in figure 1 shows when the window size becomes 16 which is the most popular way of detection, the accuracy will be slightly greater than other window sizes. But when the window size becomes 8, the computer needs more time to perform, and accuracy will be slightly lower. The second plot in figure1 shows when orientation becomes 4, the accuracy will be higher than other orientations. Accuracy will be the least when performing orientation as 8. By fitting HOG feature vectors with categories into SVM, test accuracy for HOG+SVM is very low, about 3% accuracy for 200 categories, which is not quite convincing. Thus, SVM by ResNet101 may be a better choice to extract features from images.



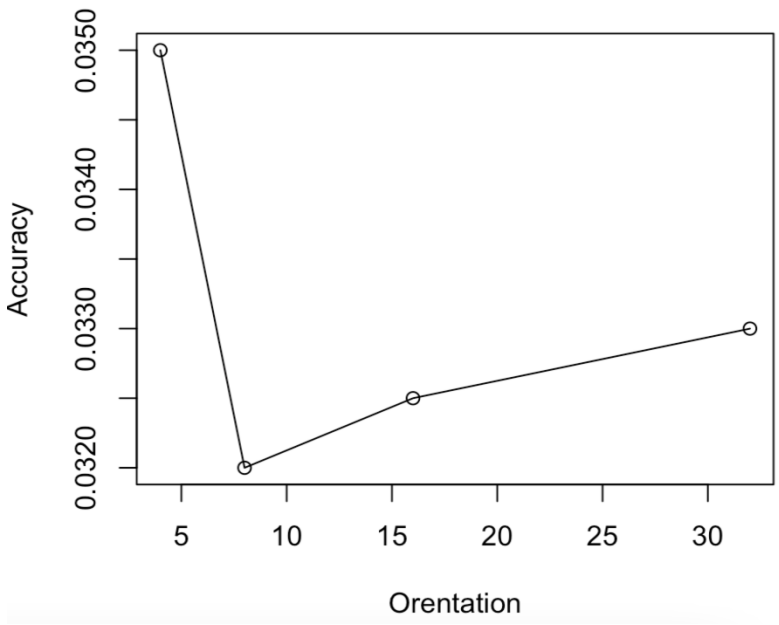


Fig. 1 plots of Relationships between HOG features vectors and SVM with different sizes of images

C. Fit ResNet features into SVM and test effects of four hyperparameters of SVM on accuracy

a. C: tells the SVM optimization how much you want to avoid misclassifying each training example. (Must be positive)

b. Gamma: another hyper parameter which we have to set before training a model. Gamma decides how much curvature we want in a decision boundary. Gamma high means more curvature.

c. Kernel: SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of the kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions such as linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid. "The coefficients of the trained linear SVM give a measure of how much weight each cell of each block can have in the final discrimination decision" [1].

d. Decision Function Shape: SVM has two decision function shapes: ovo (one vs one) and ovr (one vs rest). Ovo examines each 2-pair class combination while ovr examines one classifier for each class fitted against all other classes.

3.3 Description

When using SVM with ResNet, accuracies from tables below are better than the accuracy of SVM with HOG. Table1 shows that with C becoming greater, the algorithm will have greater testing accuracy. Table2 indicates that different Gamma values will output different test accuracy. Auto Gamma will have about 0.61 accuracy while scale Gamma will have only 0.59 test accuracy (with some overfitting since train and test accuracy are not growing at the same time). Table 3 emphasizes that the RBF kernel performs the best test accuracy while the POLY kernel performs the worst accuracy. But there is no conclusion generated based on Table 4 since decision function shapes don't influence train and test accuracy.

Table 1. Train and Test accuracy on effect of hyperparameter C

C value	0.1	1	2	10	100
Train accuracy	0.640000	0.808000	0.917000	0.955000	0.990000
Test accuracy	0.470000	0.569000	0.612000	0.629000	0.640000

Table 2. Train and Test accuracy on effect of hyperparameter Gamma

Gamma value	Auto	Scale
Train accuracy	0.917000	0.940000
Test accuracy	0.612000	0.590000

Table 3. Train and Test accuracy on effect of hyperparameter Kernel

Kernel Type	rbf	poly	linear	sigmoid
Train accuracy	0.917000	0.320000	1.000000	0.690000
Test accuracy	0.612000	0.170000	0.638000	0.560000

Table 4. Train and Test accuracy on effect of hyperparameter Decision Function Shape

Decision Function Shape	ovr	ovo
Train accuracy	0.808000	0.808000
Test accuracy	0.569000	0.569000

4. FINE-GRAINED IMAGE RECOGNITION VIA VGG

4.1 Research Methods (ML Algorithms)

The machine learning model that will be used on this dataset is VGG from Simonyan's paper [4]. Since the dataset is different species of birds and the problem that are dealing with will be like the subtle differences between the pictures to detect the bird species, CNN will obtain high accuracy in this domain, and VGG is a type of CNN architecture [5]. The input image will pass into the combination of convolution and pooling layers, then they will be flattened and pass into dense layers. After that, the softmax layer is the final layer which is for the output.

