Midterm Project for DATA620004: Neural Network and Deep Learning

Yuanye Liu
School of Data Science
Fudan University
yuanyeliu@fudan.edu.cn

Abstract

Deep learning has achieved remarkable success across a wide range of tasks. The pre-train technique has been applied to improve the performance and reduce the training cost. In this project, we explore the pre-train technique in detail to gain a deeper understanding. We evaluated it on the CUB dataset. In addition, we implement objection detection framework. We employ mmdetection architecture to train Faster-RCNN and YOLOv3 on VOC dataset. The source code and trained model weights are accessible at https://github.com/HenryLau7/Midterm/ and google drive, respectively.

1 Introduction

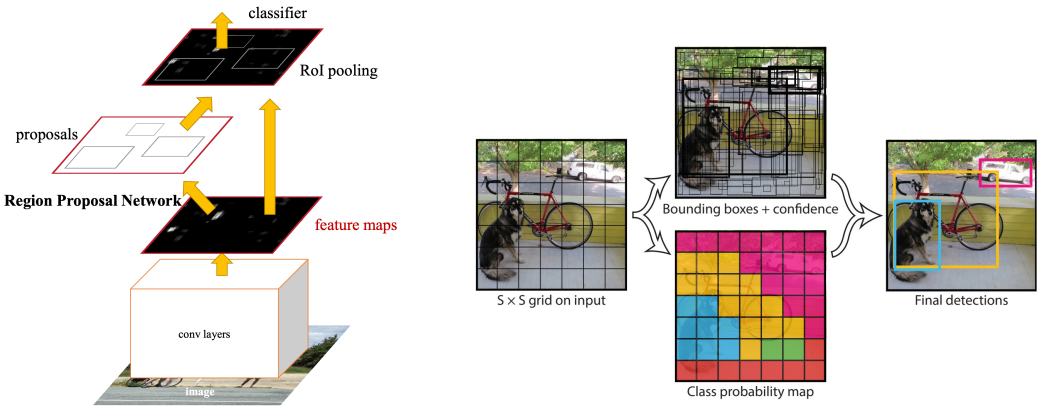
Deep learning has emerged as a pivotal technology in various machine learning tasks, significantly enhancing the performance of models across a wide range of applications [3]. However, training deep neural networks from scratchstarting with random initializationdemands extensive computational resources and time. Pretraining has thus become a critical technique to accelerate the training process and improve model performance. By initializing a model with parameters learned from a large dataset such as ImageNet [2], pretraining enables faster convergence and often leads to better generalization on specific tasks.

In this project, we explore the role of pretraining in the context of image classification. Specifically, we conduct an ablation study on CUB dataset [4], to evaluate the impact of pretraining on the performance of a neural network. We begin by fixing the early layers of the network with parameters pretrained on ImageNet, then train the fully-connected layers, and finally, fine-tune the entire network with a slower learning rate. This approach helps us understand the contribution of pretrained parameters to the overall performance and efficiency of the model training process.

Object detection, another critical area of computer vision, involves identifying and localizing objects within an image. Unlike classification, which assigns a single label to an image, object detection provides bounding boxes and labels for multiple objects within an image. This task is more complex as it requires precise localization in addition to classification. In this project, we conduct experiments using two state-of-the-art (SOTA) object detection models: Faster R-CNN and YOLOv3, on the VOC0712[1] dataset. Faster R-CNN employs a region proposal network to generate potential bounding boxes and refines these proposals for accurate detection, while YOLOv3 divides the image into a grid and predicts bounding boxes and class probabilities directly from the grid cells.

Our objective is to compare the performance of these models in terms of accuracy. By examining the effects of pretraining in both classification and object detection tasks, we aim to provide comprehensive insights into the benefits and limitations of this technique in different contexts.

The outline of this report is as follows. In Sec 2, we discuss the pretraining technique and the details of Faster R-CNN and YOLOv3. In Sec 3, we evaluate the models on datasets, including the classification model on the CUB dataset to explore pretraining, and object detection on VOC0712.



