Bird Species Image Classification Project

Nicola Mambelli
Université Paris-Saclay / CentraleSupélec
Paris-Saclay, France
nicola.mambelli@student-cs.fr

Alexandra Perruchot-Triboulet Rodríguez
Université Paris-Saclay / CentraleSupélec
Paris-Saclay, France
alexandra.perruchot-triboulet-rodriguez@student-cs.fr

Code: https://github.com/aleexx02/BDMA-7-Project.git

Abstract

We present a comprehensive approach to fine-grained bird species classification using the Caltech-UCSD Birds-200-2011 dataset subset. We developed and evaluated three distinct model architectures, all based on ConvNeXt-Small as a backbone feature extractor, with varying classifier head complexities. Our approach incorporated transfer learning, data augmentation, stratified k-fold cross-validation, and test-time augmentation (TTA) to maximize classification accuracy. The models achieved validation accuracies ranging from 91.6% *to* 92.32% *across different folds on a 20-class bird species dataset with only 1,082 training images, demonstrating the effectiveness of deep convolutional architectures for fine-grained visual recognition tasks.*

1. Introduction

1.1. Background and Motivation

Fine-grained visual recognition represents one of the most challenging tasks in computer vision, requiring models to distinguish between visually similar categories that differ only in subtle details [10]. Bird species classification illustrates this challenge, as different species often share similar body shapes, colors, and patterns, with distinctions primarily in texture details or beak shapes.

The *Caltech-UCSD Birds-200-2011* dataset [9] is a recognized benchmark for fine-grained classification, containing images of 200 bird species with annotations. This project focuses on a subset of 20 species, providing a manageable challenging classification task suitable for exploring architectural approaches and training strategies.

1.2. Problem Statement

The primary objective of this project is to develop a strong image classification model that maximizes accuracy on the test dataset while adhering to specific constraints. The key challenge lies in achieving high performance with limited training data (1,082 training images and 103 validation images in 20 classes), which requires the effective use of transfer learning [6], data augmentation [8], and model regularization techniques.

Constraints include the prohibition of using external labeled bird species data for training, though unlabeled bird images or other image types are permitted. This constraint highlights the importance of transfer learning from pretrained models and effective data augmentation strategies.

1.3. Approach Overview

Our approach focuses on five key strategies:

- **Transfer Learning**: using *ConvNeXt-Small* [5] pretrained on ImageNet [1] as a feature extractor, taking advantage of features learned from large-scale image data.
- **Architectural Exploration**: developing three distinct classifier architectures with increasing complexity to explore the impact of model capacity on classification performance.
- **Stratified K-Fold Cross-Validation**: ensuring robust performance evaluation [4] and addressing class imbalance through stratified data splitting.
- **Hyperparameter Optimization**: employing grid search to identify optimal learning rates, dropout rates, and MLP configurations.
- **Ensemble and Test-Time Augmentation**: combining predictions from multiple models [2] and augmented test images [7] to maximize accuracy and reliability.

2. Data

2.1. Dataset Description

The dataset comprises a subset of the *Caltech-UCSD Birds-200-2011* collection [9], focusing on 20 bird species. The data is organized into three primary splits:

Split	Images	Classes
Training	1,082	20
Validation	103	20
Test	400	20
Total	**1,585**	**20**

Table 1. Dataset splits.

The dataset contains 20 distinct bird species, with a total of 1,585 images across all splits. This relatively small dataset size makes transfer learning and data augmentation essential strategies for achieving a strong performance.

2.2. Data Exploration and Characteristics

Comprehensive exploratory data analysis revealed various characteristics of the dataset.

2.2.1. Class Distribution

The dataset exhibits minimal class imbalance across the training and validation splits, as illustrated in Figure 1. The training set contains 51 to 58 samples per class (mean: 54.1 $\pm$ 2.1), while the validation set contains 2 to 8 samples per class (mean: 5.2 $\pm$ 2.1). When combined for k-fold cross-validation (1,185 total images), the distribution is relatively balanced, ranging from 56 to 60 samples per species (mean: 59.2 $\pm$ 1.0). We employed stratified k-fold cross-validation to maintain consistent class proportions across all folds.