4.2 Evaluation (Programming Experiments)

The API will be using Tensorflow Keras because they have many pre-train models ready for use. Since the dataset has different resolution sizes, the project process will contain several steps, such as resizing the image dataset, using TensorFlow Keras pretrained model for training, tuning the hyperparameter, and comparing the output. There are many hyperparameters in this model, but learning rate, optimizer, activation layer, dropout, and regulation will be applied to the model. The imageDataGenerator will be used in this project for preprocessing images, normalization, and data argumentation. There are two versions of imageDataGenerator in this project, the first coding below is without data argumentation and the second one is with extra data argumentation. The preprocess_input uses the Keras library currently, which causes the test accuracy to be only 30%. Once the library changes to Tensorflow.Keras, the test accuracy increases by double because the Keras library and Tensorflow.Keras library are not compatible.

The original dataset from Kaggle is resized to 256x256 and randomly cropped to 224x224 then applied to the model. Since Tensorflow Keras has no such methods in its library, the random crop methods were taken from Jung's blog and his Github [6]. This method will take the input from ImageDataGenerator and randomly crop the image to any size.

Then, the VGG16 model is initialized, compiled and fitted by the shuffle. There are two versions of grouping and customized dense layers. The first approach from Thakur's website groups the layer by sequential methods which will generate some unused trainable parameters, and the second approach eliminates this problem [7].

At the beginning of the project, the image dataset without the data argumentation method is applied to the model. The test accuracy is about 15% which is expected since there is nothing change from the original dataset. After using the data

argumentation method, which is discussed in the introduction, the test accuracy increases to 30%. All these result accuracies are using the sequential approach to build the model. The model applies to different learning rates, the output details are shown in Table 5 below. Due to consideration of overfitting, adding the 0.5 drop-out layer will be the next step, but it ends up increasing only 2% for accuracy. The model also tries to test with different batch sizes and different functions in the last layers, softmax and sigmoid. Unfortunately, these ideas do not increase the test accuracy of the model. Therefore, the model has been adding more data argumentation by the imageDataGenerator, the result shown in Table 6. Compared with no extra data argumentation, the model with extra data argumentation has increased 5% in test accuracy on average.

Since the test accuracy is stable at around 30%, the consideration of using a different approach has been made. In this time, instead of using the sequential to group up the layers, Jung's approach has been used for the model. Currently, there is no unused trainable parameter that shows up in the first approach. However, this change does not solve the low accuracy, the problem is that the processing function is from the Keras library, which is not compatible with the Tensorflow Keras library. When the model was built with the new approach, the test accuracy increased double, it was 30% and 60% in the new model. This model applies to different learning rates, and the output detail shows in Table 7. Most of the test accuracy is between 60% and some of them are very low. The next tuning hyperparameter is the optimizer, the model test with different optimizers which is available in the TensorFlow Keras library. The results table Table 8 shows below. As a result of this table, Adamax acquires the highest test accuracy, and Adagrad follows up, then Adadelta and Adam. This model also tests for dropout, L2 regularization and ELU activation function, but they all end up not increasing the test accuracy anymore. As a result of this project, the best test accuracy is 0.66333, with the second approach, Adamax optimizer, the learning rate of 0.00001, batch size 32, and data argumentation and normalization.