(a) Framework of Faster-RCNN (a) Framework of YOLO

Figure 1: The framework of Faster-RCNN (a) and YOLO (b). Faster-RCNN has two stages, where the first stage is a RPN to generate proposals, and the second stage is to classify these proposals. YOLO has only one stage, where it defines a grid to regress the bounding box.

Additionally, we conduct evaluations on specific samples to provide demonstrations. In Sec 4, we present our conclusions.

2 Method

In this section, we introduce the detailed methodologies employed in this project, including the pretraining technique, the Faster R-CNN model, and the YOLOv3 model.

2.1 Pretrain

Pretraining plays a crucial role in our methodology by providing a strong initialization for the subsequent fine-tuning classification tasks. We utilize a pretrained ResNet-34 model, leveraging weights that have been trained on the ImageNet dataset. This initialization helps in speeding up the convergence and improving the overall performance of our models. The pretrained model is further fine-tuned on our specific dataset to adapt the learned features to the target domain. During pretraining, standard data augmentation techniques such as random cropping, flipping, and normalization can be applied to enhance the robustness of the model.

2.2 Faster-RCNN

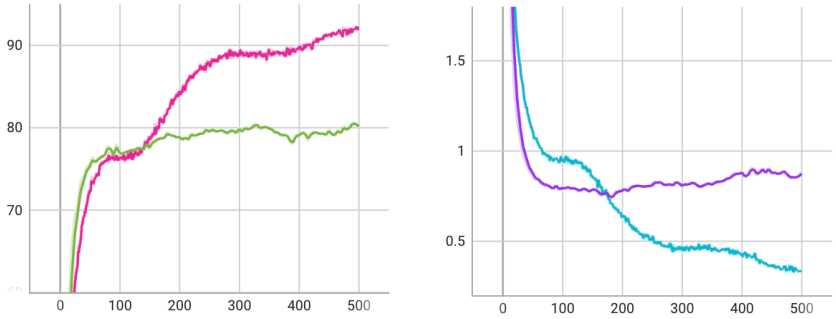
For object detection, we employ the Faster R-CNN model, which is popular for its accuracy and efficiency. As shown in Fig. 1(a), the Faster R-CNN architecture consists of a Region Proposal Network (RPN) that generates region proposals, followed by a Fast R-CNN detector that classifies and refines these proposals. Our implementation utilizes a ResNet-50 backbone, pretrained on ImageNet, to extract rich feature representations from the input images. The model is trained using the PASCAL VOC dataset, which provides annotated bounding boxes for various object categories. We adopt a learning rate schedule with an initial learning rate of 0.001, which is decreased by a factor of 10 at specified epochs. The model is trained for 12 epochs, with a batch size of 16 and an Adam optimizer to minimize the cross-entropy loss and bounding box regression loss.

2.3 YOLOv3

YOLOv3 (You Only Look Once version 3) is another powerful object detection model used in our experiments. Unlike Faster R-CNN, YOLOv3 formulates object detection as a single regression problem, directly predicting bounding boxes and class probabilities from full images in one evaluation, as shown in Fig. 1(b). This makes YOLOv3 significantly faster while maintaining competitive accuracy.

Table 1: The available parameters for tuning.

Parameters	Values
Learning rate	$3e-4, 3e-5$
Pretrained	w/wo
Cutout	w/wo
Dropout	w/wo



(a) Accuracy (b) Loss

Figure 2: Training procedure of ResNet34 on CUB dataset, including the accuracy curve and loss curve. Visualized through tensorboard.

3 Experiment

In the experiment part, we divided into three parts, classification on CUB dataset in Sec. 3.1, faster r-cnn on VOC in Sec. 3.2 and YOLOv3 on VOC in Sec. 3.3.

3.1 Classification task on CUB dataset

In this part, we evaluated our classification model on the CUB dataset, especially the ablation study on several hyper-parameters.