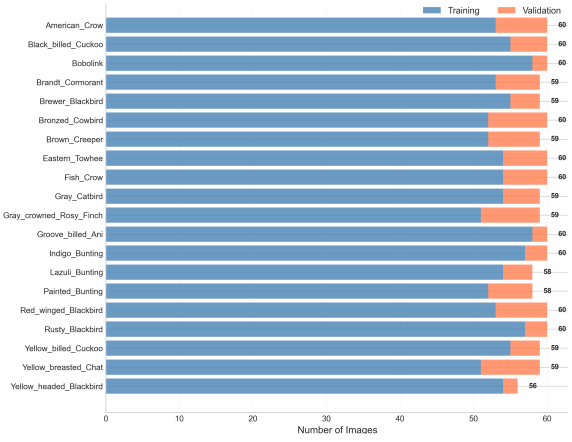


Figure 1. Class distribution across 20 bird species showing training (1,082 images) and validation (103 images) splits. Numbers indicate total samples per class.

Figure 2. Representative training samples from 12 different bird species, shown in original resolution before preprocessing. The visual similarity across species highlights the fine-grained classification challenge.

2.2.2. Image Dimensions

Analysis of the 1,185 training and evaluation images revealed considerable variation in image dimensions. To ensure compatibility with the ConvNeXt-Small architecture [5] and enable efficient batch processing, all images are resized to a consistent 224×224 pixel resolution.

2.2.3. Sample Visualizations

Figure 2 shows representative training samples from 12 species in their original form, before resizing and augmentation, illustrating the diversity and challenging nature of the classification task. Despite being different species, many birds appear visually very similar, sharing comparable body shapes, colors, and poses. The key differences are often subtle details in feather patterns or beak shapes

2.3. Data Preprocessing and Augmentation

We implemented a data preprocessing and augmentation pipeline to improve model performance and generalization. The pipeline differs between training and evaluation phases.

2.3.1. Training Augmentation Pipeline

During training, we apply several transformations to each image:

- **Resize to 224×224 pixels**: ensures all images have the same dimensions required by the model.
- **Random Horizontal Flip**: Each image has a 50% chance of being flipped horizontally, effectively doubling the training data and helping the model recognize birds from different orientations.
- **Random Rotation**: randomly rotates images by ± 5 to simulate different camera angles.
- **Color Adjustments**: Randomly varies brightness, contrast, and color saturation to simulate different lighting conditions and camera settings.

- **Normalization**: Adjusts pixel values using ImageNet statistics to match the pre-trained ConvNeXt model's expectations.

2.3.2. Evaluation and Test Pipeline

For validation and testing, we use a simpler approach without random transformations. Images are resized to 224×224 pixels and normalized using ImageNet statistics. Additionally, during final test predictions, we employ test-time augmentation (TTA) [7] by averaging predictions from both the original image and its horizontal flip, further improving prediction reliability.

2.4. Cross-Validation Strategy

To make the most of our limited data and get reliable performance estimates, we combined the training and validation sets (1,082 + 103 = 1,185 images) and applied 8-fold stratified cross-validation. This means we divide the data into 8 equal parts and train 8 separate models, each using a different part for validation. The stratified approach ensures each part has the same proportion of each bird species. This method gives us more trustworthy performance estimates compared to using just one fixed train-validation split.

3. Model

3.1. Baseline Architecture: ResNet-50

As a starting point for this project, we adopted **ResNet-50**, a well-known and widely used deep learning architecture for image classification. ResNet-50 belongs to the family of *Residual Networks* [3], which had a major impact on computer vision thanks to a simple but powerful idea: residual connections. These skip connections allow the network to bypass certain layers, helping it avoid problems that typically occur when training very deep models, such as vanishing gradients. As a result, ResNet-50 can learn meaningful visual features more reliably and efficiently than many earlier architectures.

ResNet-50 was chosen as a starting backbone for three reasons. First, it is known for being reliable and straightforward to train. Second, high-quality pre-trained weights on ImageNet are readily available, which is essential when working with a small dataset like ours. Finally, because ResNet-50 has been used for many years in both research and industry, its behavior is very well understood, and extensive resources exist to support its use.

Together, these factors made ResNet-50 a natural baseline for evaluating more advanced architectures on our dataset.

3.2. Limitations of ResNet-50 for Fine-Grained Recognition

Although ResNet-50 provided stable performance, its capacity to capture fine-grained visual differences proved insufficient in our context. As mentioned earlier, bird species classification relies on detecting minor variations across classes, such as delicate textural differences in plumage, small changes in beak shape or size, fine color gradients, or localized patterning.

