Comparing Three Models' Performances on Fine-grained Classification

Anni Li
a5li@ucsd.edu

Ruoxuan Li
ruli@ucsd.edu

Abstract

Our project aims to compare the performances of three deep-learning models on a fine-grained classification task of classifying 200 bird species images using the CUB-200-2011 dataset. The three models include two convolutional neural networks (CNN) on both full and masked images (with the background removed) and one pre-trained ResNet50 model [1]. However, due to the heavy GPU requirements of the model, with one epoch taking 10 minutes to run on a 3070 Ti GPU, the training and tuning are difficult for us, so the results did not meet our expected level of accuracy. Additionally, even though the ResNet50 model required significant computational resources, we estimated that the model was the most effective among the three as we were able to observe a steady reduction in the loss function with each epoch, indicating that the model has the potential for improved accuracy with additional training time. Future work in this area could involve exploring different ways to optimize the GPU requirements for bird classifiers or exploring alternative models that can deliver comparable accuracy with fewer computational resources. In summary, by comparing these three models, we found that the traditional CNNs with simple structures could not perform fine-grained classification well, because their ability to effectively extract important features is limited. The ResNet-50 model, both because it was pre-trained on a larger dataset (ImageNet) and it has a more complicated structure, can finish this task much better than traditional CNNs. Our code for this project is publicly available on Github: https://github.com/AnniLi1212/Fine_Grained_Bird_Classification.git

1 Introduction

1.1 Motivation

Bird classification has always been an interesting and challenging task in the scientific field. Different species of birds play various essential roles in an ecosystem and are significantly important for wildlife conservation. The categorization of birds strengthens our understanding of the species and improves the efficiency of the reservation programs. In the past, biologists had to manually label the birds by comparing the field guide and the images of the birds, which could be exhausting and time-consuming. Nowadays, with the development of artificial intelligence and computer vision, we become able to perform complex bird-classification tasks with the help of deep learning models. While there are many types of bird classification tasks with different input features such as the birds' soundtrack, text descriptions, and photos of birds, in our project, we are particularly interested in classifying the birds using images as we expect this task will contribute positively to the conservation of wildlife and the environment. Besides that, one of the authors enjoys bird-watching and quite often relies on bird-classification applications such as Merlin Bird ID for effective categorization of a bird image shot. Thus, we determined to perform an experiment in which we compared the performance of different models on CUB-200-2011 [2], a dataset that consists of over 10,000 images of 200 species of birds.

The task of classifying birds using images is a type of fine-grained classification. By definition, fine-grained classification is a sub-field of object recognition that aims to distinguish subordinate categories within every entry-level category. We hope by training and comparing different models on the bird image datasets, we will gain more experience in training some widely-adopted deep learning models and a deeper understanding of the challenging fine-grained classification task. We hope our work could provide some insights for future directions on improving the performance of bird image classification tasks.

1.2 Related Work

Fine-grained visual classification (FGVC) is a subfield of computer vision focused on distinguishing between highly similar object categories, often belonging to the same superclass, such as bird species, car models, or dog breeds. The challenge in FGVC lies in the subtle differences between categories, which often require domain-specific knowledge and expertise to differentiate. The breakthrough of deep learning, particularly convolutional neural networks (CNNs), has significantly improved the performance of fine-grained classification. CNNs are capable of learning hierarchical features from raw images without the need for manual feature engineering or part annotations [3]. Models such as AlexNet, VGG, and ResNet have been employed for FGVC tasks with great success [4].

The paper "Deep Residual Learning for Image Recognition" by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun introduced the Residual Network (ResNet) architecture, presenting a novel approach to addressing the vanishing gradient problem in deep networks, enabling the training of very deep networks while maintaining high performance. Its applications in fine-grained classification is also demonstrated.(He et al., 2016) The pre-trained ResNet-50 model, trained on the ImageNet dataset, can be fine-tuned on smaller datasets for fine-grained classification tasks, achieving high accuracy with less training data and time. Examples of successful applications include the CUB-200-2011 bird species dataset, the Stanford Cars dataset, and the Oxford Flowers dataset.(He et al., 2016; Wah et al., 2011) In this project, we are going to compare the performance of three models in fine-grained classification: the traditional CNN, the CNN using masked image, and the pre-trained ResNet-50 model.