3.1.1 CUB Dataset

The Caltech-UCSD Birds (CUB) dataset stands as a pivotal resource in the domain of fine-grained visual categorization, specifically tailored for bird species classification tasks. Introduced by Wah et al.[4], this dataset encapsulates the rich biodiversity of avian species native to North America, comprising over 11,000 high-resolution images meticulously annotated with species labels.

3.1.2 Implementation details

In the classification task, we utilize ResNet34 as the backbone, pretrained on ImageNet. We employ a batch size of 256 and train the model for 500 epochs. The Adam optimizer is used with a learning rate of 3×10^{-4} for the last layer, while the learning rate for the other layers is set to 3×10^{-5}, which is one-tenth of the last layer's learning rate. Additionally, we apply a weight decay of 3×10^{-5}. All experiments are conducted on an NVIDIA TITAN RTX GPU.

As shown in Table 1, we tuned various hyperparameters, including the learning rate, pretraining status, cutout, dropout, and the architecture of the backbone.

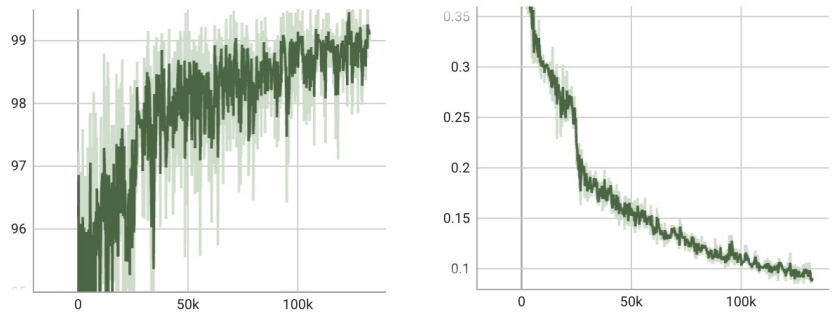
3.1.3 Results

As shown in 2, we train the ResNet34 until converging to a stable accuracy and loss.

As shown Table. 2, we can find that, the setting with learning rate $3e-4$, pretrain, dropout achieves best performance, an 83.12% accuracy on test set. Notably, the cutout augmentation didnot work,

Table 2: The results on VOC0712 with different hyper-parameters combination.The method with pretrain, cutout augmentation and Dropout achieved best performance 83.12%.

Learning rate	Pretrain	Cutout	Dropout	Model	Test Accuracy ↑
$3e-4$	✔	✗	✔	ResNet34	$\mathbf{83.12\%}$
$3e-4$	✔	✔	✔	ResNet34	82.27%
$3e-4$	✔	✔	✗	ResNet34	81.72%
$3e-5$	✔	✔	✔	ResNet34	78.20%
$3e-4$	✔	✔	✔	ResNet18	77.14%
$3e-4$	✗	✔	✔	ResNet34	43.17%



(a) Accuracy (b) Loss

Figure 3: Training procedure of Faster-RCNN on VOC0712 dataset, including the accuracy curve and loss curve. Visualized through tensorboard.

which may result from the characteristic of the dataset. The bird classification relies on the critical information which is detailed, so the random cutout may ruin the feature map and furthermore the performance.

3.2 Faster R-CNN on VOC

In this part, we trained a Faster R-CNN model on VOC0712 dataset with the mmdetection architecture.

3.2.1 Pascal VOC Dataset

The PASCAL VOC (Visual Object Classes) dataset stands as a cornerstone resource in the realm of computer vision, particularly renowned for its role in object detection, recognition, and segmentation tasks. Originally introduced by Everingham et al. [1], this dataset represents a diverse array of object categories across various domains, facilitating comprehensive research and benchmarking in the field.

3.2.2 Implementation details

In the object detection task, for Faster R-CNN we trained 16 epochs, the RPN network backbone utilized a pretrained ResNet50.

