Optimizing Bird Species Classification- Self-Supervised vs. Supervised Models

Dor Danino | Tal Polak

Technion - Israel Institute Of Technology

16/08/24

1. Introduction

This project aims to compare the performance of self-supervised and supervised models across different optimizers for fine-grained image classification, with a particular focus on distinguishing between similar bird species[1]. By leveraging transfer learning, we can utilize new pre-trained models and cutting-edge optimizers to achieve high accuracy in classifying these closely related species. This project and the dataset used are both publicly available.

Motivation

We undertook this project to explore which model, DINOv2[2] (a self-supervised model) or YOLOv8[3] (a supervised model), performs better in fine-grained image classification, while testing which optimizers gave the best result.

Previous Works

Prior works have created a great deal of models designed for this particular dataset[5]. In particular, Bird Detection and Species Classification: Using YOLOv5 and Deep Transfer Learning Models compared the performance of YOLOv5, VGG19, InceptionV3, and EfficientNetB3, using optimizers such as AdamW, SGD, and AdamMax[4].
However, works measuring the performance of newer models, such as DINOv2 and YOLOv8, do not currently exist.

2. Method

In this section, we'll discuss the models, training, optimizers, and hyperparameters.

DINOv2

DINOv2 (self-DIstillation with NO labels v2) is an open-source self-supervised framework that leverages vision transformers. Developed by researchers at Meta, it was trained on 142 million unlabeled images. We used the DINOv2 large image classification model, which contains 300 million parameters, as a feature extractor and trained the last layer as a classification layer. Using this model, we trained a total of 57 different models, each utilizing a unique combination of parameters, optimizers, or augmentations.
Each one of the models was trained with one of the following optimizers: SGD, Adam, AdamW, RMSProp, AdamN, AdamR, schedule-free AdamW and, schedule-free SGD[6]. For each optimizer at least six models were trained, all training for 3 epochs.
All optimizers used their default settings, excluding the learning rate.
The following sets of parameters were used to create the models:

1. Learning rate of 2.5e-4, batch size of 256, no augmentations with full precision.

2. Learning rate of 2.5e-4, batch size of 128, kornia augmentations, and automatic mixed precision.
3. Learning rate of 2.5e-4, batch size of 256, dropout rate of 0.3, no augmentations, and automatic mixed precision.
4. Learning rate of 2.5e-4, batch size of 64, dropout rate of 0.3, kornia augmentations, and automatic mixed precision.
5. Learning rate of 5e-4, batch size of 128, no augmentations, and automatic mixed precision.
6. Learning rate of 5e-4, batch size of 128, kornia augmentations, and automatic mixed precision.

The kornia augmentations used were:

- RandomRotation of up to 45 degrees with a probability of 0.3.
- RandomHorizantalFlip with a probability of 0.3.
- RandomVerticalFlip with a probability of 0.3.
- RandomAffine of up to 30 degrees with a probability of 0.3.

Data augmentations that changed the color of the image, or erased parts of it were not used, because we believed that they might make it physically impossible for the model to accurately classify the image, as they change or remove vital information for classification.

YOLOv8

YOLOv8 is the eighth version of YOLO (You Only Look Once), an open-source state-of-the-art model designed by Ultralytics. It is built on a CNN architecture and is tailored for various tasks, including object detection and tracking, instance segmentation, image classification, and pose estimation. The YOLOv8 image classification models were all trained on the ImageNet dataset, which contains over 14 million images across 1000 classes. We mainly used the YOLOv8n-cls model which has 2.7 million parameters, but we also tested the YOLOv8l-cls model which has 37.5 million parameters. Using transfer learning, we fine-tuned the models and created 54 different models, each using a different combination of parameters, optimizers, and augmentations.

Each one of the models was trained with one of the following optimizers: SGD, Adam, AdamW, RMSProp, AdamN, and AdamR. For each optimizer, at least 8 models were trained all training for 10 epochs using a batch size of 32 and automatic mixed precision(AMP). The models were trained using the Ultralytics library, using mainly the default settings, aside from those that were changed as parameters.