2 Dataset

The dataset we chose comes from the Kaggle platform (https://www.kaggle.com/datasets/veeralakrishna/200-bird-species-with-11788-images?resource=download) and is named Caltech-UCSD Birds-200-2011. This dataset is an extended version of the CUB-200 dataset and consists of 200 bird species with 11,788 images. In this dataset, each image has 15 part locations, 312 binary attributes, and 1 bounding box. These informations are stored separately in txt files.

To process the data, we first unzip the .tgz format file which contains several important data files. Once unzipped, we had several .txt files containing labels (bird species), image IDs, and training flags (‘is_training‘). To prepare the dataset for processing for our models, we read all the information from these .txt files and stored them into separate columns in a Pandas data frame. We split the data based on the ‘is_train‘ column. Next, we define a Pytorch class called CUBdata that takes into the Pandas data frame and a transformation object transform. Under the class, we implemented two functions ‘len‘ and ‘getitem‘. While ‘len‘ function returns the number of samples in our data frame, the ‘getitem‘ function returns the image that the ID in the data frame corresponds to. The transform we defined here first resized the image to 448x448 and then converted it into a tensor. We then normalize the image by calling the ‘normalize_image‘ function to obtain the normalized pixel values for each image tensor. Following this data-cleaning process, we fed our Pandas data frame into the CUBdata and went through the transformation process, and obtained the dataset used for training. For the pre-trained ResNet50 model, we resized the images to 224x224 and did some random rotations and flips on the images before converting them to vectors.

3 Methods

3.1 Models

3.1.1 Model 1

Mechanism: This model is a convolutional neural network model that is trained on the full bird images.

Construction: We defined a batch size of 8 for each DataLoder. Then we construct our CNN with two convolutional layers, a max-pooling layer, and a fully connected layer. We used ReLU as the activation function in this basic neural network. For the first layer, we defined it to take into a 3 channels input and output 16 channels using a 3x3 kernel with padding 1. Then we called our activation function ReLU and performed max-pooling of size 2x2 on the output. The result was then fed into the second layer and outputted into 32 channels. The ReLU function was called again and the same 2x2 max-pooling was performed on these features. Finally, the output of the layer was flattened into a one-dimensional tensor and passed through a fully connected layer with 32x112x112 input features and 200 output features and underwent the activa-

tion function. The result was returned as the final output of the neural network.

3.1.2 Model 2

Mechanism: This model is a convolutional neural network model that is trained images which only include bird bodies using the information from parts.txt.

Construction: First, we masked out the background according to the outermost part points and ignore one outermost point to prevent outlier points, since the part points are marked manually. Then we extracted the main body of the bird from the image. For the masked image, we constructed a more sophisticated three-layer neural network with batch normalization, max pooling, two fully connected layers, and dropout layers for regularization. The first convolutional layer took a 3-channel input and applied a 3x3 kernel with padding 1 and output to 16 channels. The output was then passed through batch normalization and the leaky ReLU activation function. The second layer took into the output of the previous layer and applied a 3x3 kernel with padding 1 and batch normalization and the leaky ReLU again. The final layer was constructed using the same logic but outputted 64 channels. After the convolutional layers, the output was flattened and passed through two fully-connected layers with leaky ReLU and random dropout. Batch normalization and dropout regularization were used to prevent overfitting and improve generalization.

3.1.3 Model 3

Mechanism: This model uses the ResNet50 model that is pre-trained on the ImageNet dataset[1], which contains more than one million images across 1000 different classes. By leveraging the knowledge gained from this pre-training, the ResNet-50 model can be fine-tuned on smaller datasets, such as the CUB-200-2011 dataset, to achieve high accuracy with relatively less training data and time.

Construction: The ResNet-50 model is a deep convolutional neural network designed for image classification tasks. The model is built on the Residual Network (ResNet) architecture, which employs residual connections to enable the training of deeper networks. The structure of this model can be divided into several parts:

Initial Convolutional Layer: The input image passes through a 7x7 convolutional layer with 64 filters, stride of 2, and padding of 3. This is followed by batch normalization and a ReLU activation function. The output feature maps are then downsampled using a 3x3 max-pooling layer with a stride of 2 and padding of 1.