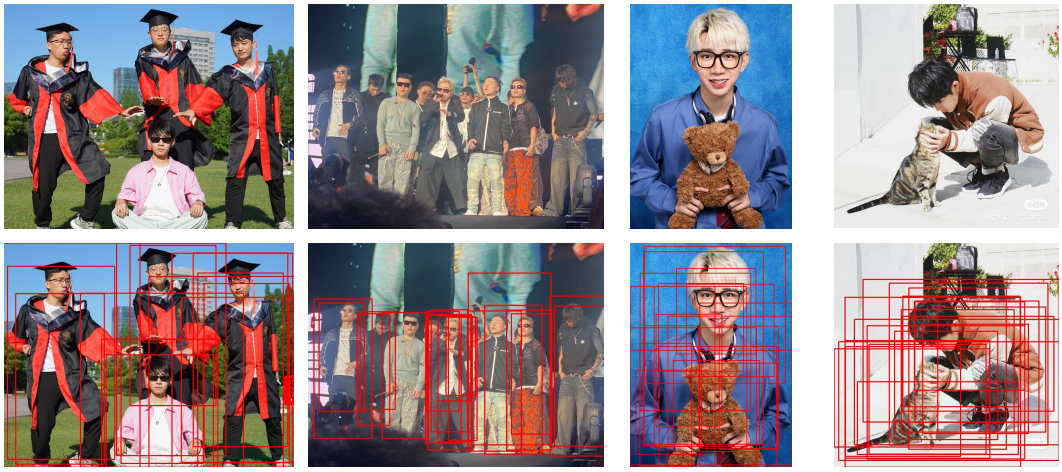
3.2.3 Results

As shown in Fig. 3, we trained the Faster R-CNN model until it converged to a stable accuracy and loss. Additionally, as illustrated in Table 3, we evaluated our trained Faster R-CNN model on the test set and achieved good performance.

Notably, small objects such as bottles, potted plants, and birds are challenging to detect, whereas larger objects are easier to identify.

Table 3: Object Detection Evaluation Metrics (Transposed)

Metric	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
gts	155	177	243	150	252	114	625	190	398	123
dets	415	487	560	486	450	323	1198	525	1986	488
Recall	0.826	0.898	0.761	0.713	0.516	0.842	0.790	0.958	0.837	0.935
AP	0.772	0.790	0.653	0.604	0.433	0.772	0.705	0.861	0.634	0.708
Metric	**diningtable**	**dog**	**horse**	**motorbike**	**person**	**pottedplant**	**sheep**	**sofa**	**train**	**tvmonitor**
gts	112	257	180	172	2332	266	127	124	152	158
dets	698	890	652	692	5197	750	486	899	498	474
Recall	0.911	0.926	0.939	0.924	0.823	0.707	0.756	0.952	0.928	0.861
AP	0.654	0.824	0.851	0.802	0.747	0.486	0.599	0.730	0.854	0.748



(a) Four people (b) CDC rappers (c) Man and bear (d) Qianxi and Cat

Figure 4: Visualization of testing samples and proposal boxes.

3.2.4 Visualization of Proposal Boxes in the First Stage

Notably, we also visualized the proposal boxes generated in the first stage. As shown in Fig. 4, we selected four photos that are not part of the VOC dataset but contain objects present in the VOC dataset to conduct inference.

We observed that many proposal boxes were generated to cover the target objects, preparing them for classification in the second stage, thereby completing the object detection task.