The following sets of parameters were used to create the models:

1. Default learning rate and momentum, no augmentations using the YOLOv8n-cls model.
2. Default learning rate and momentum, default augmentations using the YOLOv8n-cls model.
3. Default learning rate and momentum, manual augmentations using the YOLOv8n-cls model.
4. Default learning rate and momentum, no augmentations, dropout of 0.3 using the YOLOv8n-cls model.
5. Default learning rate and momentum, default augmentations, dropout of 0.3 using the YOLOv8n-cls model.

6. Default learning rate and momentum, no augmentations using the YOLOv8l-cls model.
7. Default learning rate and momentum, default augmentations using the YOLOv8l-cls model.
8. Default learning rate and momentum, manual augmentations using the YOLOv8l-cls model.

The default set of augmentations includes:

- hsv_h=0.015, which controls the amount of hue adjustment in HSV (Hue, Saturation, Value) color space, which controls the saturation change in the image.
- hsv_s=0.7, controls the saturation change in the image
- hsv_v=0.4, adjusts the brightness of the image.
- translate=0.1, randomly translates the image horizontally and/or vertically by a fraction of the image size.
- scale=0.5, randomly zooms in or out.
- fliplr=0.5, flip image horizontally.
- erasing=0.4, randomly erases patches of the image.

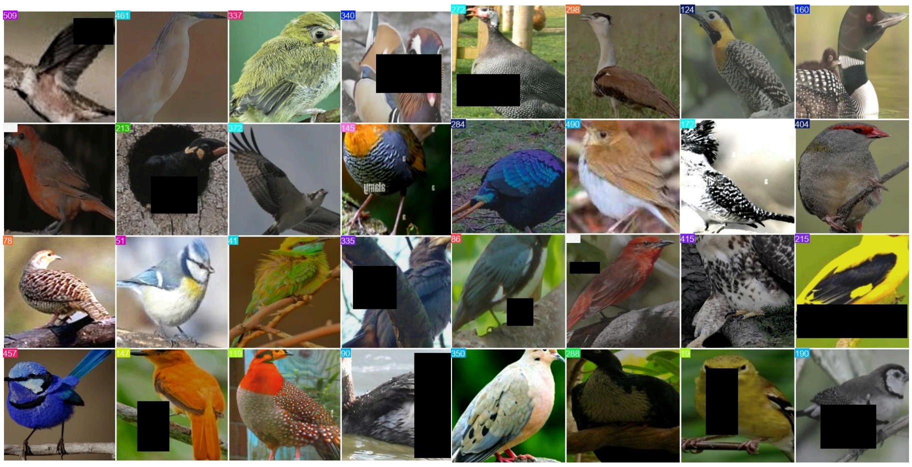

Figure 1: two training batches of the default augmentations

The manual set of augmentations includes:

- hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, translate=0.1, scale=0.5, and fliplr=0.5 from the default augmentations list
- degrees=45, rotates the image by a random degree.
- perspective=0.3, alters the image as if viewed from a different angle.

The main reason for the manual augmentation set was concern that the erasing augmentation was erasing vital patches, making accurate classification impossible.

3. Results

Dataset:

The BIRDS 525 SPECIES - IMAGE CLASSIFICATION dataset is a high-quality dataset comprising 84,635 training images and 2,625 validation and test images each, representing 525 different species of birds.

DINOv2 results:

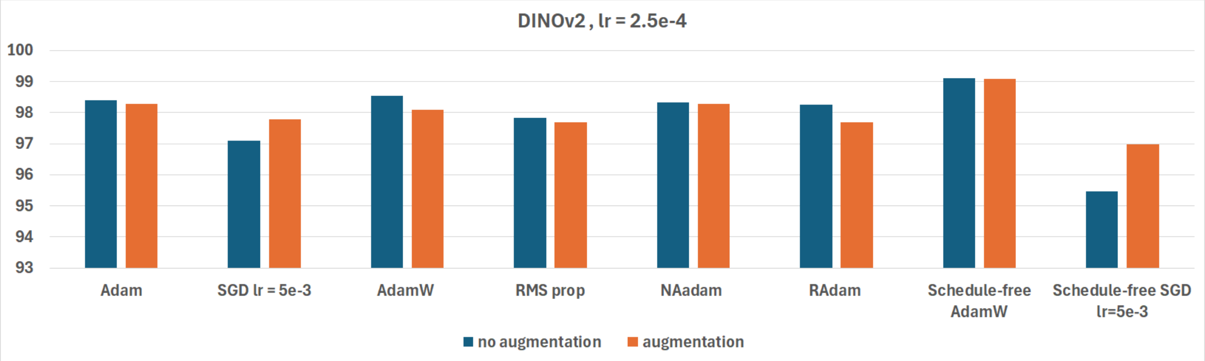

Figure 2: validation set accuracy on DINOv2 first and second parameter sets

Figure 2 reveals several notable results:

1. Schedule-free AdamW outperforms all other optimizers, achieving an accuracy of over 99%.
2. SGD requires a significantly higher learning rate to reach high accuracy. For instance, without augmentation, a learning rate of 1e-4 yielded only 2.32% validation accuracy, while with augmentation, a learning rate of 2.5e-4 increased accuracy to 20.91%. Similarly, schedule-free SGD with learning rates of 2.5e-4 and 1e-3 achieved validation accuracies of 5.33% and 52.76% without augmentations, and 21.22% and 73.49% with augmentations, respectively.
3. The use of AMP, augmentations, and different batch sizes had a minimal impact on accuracy, generally leading to either a slight decrease or a minor increase.

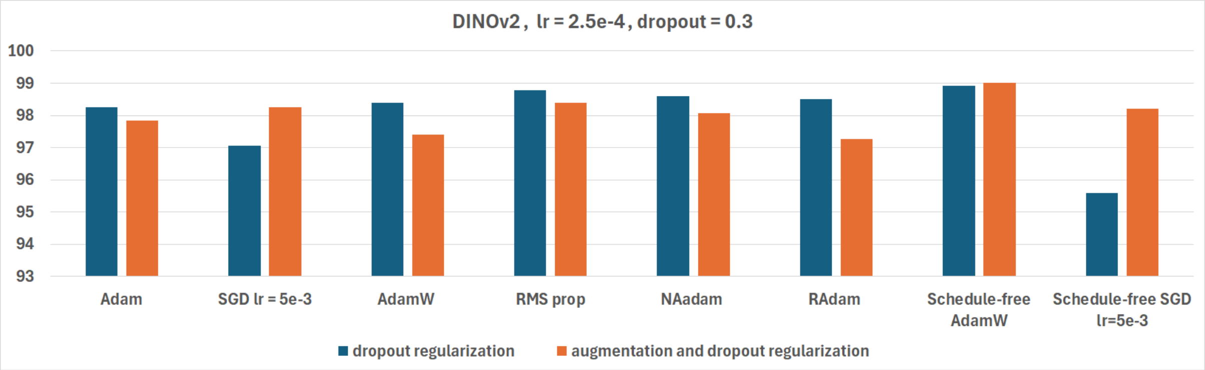

Figure 3: validation set accuracy on DINOv2 third and fourth parameter sets

As shown in Figure 3, dropout did not significantly alter the results and generally had a positive effect on accuracy. The observations from Figure 2 remain relevant, including the challenges with SGD and Schedule-free SGD. Specifically, the SGD optimizer with a learning rate of 1e-4 and no augmentations achieved an accuracy of 2.32%, while the Schedule-free SGD with a learning rate of 2.5e-4 and augmentations reached an accuracy of 21.22%.