Residual Blocks: The core of the ResNet-50 architecture consists of 4 groups of residual blocks, where each group contains a different number of bottleneck blocks. A bottleneck block is a composition of three convolutional layers. First, a 1x1 convolutional layer is used to reduce the number of channels followed by batch normalization and a ReLU activation function. Next, a 3x3 convolutional layer is applied to extract spatial features, followed by batch normalization and a ReLU activation function. Finally, a 1x1 convolutional layer is used to restore the number of channels and increase them by a factor of 4 followed by batch normalization.

The output of the final layer in each bottleneck block is added to the input feature map (residual connection) before passing through a ReLU activation function. After passing through the four residual blocks, the output feature maps are passed through an adaptive average pooling layer to reduce their spatial dimensions to 1x1. The resulting feature maps are flattened into a 1D vector, which is then passed through a fully connected layer to produce the final class probabilities. The number of output units in the fully connected layer corresponds to the number of classes in the classification task.

3.2 Training

We applied optuna for the first two models to select the best hyperparameters. Since the process takes too long, we set optuna to run 5 epochs with 6 trials. The selected hyperparameters are then passed into the model for training. Through manual practices, we found the cross entropy loss is the best loss function and the SGD is better than Adam as optimizers for the first two models, as Adam would cause more severe over-fitting. The learning rate and momentum are chosen by optuna. For model 1, lr=0.0036 and momentum=0.9129; for model 2, lr = 0.0066 and momentum=0.6489.

During the training and validation loop, we applied an early stopping with patience=10 to prevent the model from overfitting. If the validation loss is larger than the best validation loss ten times, the training process would stop. The model with the least validation loss is then copied as the best model

to be tested on the test dataset.

For model 3, we did not use optuna to select best hyperparameters, because it takes too long to finish. We finally chose cross-entropy loss as the loss function and Adam as the optimizer with learning rate=0.001 and weight decay=0.0001 to prevent overfitting. We also set the validation patience to 10 to trigger the early stopping and copy the best model to be tested on the test dataset.

3.3 Evaluation Metrics

We used metrics including loss, accuracy, precision, and recall to evaluate the performances of our models as these dimensions are important in a bird classification task. We used the cross-entropy loss to calculate the loss for each model. The accuracy was calculated by taking the portion of correct predictions among all the true labels. We also include precision and recall in our metric because precision represents the fraction of correctly identified bird species out of all the bird species predicted by the model and recall represents the fraction of correctly identified bird species out of all the bird species in the dataset. We included these two metrics in our experiment as it provides a more comprehensive review of the performances of the models.

4 Results

The results of our experiment and comparison indicated that the ResNet50 was the most suitable model for the fine-grained bird classification task. Among the three models, it was the most effective model with a precision of 0.334, a recall of 0.290, and an accuracy of 28% on the test set after 77 epochs of training. The second most effective model for the task in our experiment was Net_2 which only trained on masked bird images, with a precision of 0.075, a recall of 0.070, and an accuracy of 7%. We observed some improvement in Net_2 when we removed the backgrounds from the bird images for training with a more sophisticated construction of the neural network. This improved network outperformed our Net_1, which had a precision of 0.001, a recall of 0.014, and an accuracy of 1% after 11 epochs on a general look. Overall, we observed the fastest reduction in the loss function of the ResNet50 model, this finding reconfirmed that ResNet50 was relatively effective in performing fine-grained classification like this one. The performance of each model could be found in Table 1. The change of the loss function was shown in Graph 1, Graph 2, and Graph 3.

Model	Accuracy	Precision	Recall
Net_1	1%	0.001	0.014
Net_2	7%	0.075	0.070
ResNet50	28%	0.334	0.290

Table 1: shows the performance for each model. Please note that Net_1 was trained on 11 epochs, Net_2 was trained on 17 epochs and ResNet50 was trained on 77 epochs. We observed an improved performance when we add more complexity to the model.

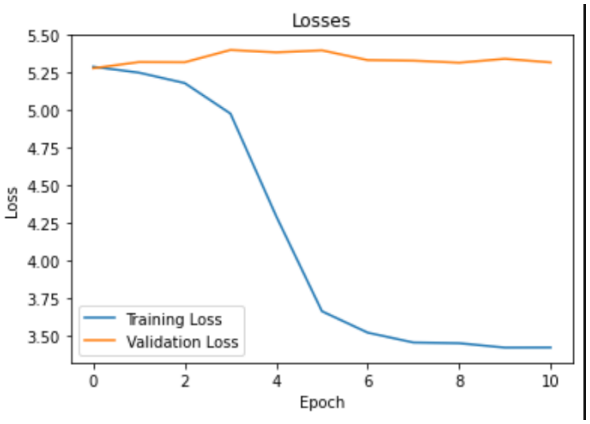

Figure 1: Training loss curve and validation loss curve on the training of Net_1.