These subtle differences require a model capable of capturing both high-resolution local details and the overall structure of the bird. With our limited amount of data, we observed that ResNet-50 sometimes struggled to distinguish between visually similar classes. This suggested that a more recent architecture, one designed to capture finer details while still understanding the broader image, might be better suited for this task.

3.3. ConvNext as the Feature Extraction Backbone

To address the observed limitations, we transitioned to the **ConvNext** family of architectures [5], specifically **ConvNext-Small**, as the backbone for all subsequent models. ConvNext is a modern reinterpretation of traditional convolutional networks, integrating architectural ideas inspired by Vision Transformers while remaining fully convolutional. This choice was motivated by two considerations:

- **Modernized convolutional design**: ConvNext matches the performance of Transformer-based backbones while retaining the useful inductive biases of CNNs, such as locality and translation equivariance.
- **Suitability for moderate-scale datasets**: CNNs typically generalize well in settings with limited training samples, making ConvNeXt an excellent fit for our data-constrained problem.

Thus, ConvNext-Small served as a strong middle ground between traditional CNNs and pure Transformer architectures.

3.4. Key Architectural Improvements Over ResNet

ConvNeXt brings several design updates that make it more effective for fine-grained image recognition tasks. These improvements can be grouped into changes in the overall structure of the network and changes inside the individual building blocks.

Macro-Architecture Enhancements

These refer to updates in the general layout of the network:

- The network is organized in a more regular and modern hierarchy, somewhat similar to the structure used in Transformer models.
- It introduces additional downsampling steps to gradually reduce the spatial size of the feature maps in a controlled manner.
- The blocks within each stage are generally deeper and wider, allowing the model to learn richer and more expressive features.

Micro-Architecture Modifications

These are smaller improvements inside each convolutional block:

- **Large depthwise convolutions** (7×7) help the model capture a wider area of the image, improving its ability to understand context.
- **Layer Normalization** replaces BatchNorm, offering more stable behavior and aligning with techniques used in newer architectures.
- **Simplified activation patterns** reduce the number of nonlinear operations, making training smoother.
- **MLP-style layers** inside each block help the model mix information across channels more effectively.
- **Cleaner residual block design**: keeps the skip connections but uses a simpler and more efficient internal structure.

These architectural refinements are illustrated in Figure 3.

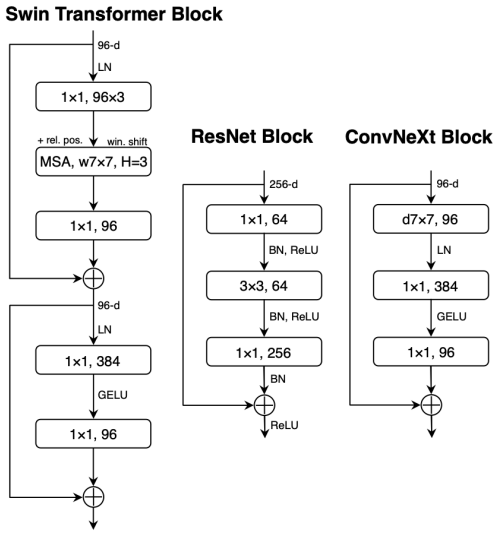


Figure 3. Comparison between the building blocks of Swin Transformer, ResNet, and ConvNeXt. ConvNeXt introduces larger depthwise convolutions, Layer Normalization, and a simplified residual design, bridging ideas from Transformers while remaining fully convolutional.

Together, these design choices allow ConvNeXt to extract fine local details while also capturing broader spatial information, which is particularly important for fine-grained visual recognition tasks such as bird species classification.

4. Experiments and Results

This section presents the experimental methodology and results for developing an effective bird species classification model. We describe our experimental setup, detail three distinct classifier architectures, present the hyperparameter tuning procedure, and analyze the resulting performance.

4.1. Experimental Setup

4.1.1. Implementation Framework

All experiments were implemented in Python using PyTorch, a widely adopted deep learning framework. We leveraged the `torchvision.models` library for pre-trained architectures, allowing us to focus on experimental design and model optimization rather than low-level implementation details.