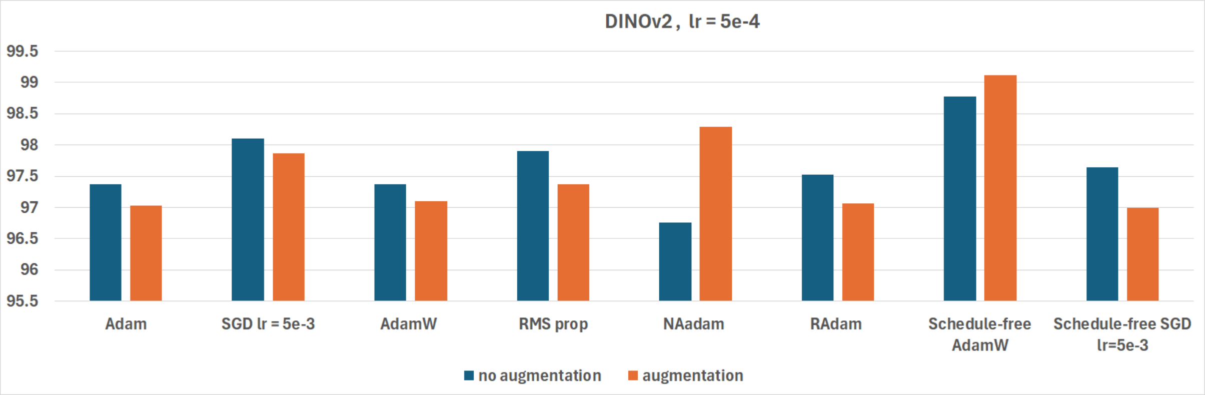

Figure 4: validation set accuracy on DINOv2 fifth and sixth parameter sets

Figure 4 shows that increasing the learning rate from 2.5e-4 to 5e-4 negatively impacted the accuracy of all models, except for those using SGD or Schedule-free SGD. With augmentations, these two optimizers achieved validation accuracies of 79.73% and 51.54%, respectively.
Overall, Schedule-free AdamW outperformed all other optimizers, being the only one to exceed 99% validation accuracy. Additionally, the use of AMP and augmentations did not significantly influence accuracy. Aside from the SGD and Schedule-free SGD models mentioned earlier, all other models achieved a validation accuracy of 95% or higher and showed no significant improvement in performance after 3 epochs.

YOLOv8 results:

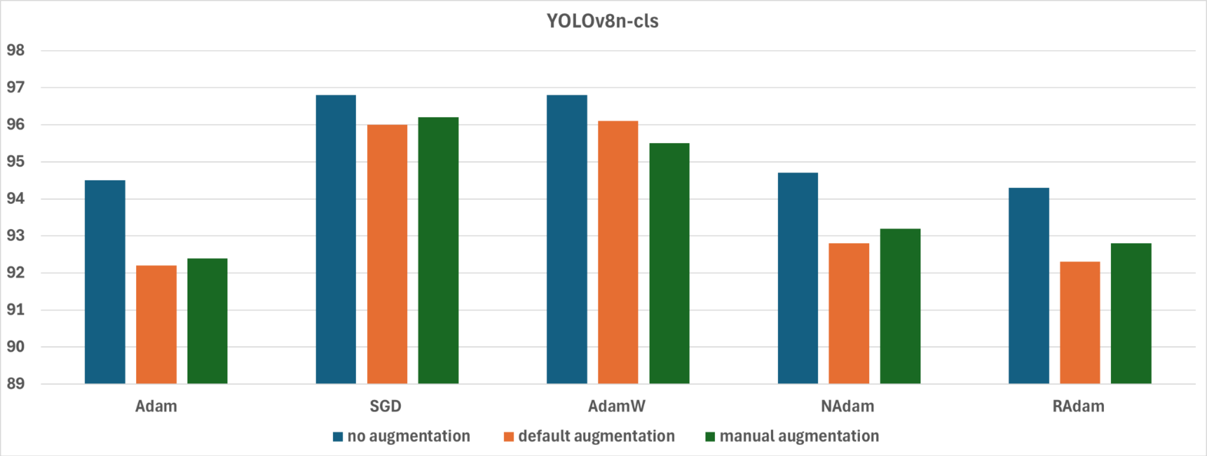

Figure 5: validation set accuracy on YOLOv8n-cls with parameters sets one to three