Table. 5 first approach with different learning rate

Learning Rate	Test Accuracy (%)	Train Accuracy (%)
1e-3	29.86	48.54
1e-4	36.77	93.35
1e-5	34.12	62.60
1e-6	33.79	63.26

Table. 6 with extra data argumentation

Learning Rate	Test Accuracy (%)	Train Accuracy (%)
1e-6	38.82	75.42
5e-5	39.53	59.71
1e-5	39.15	72.89
1e-4	38.56	50.56

Table. 7 second approach with different learning rate

Learning Rate	Test Accuracy (%)	Train Accuracy (%)
1e-3	0.518	0.47
1e-4	0.518	0.55
1e-5	62.66	95.17
1e-6	62.99	90.58
5e-5	62.28	95.33

Table. 8 with different Optimizer

Optimizer	Test Accuracy (%)	Train Accuracy (%)
Adam	63.78	96.53
Adadelta	63.83	98.19
Adagrad	65.68	98.74
Adamax	66.33	99.46
Nadam	0.518	0.390
RMSprop	0.518	0.500
SGD	1.364	1.080

5. FINE-GRAINED BIRD IMAGE RECOGNITION VIA RESNET

5.1 Research Methods (ML Algorithms)

For this fine-grained bird recognition task, deep learning is used to realize bird recognition. The convolutional neural network is a kind of deep learning algorithm. Feature extraction and recognition in image recognition tasks can be carried out together. ResNet[8] was proposed in 2015 and won first place in the ImageNet competition classification task, which is a milestone event in the history of CNN. The residual structure is proposed to solve the problem of network degradation. Compared with fine-grained image recognition, it is very important to retain detailed features well [9]. There are five directions in this experiment. 1. Using ResNet50, the shape of the final original fully connected layer is modified from 2048*1000 to 2048*200. 2. Use ResNet50 and connect a fully connected layer after the last layer with a size of 1000*200. 3. Using ResNet101, the shape of the final original fully connected layer is modified from 2048*1000 to 2048*200. 4. Using DenseNet121[10], the shape of the final original fully connected layer is modified from 1024*1000 to 1024*200. 5. Using the ResNet50 without pre-trained by ImageNet to train the model for bird recognition.

First, five experiments are completed to determine a suitable model. By optimizing and upgrading the model, ResNet50 shows the best result on the test dataset when using the conventional default parameters, the result is 75.1%, and the generalization performance of the model is better. In the experiment, it is easy to overfit without using the pre-training model. At the same time, the effect of modifying the dimensionality of the output of the last layer is better than adding layers later.

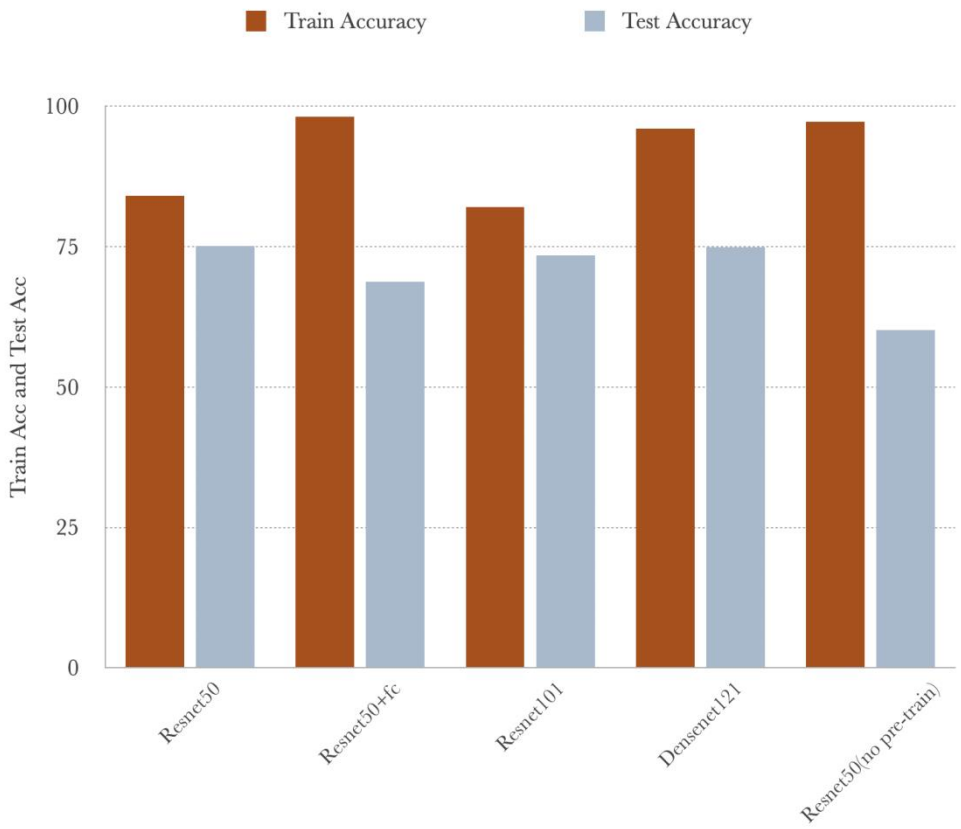


Fig. 2 Comparison of five train methods.On the training data, the ResNet50 and DenseNet121 models have achieved good recognition accuracy. Through the comparison of the first four methods and the last method, it can be concluded that the pre-trained model has a higher accuracy on the test data.