4.1.2. Initial Baseline: ResNet-50

Our initial approach employed ResNet-50 as the feature extraction backbone. This choice was motivated by the goal of improving model performance through training strategies rather than relying on excessive model complexity or parameter count.

The baseline configuration consisted of:

- Pre-trained ResNet-50 backbone for feature extraction.
- Single fully connected linear layer adapted to 20 classes.
- Standard training techniques:
  - **Adam optimizer**: Adaptive learning rate optimization for stable convergence.
  - **Dropout**: Applied in the classification head to reduce overfitting.
  - **CosineAnnealingLR scheduler**: Gradual learning rate decay for refined optimization.
  - **Data augmentation**: Random crops, flips, and color variations to increase training diversity.

However, validation and Kaggle test accuracies indicated that a more modern architecture would be beneficial for this fine-grained classification task.

4.1.3. Transition to ConvNeXt-Small

Based on baseline results, we adopted ConvNeXt-Small [5] as our primary feature extraction backbone. ConvNeXt-Small offers a modernized convolutional design while maintaining comparable model capacity to ResNet-50, making it well-suited for fine-grained visual recognition. The model was initialized with ImageNet-1K pre-trained weights (approximately 1,000 classes), which serves as the standard pre-training protocol for image classification tasks. We maintained the same training approaches used in the baseline (Adam, Dropout, CosineAnnealingLR, and data augmentation), ensuring controlled comparison between backbones while remaining consistent with the original ConvNeXt training protocol [5].

Decision Against Advanced Augmentation Techniques

The original ConvNeXt paper discusses several advanced augmentation strategies:

- Mixup: Linearly interpolates pairs of training samples.

- CutMix: Combines image patches from different samples.
- Random Erasing: Randomly masks portions of images.

We deliberately chose not to incorporate these techniques due to the fine-grained nature of our task:

- **Mixup and CutMix**: Combining visually similar bird species could create ambiguous training samples, potentially introducing noise rather than beneficial regularization.
- **Random Erasing**: Masking image regions risks removing discriminative details (plumage patterns, beak characteristics) crucial for distinguishing similar species.

While these techniques can improve robustness in other contexts, we determined they were inappropriate for fine-grained bird classification.

Test-Time Augmentation During inference, we employed test-time augmentation (TTA) [7]: for each test image, we computed predictions on both the original and horizontally flipped versions, then averaged the resulting logits. This technique reduces orientation sensitivity and improves prediction stability at minimal computational cost.

4.2. Classifier Architecture Exploration

After establishing the backbone and training strategy, we investigated the impact of classifier head design on performance. We conducted three systematic experiments with different head configurations:

1. Fixed-depth MLP classifier
2. Variable-depth MLP classifier with tunable architecture
3. Three-head ensemble with learnable fusion

4.2.1. Experiment 1: Fixed-Depth MLP Classifier

Motivation This experiment investigates whether a deeper classification head can improve separation between visually similar species by learning more complex decision boundaries than a simple linear classifier.

Architecture We implemented a three-layer MLP with progressive dimensionality reduction:

- **Input**: ConvNeXt features (768 dimensions)
- **Hidden Layer 1**: 512 dimensions
- **Hidden Layer 2**: 256 dimensions
- **Output Layer**: 20 dimensions (class logits)

Each hidden layer includes:

- Linear transformation
- GELU activation function
- Layer Normalization
- Dropout regularization

This configuration enables non-linear feature transformations, potentially improving fine-grained discrimination compared to a purely linear head.

4.2.2. Experiment 2: Variable-Depth MLP Classifier

Motivation. This experiment explores whether optimal classifier architecture can be identified through systematic hyperparameter tuning, allowing the model to adapt its complexity to the task requirements.

Architecture. The classifier structure is dynamically defined through configurable hyperparameters:

- When depth = 1: A regularized linear classifier (Dropout + fully connected layer).
- When depth > 1: Multiple hidden layers with progressive dimensionality reduction.

Each hidden layer consists of:

- Linear transformation with GELU activation.
- Dropout regularization.
- Optional Layer Normalization.

Hidden dimensions are reduced across layers according to a compression ratio, controlling model capacity and encouraging generalization.

Tunable Hyperparameters.