3.3 YOLOV3 on VOC

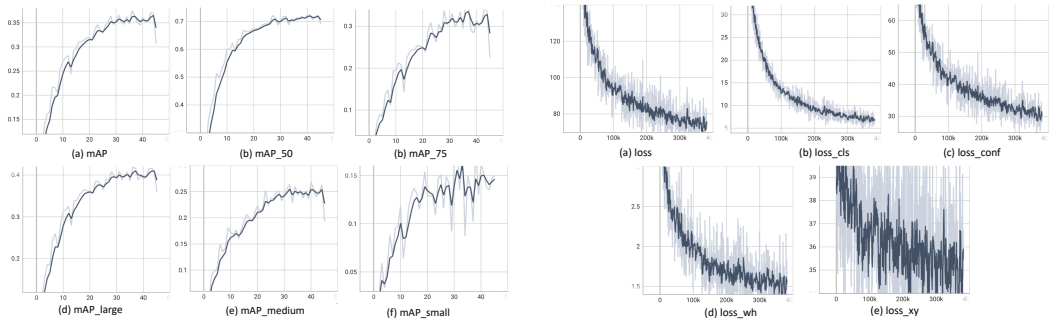
In this part, we evaluated a YOLOv3 model on VOC dataset.

3.3.1 Implementation details

We employ the YOLOv3 architecture with a Darknet-53 backbone, pretrained on ImageNet. The model is trained on the PASCAL VOC dataset using a batch size of 64 and a learning rate of 0.001. Training is conducted for 50 epochs, employing data augmentation techniques such as random scaling, cropping, and flipping to improve generalization. The loss function used in YOLOv3 combines mean squared error for bounding box regression and binary cross-entropy for class prediction.

3.3.2 Results

As shown in Fig.6, the mAP curve and loss curve, we can find that we could train into a considerable stable stage, where we could obtain not that satisfactory results.



(a) mAP curve (b) loss curve

Figure 5: Results

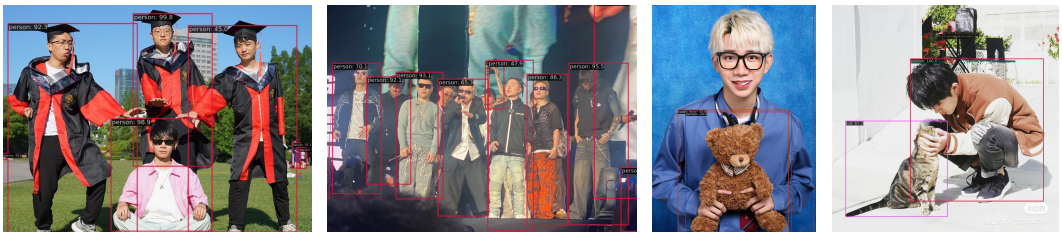
Figure 6: Training procedure of object detection of YOLOv3 which is trained on VOC0712.

3.3.3 Visualization

We further input the four samples for inference using the trained Faster R-CNN and YOLOv3 models, as illustrated in Fig. 7.

The performance of Faster R-CNN is notably better, especially for the person detection boxes, where Faster R-CNN provides higher confidence scores for the correct objects. Notably, in the third figure, YOLOv3 failed to detect the person, which is usually an easier object to identify.

(a) Inference by Faster-RCNN



(b) Inference by YOLO-v3

Figure 7: Visualization of prediction results of Faster RCNN and YOLOv3.

4 Conclusion

In this project, we conducted comprehensive experiments in two key areas: pretraining techniques and object detection. Firstly, we explored the role of pretraining in classification tasks using the CUB dataset. By leveraging a pretrained ResNet34 model, we investigated how initialization with ImageNet weights impacts performance on fine-grained bird species classification. Our ablation study further examined the effects of additional techniques such as cutout augmentation and dropout.

In the object detection part of our project, we evaluated the performance of two state-of-the-art models: Faster R-CNN and YOLOv3, on the VOC0712 dataset. Our findings indicate that Faster R-CNN generally outperforms YOLOv3, particularly in detecting smaller objects like bottles, potted plants, and birds. We also visualized the proposal boxes generated in the first stage of Faster R-CNN, highlighting its capability to accurately localize objects early in the detection process.

Overall, our results demonstrate the significant benefits of pretraining and the effectiveness of carefully selected augmentation and regularization techniques in improving model performance for both classification and object detection tasks.

Acknowledgement

Thanks for the inspiring teaching of Professor Li Zhang and the correcting of the TA.

References

[1] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[3] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[4] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.