5.2 Evaluation (Programming Experiments)

In the experiment, the programming language is Python3.7, the deep learning framework is Pytorch 1.4, and the GPU used for model training is NVIDIA 2080Ti. It takes 6 hours to run on 2080 Ti and 200 epochs to get the best results. To determine that the model used in this task is ResNet50, four hyperparameters are chosen: Batch_size, Learning_rate, Weight_decay, and K of K-fold cross-validation. Through many experiments, the best result is when the Batch_size is 8, the Learning_rate is 0.0001, the Weight_decay is 0.001, and K is 5. The key parameters and corresponding results are shown in Fig.3.

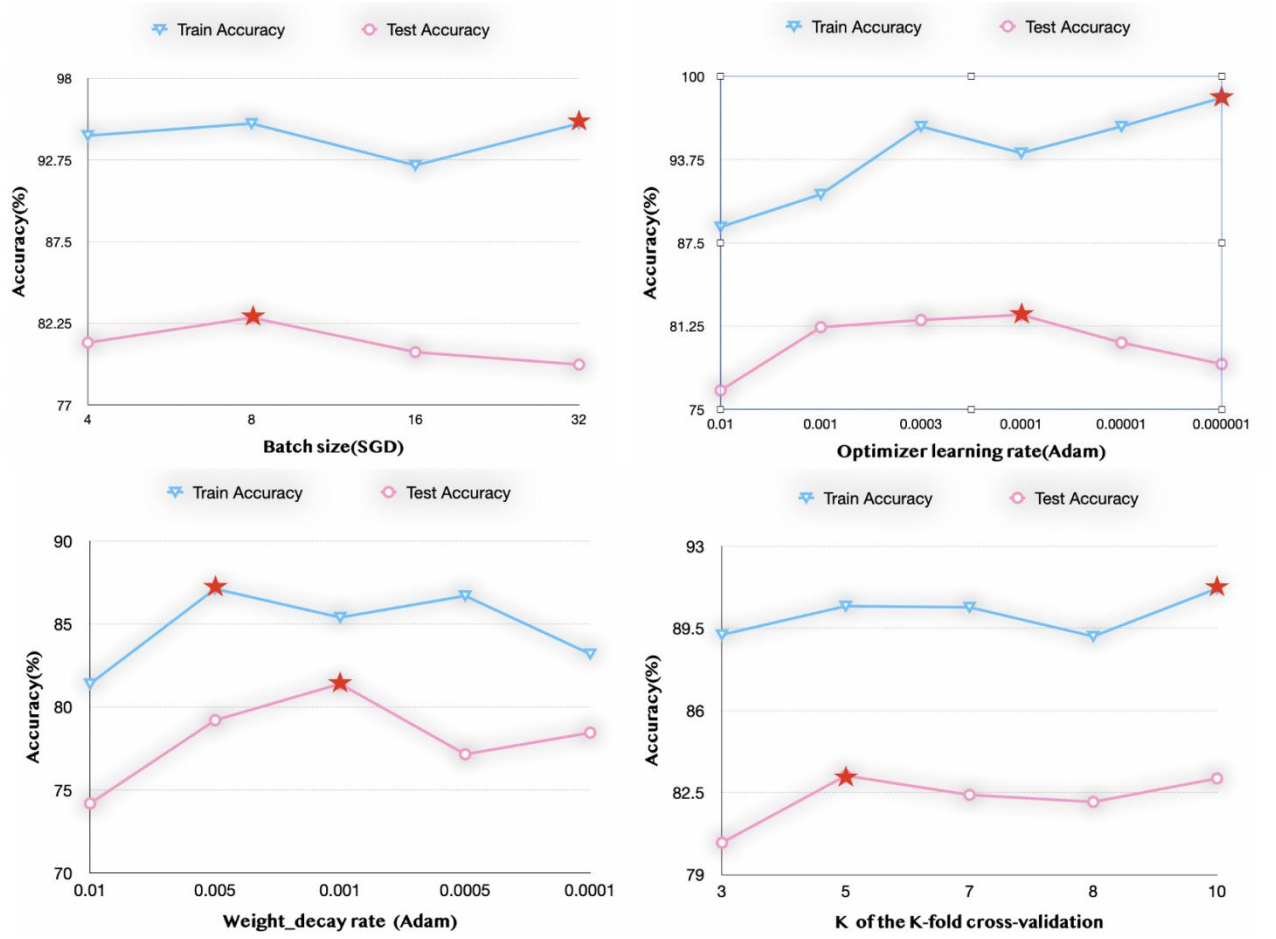


Fig. 3 The result of accuracy by tuning four hyperparameters

Overall, this work shows best results through six steps. Among them, the most profitable steps in the optimization process are data argumentation and changing optimizer. The detailed effect is shown in Fig.4.

Model optimization steps

Optimization Steps	Basic Model	Change Optimizer	Change Learning rete	Use data argumentation	Change Loss_function	Use Weight_deacy	Use Cross_valiadation
Model selection	Resnet50	Resnet50	Resnet50	Resnet50	Resnet50	Resnet50	Resnet50
Optimizer	Adam	SGD	SGD	SGD	SGD	SGD	SGD
Learning_rate	0.001	0.001	0.0001	0.0001	0.0001	0.0001	0.0001
Argumentation	NO	NO	NO	YES	YES	YES	YES
Loss_functiuon	CrossEntropy	CrossEntropy	CrossEntropy	CrossEntropy	CrossEntropy+ Center_loss	CrossEntropy+C enter_loss	CrossEntropy+ Center_loss
Weight_deacy	NO	NO	NO	NO	NO	YES	YES
Cross-validation	NO	NO	NO	NO	NO	NO	YES
Test Accuracy	75.1%	77.1%(+2.00%)	78.21%(+1.11%)	80.61%(+2.400)	81.71%(+1.10%)	82.62%(+0.91%)	83.22%(+0.60%)

Fig. 4 Six optimizations for the ResNet model. After using the pre-trained model for training, we found that SGD optimizer performs better when fine-tuning. The biggest improvement to the model is data argumentation. The continuously optimized and improved model achieved a recognition accuracy of 83.22% on the test data.

6. CONCLUSION

With the fast development of algorithms and supercomputers, image recognition is becoming more precise and efficient. In the research, three main methods are studied. HOG was an algorithm that was invented over ten years ago, which is not suitable for the same species with minor differences in appearance. But HOG would be a good classification for simple species with giant