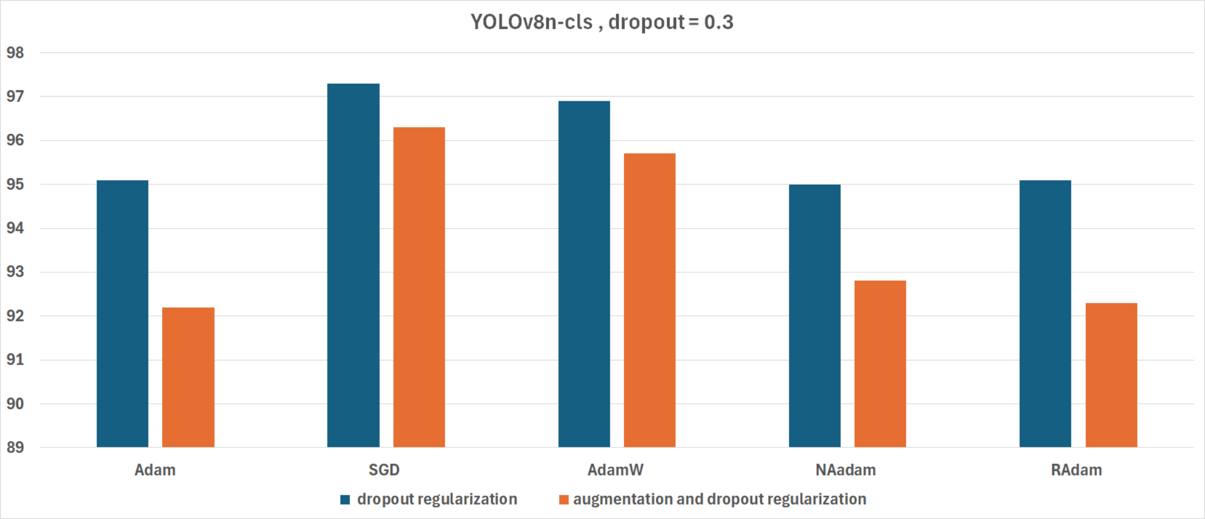

Figure 6: validation set accuracy on YOLOv8n-cls with parameters sets four and five

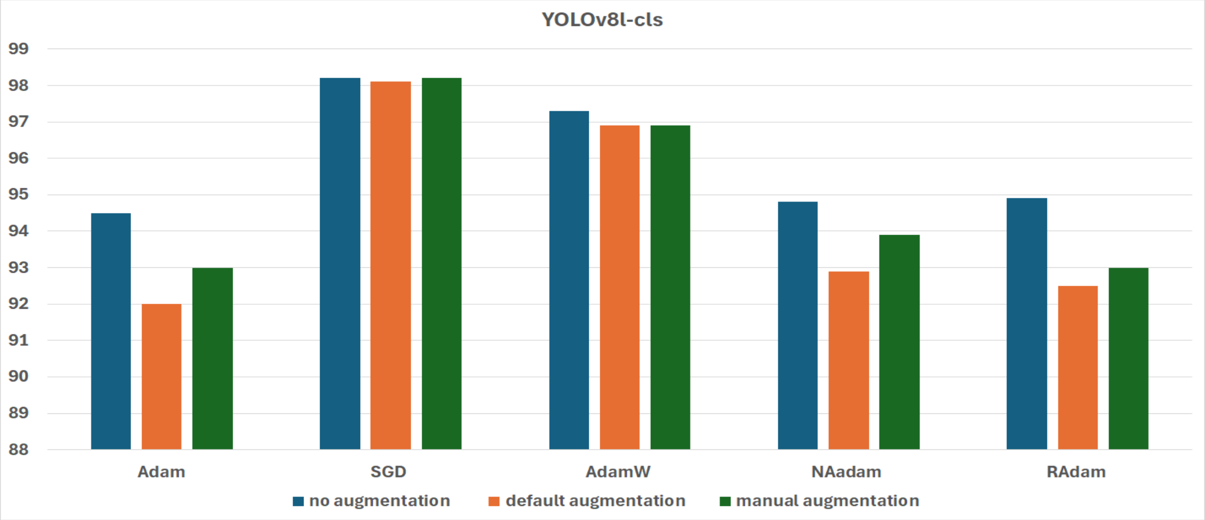

Figure 7: validation set accuracy on YOLOv8l-cls with parameters sets six to eight

From Figures 5, 6, and 7, we observe the following:

1. SGD performed the best when using YOLOv8 on this specific dataset.
2. Models that used augmentations performed worse than those without them.
3. The default augmentations in YOLO resulted in slightly worse performance compared to our custom set of augmentations. We believe this is due to the erasure of important patches in the images, leading to poorer results.
4. Except for the SGD optimizer, YOLOv8l-cls did not outperform YOLOv8n-cls, despite being the more powerful version of the two.