5 Limitation

We must acknowledge that we might have underestimated the complexity of the task and we failed to discuss the feasibility of this project with the instructor team while planning. Though bird classification is a widely-performed classification task, it didn't imply anything about the complexity and hardware requirements of this task. It is also im-

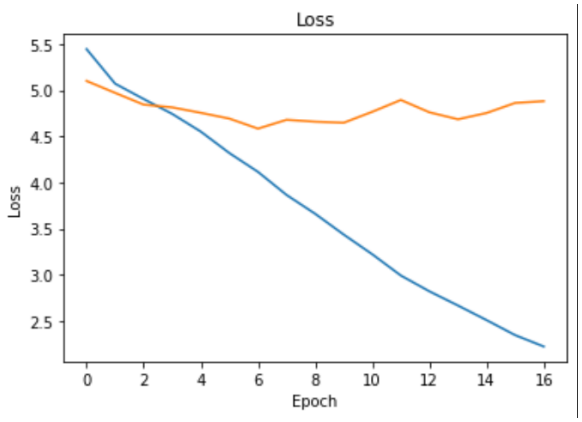

Figure 2: Training loss curve and validation loss curve on the training of Net_2.

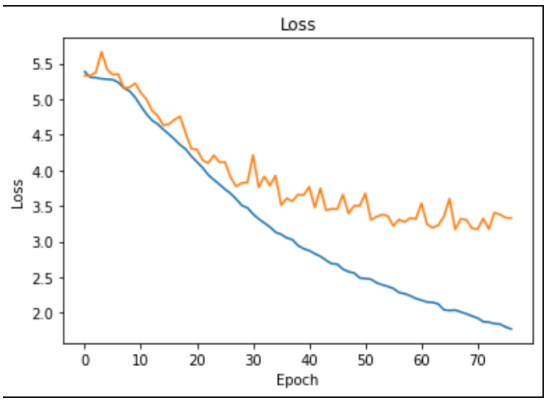

Figure 3: Training loss curve and validation loss curve on the training of ResNet50.

portant to note that the performances of all models were greatly restricted to the GPUs that were accessible to us. We ran into huge difficulties while training the dataset due to the heavy GPU requirements from all three models. Thus, the results we obtained might not be an accurate representation of the performance of the models as we could not compare the performances of the models on the same basis with the ResNet50 (77 epochs) being relatively faster compared to our models (11 epochs and 17 epochs respectively).

6 Future Direction

In the future, we hope to reconstruct the current state-of-art models such as MetaFormer to approach such a fine-grained classification task. We are interested in seeing a comparison between the performances of these models. Other machine-learning tasks related to bird classification are also worth attempting, such as using audio to classify birds. In summary, we believe that there is a need for optimization of models that are suitable for fine-grained classification to reduce the computational powers so the models are more accessible to the general public.

7 Conclusion

In this project, we constructed three CNN models using different strategies to perform fine-grained classification on the CUB-200-2011 dataset to distinguish different bird species images. We compared the performance of 1. traditional CNN, 2. CNN using masked images, and 3. Pre-trained ResNet-50 model on this task, and found that the CNN using masked images indeed has an improvement in accuracy, precision, and recall compared to traditional CNN. The model that performs the best is the pre-trained ResNet-50 model, which reaches 28% accuracy on test images with precision=0.334 and recall=0.290. From this project, we found that traditional CNNs with simple structure could not do fine-grained classification well, because their ability to effectively extract important features is limited.

8 Contribution

Both authors actively collaborated and communicated throughout the project:
Anni Li: Data processing, models construction, training, final report
Ruoxuan Li: Project initiation, validation, metrics, final report, presentation slides

References

[1] He, K., Zhang, X., Ren, S., Sun, J. (2015). Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385.

[2] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S. (2011). The Caltech-UCSD Birds-200-2011 Dataset. California Institute of Technology. http://www.vision.caltech.edu/visipedia/CUB-200-2011.html.

[3] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[4] Simonyan, K., Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.