differences. Accuracy performed by SVM with ResNet is higher than the accuracy performed by SVM with HOG. Therefore, ResNet with SVM is a better choice to identify different categories. VGG is a time-consuming project, but it is an ideal picture recognition algorithm. If more time is provided, the decay learning rate method will be applied for it since the learning rate has much impact on the testing accuracy. Through continuous optimization and improvement, the recognition accuracy rate of the model was 83.22%, which is 4% higher than the best open-source code on Kaggle. Finally, through the improvement 4and optimization of ResNet, the task of identifying fine-grained bird pictures is basically realized. This research provides a possible way to protect endangered species by image recognizing species so that people can calculate population by frequencies of photos taken by infrared cameras. People can protect endangered species with appropriate measures. Image recognition will maintain a species diversity world.

REFERENCES

[1] Navneet Dalal, Bill Triggs (2005). Histograms of Oriented Gradients for Human Detection. International Conference on Computer Vision & Pattern Recognition (CVPR '05), San Diego, United States. pp.886–893, 10.1109/CVPR.2005.177. Inria-00548512

[2] S Tukra (2018). Object-detection-via-HOG-SVM. https://github.com/SamPlvs/Object-detection-via-HOG-SVM.

[3] Jianlong Yuan (2017). HOG-SVM-Python. https://github.com/jianlong-yuan/HOG-SVM-python.

[4] Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, .

[5] Morton, D. (2020, July 4). This model is for the birds. Medium. Retrieved September 20, 2021, from https://towardsdatascience.com/this-model-is-for-the-birds-6d55060d9074.

[6] JK Jung (2018, April 16). Extending Keras' ImageDataGenerator to Support Random Cropping. https://jkjung-avt.github.io/keras-image-cropping/.

[7] Thakur, R. (2020, November 24). Step by step VGG16 implementation in Keras for beginners. Medium. https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c.

[8] He K, Zhang X, Ren S, et al (2016). Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition. Pp.770-778.

Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

[9] Wen, Y., et al (2016). A discriminative feature learning approach for deep face recognition. In European conference on computer vision. Springer.

[10] Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.