Another noteworthy observation is RMSProp's poor performance with the model, regardless of the learning rate, augmentation, or dropout rate. The highest validation accuracy we achieved with RMSProp was 43.6%, with most models failing to surpass 1% accuracy. This result was obtained without augmentations, using a starting learning rate of 0.008 and an ending learning rate of 0.005. Despite creating 13 additional models, we were unable to achieve better results with RMSProp. We suspect that RMSProp may simply be unsuitable for this task when using the YOLO model, though a potentially incorrect implementation of the optimizer in the YOLO model could also be the reason.

4. Conclusions

Our results indicate that DINOv2 outperformed YOLOv8, particularly when paired with the Schedule-free AdamW optimizer, achieving a final accuracy of over 99%. In contrast, the YOLOv8 model generally underperformed, with the exception of models that utilized the SGD optimizer.

We also found that using AMP on DINOv2 did not result in any significant loss in performance, nor did augmentations that avoided altering the color space or erasing patches from the images. This is likely due to DINOv2's robust pre-training process. In addition, our findings highlighted the negative impact of certain augmentation techniques on the YOLOv8 model, particularly those that modified the color space or erased parts of the image.

As mentioned, the best result achieved by DINOv2 was 99.12%, using the first set of parameters. In comparison, the best result achieved by YOLOv8 was 98.2%, utilizing the sixth and eighth sets of parameters.

Future Works

We suggest the following future works:

1. Implement and test additional Schedule-free optimizers on DINOv2 to further explore their performance.
2. Implement and evaluate additional optimizers for YOLOv8 to assess their effectiveness.
3. Develop more YOLOv8 models using RMSProp to investigate the reasons behind its poor performance.
4. Benchmark the performance of new YOLO models as their classification versions become available.

5. Ethics Statement

stakeholders that will be affected by the project:

1. **Data Scientists/Researchers -** who are developing and testing models for fine-grained image classification.
2. **Conservationists/Bird Enthusiasts** - who are interested in species identification for conservation or ecological research.
3. **Technology Companies** - that can integrate this classification technology into applications such as wildlife monitoring or mobile applications.

The explanation that is given to each stakeholder

1) **Data Scientists/Researchers:** This project explores the effectiveness of self-supervised models like DINOv2 and supervised models like YOLOv8 for bird species classification, offering insights into the performance of various optimizers. These findings can help guide the selection of model architectures and optimizers for future classification tasks.

2) **Conservationists/Bird Enthusiasts:** This technology allows accurate identification of bird species, even those that look similar, by training AI models on a large dataset of bird images. It could assist in ecological research and help track bird populations for conservation efforts.
3) **Technology Companies:** The models developed in this project could be used to improve wildlife tracking and monitoring systems or be integrated into consumer-facing applications like mobile apps for bird watching, offering accurate, real-time species identification.

While this explanation covers a great deal, we feel it overlooks some potentially negative consequences, particularly for people involved in research tracking bird migration and population studies or in fields where a high-quality fine-grained classification model might pose competition. In such fields, people could lose their jobs as a result of these kinds of models.

6. References

[1] BIRDS 525 data set: https://www.kaggle.com/datasets/gpiosenka/100-bird-species/data
[2] DINOv2 Github: https://github.com/facebookresearch/dinov2
[3] YOLO v8: https://docs.ultralytics.com/models/yolov8/
[4] Bird Detection and Species Classification: Using YOLOv5 and Deep Transfer Learning Models: https://thesai.org/Downloads/Volume14No7/Paper_102-Bird_Detection_and_Species_Classification.pdf
[5] BIRDS 525 SPECIES projects: https://www.kaggle.com/datasets/gpiosenka/100-bird-species/code
[6] Aaron Defazio and Xingyu Alice Yang and Harsh Mehta and Konstantin Mishchenko and Ahmed Khaled and Ashok Cutkosky. The Road Less Scheduled. https://arxiv.org/abs/2405.15682