- **MLP depth**: Number of layers in the classification head.
- **Hidden dimension**: Size of the first hidden layer.
- **Width reduction ratio**: Factor governing progressive dimensionality reduction.
- **Layer normalization**: Optional normalization for training stability.

4.2.3. Experiment 3: Three-Head Ensemble Classifier

Motivation. This experiment tests whether parallel classification heads with learnable fusion can capture complementary patterns, improving overall accuracy through ensemble-like behavior.

Architecture. The backbone's classification layer is replaced by three parallel heads that independently process ConvNeXt features:

- Each head is a configurable MLP (depth 0-3 layers).
- Heads can have different widths, introducing structural diversity.
- Each head produces independent class logits.

Head construction follows the variable-depth design with:

- Fully connected layers with GELU activation.
- Dropout regularization.
- Optional Layer Normalization.
- Progressive dimensionality reduction via compression ratios.

Learnable Fusion Mechanism. The three heads' predictions are combined through adaptive weighted fusion:

$$\text{logits}_{\text{final}} = \sum_{i=1}^{3} w_i \cdot \text{logits}_i$$

where fusion weights w_i are learned per-class and normalized via temperature-scaled softmax:

$$w_i = \frac{\exp(\alpha_i/T)}{\sum_{j=1}^{3} \exp(\alpha_j/T)}$$

The temperature parameter T controls fusion behavior:
- Lower temperatures: Favor the dominant head.
- Higher temperatures: Produce smoother, more balanced combinations.

Tunable Hyperparameters.
- **Number of hidden layers** (L): Depth of each head (0-3 layers).
- **Main hidden dimension** (h_{main}): Primary width reference.
- **Alternative hidden dimension** (h_{alt}): Secondary width for diversity.
- **Width assignment pattern**: Distribution of h_{main} and h_{alt} (e.g., `1_2`, `2_1`).
- **Compression ratios** (r_2, r_3): Progressive dimensionality reduction factors.
- **Layer normalization**: Enable/disable LayerNorm within heads.
- **Linear-head normalization**: Whether to apply LayerNorm to linear classifiers.
- **Fusion temperature** (T_w): Controls softmax sharpness for weight normalization.

The parameters h_{main} and h_{alt} define two alternative hidden-layer widths for the classification heads. The assignment pattern (`1_2` or `2_1`) specifies how these widths are distributed across the three heads, introducing architectural diversity among parallel classifiers.

4.3. Hyperparameter Tuning

4.3.1. Tuning Methodology

Model performance depends critically on both training dynamics and architectural design choices. To systematically identify optimal configurations, we employed grid search using Weights & Biases (W&B), which provides structured experiment tracking and visualization tools.

Shared Training Parameters. The following hyperparameters were tuned across all experiments:
- Batch size.
- Learning rate.
- Training duration (split into two phases):
  - **Frozen phase**: Classification head trained with frozen backbone.
  - **Fine-tuning phase**: Full network trained end-to-end.
- Dropout rate in classification head.

Experiment-Specific Parameters. Experiments 2 and 3 included additional architectural hyperparameters specific to their designs (e.g., MLP depth, hidden dimensions, fusion patterns).

Learning Rate Scheduler Selection. In Experiments 2 and 3, we also tuned the learning rate scheduler type, comparing:
- **CosineAnnealingLR**: Smooth cosine decay.
- **ReduceLROnPlateau**: Adaptive reduction based on validation performance.

Selection Criterion. Each hyperparameter configuration was evaluated via stratified k-fold cross-validation. The configuration achieving the highest mean validation accuracy across folds was selected as optimal for that experiment.

4.3.2. Optimal Configurations

Tables 2, 3, and 4 present the best hyperparameter settings identified for each experiment.

Table 2. Optimal hyperparameters for fixed-depth MLP classifier.

Hyperparameter	Value
Learning rate	3.95×10^{-4}
Weight decay	1×10^{-2}
Batch size	64
Dropout	0.25
Frozen epochs	30
Fine-tuning epochs	10
Rotation degrees	30°
Color jitter strength	0.10

Table 3. Optimal hyperparameters for variable-depth MLP classifier.

Hyperparameter	Value
Learning rate	2×10^{-4}
Weight decay	1×10^{-3}
Batch size	32
Dropout	0.30
Frozen epochs	25
Fine-tuning epochs	10
MLP depth	2 layers
MLP hidden size	1024
MLP width ratio	0.50
Rotation degrees	30°
Color jitter strength	0.10
Scheduler type	ReduceLROnPlateau
Plateau factor	0.50
Plateau patience (frozen)	2
Plateau patience (fine-tuning)	1

4.4. Results

As summarized in Tables 2, 3, and 4, three independent experiments were carried out using the optimal hyperparameter configurations selected via grid search and cross-

Table 4. Optimal hyperparameters for three-head ensemble architecture.

Hyperparameter	Value
Learning rate	2×10^{-4}
Weight decay	1×10^{-2}
Batch size	64
Dropout	0.30
Frozen epochs	20
Fine-tuning epochs	10
Rotation degrees	15°
Color jitter strength	0.10
Scheduler type	CosineAnnealingLR
Cosine $\eta_{\min}$	1×10^{-5}
Cosine $T_{\max}$ (frozen)	40
Cosine $T_{\max}$ (fine-tuning)	10
Head hidden main	512
Head hidden alt	256
Head pattern	1_2
Hidden ratio (2nd layer)	0.75
Hidden ratio (3rd layer)	0.75
Number of hidden layers	1
Use LayerNorm	True
Linear no norm	False
Number of folds	8
Pretrained backbone	True

validation. After training the models with these settings, their performance was evaluated on both the validation split and the held-out test set. The resulting accuracies are reported in Table 5.

Table 5. Validation and test accuracy (%). Best values in bold.

Model	Val	Test
ResNet-50 baseline	89.5	73.5
ConvNeXt + Fixed Head	91.7	**82.5**
ConvNeXt + Variable Head	91.5	82.0
ConvNeXt + Three-Head	**92.5**	81.5

As shown in Table 5, the fixed-head model achieved the highest test accuracy (82.5%), outperforming both the variable-depth (82.0%) and three-head ensemble (81.5%) architectures. This result was unexpected, as the more complex three-head model was anticipated to achieve superior performance. However, the three-head ensemble obtained the highest validation accuracy (92.5%) despite lower test performance, revealing a significant validation-test discrepancy.

This discrepancy can be attributed to the limited dataset size. With only 1,185 training images, validation set performance is highly sensitive to data splitting and may not accurately reflect the test distribution. Validation accuracy can be influenced by the specific samples included in each fold, making it an unreliable indicator of generalization performance in small-dataset scenarios.

To better understand the performance differences, we employed Grad-CAM, an interpretability technique that visualizes the image regions most influential for model predictions. Grad-CAM computes gradients of the predicted class score with respect to the final convolutional layer's feature maps, using these gradients to weight and aggregate feature maps into a class activation map showing where the network focuses its attention.

The analysis revealed that while both fixed-head and three-head models generally focus correctly on bird regions, the three-head ensemble occasionally places greater emphasis on background areas. Although the overall accuracy gap is modest, this suggests the multi-head architecture exhibits increased sensitivity to background cues.

This effect is likely amplified by limited training data: the three-head architecture's additional learnable components may increase susceptibility to training-set variability, leading to reliance on spurious correlations that don't generalize to the test set. Both configurations occasionally attend to background regions, potentially exploiting contextual habitat information for classification.

A potential mitigation would be tighter cropping around birds to reduce background influence. However, without precise bounding box annotations, aggressive cropping risks removing discriminative details such as beak shapes or plumage patterns. We therefore retained full images to preserve informative features.

5. Conclusions

5.1. Summary of Findings

This project successfully developed a fine-grained bird species classification system using ConvNeXt-Small, achieving 82.5% test accuracy with only 1,185 training images (approximately 54 per class). Key findings include:

- **Modern architectures matter**: ConvNeXt-Small outperformed ResNet-50 by 9 percentage points (73.5% $\rightarrow$ 82.5%), validating architectural innovation for fine-grained tasks.
- **Simplicity wins**: The fixed-depth three-layer MLP (82.5%) marginally outperformed variable-depth (82.0%) and three-head ensemble (81.5%) architectures, demonstrating that well-regularized simple models often generalize better with limited data.
- **Interpretability provides insights**: Grad-CAM revealed that the three-head ensemble occasionally emphasizes background regions more than the fixed-head model, potentially explaining its lower test performance despite higher validation accuracy (92.5% vs. 81.5%).
- **Transfer learning is essential**: ImageNet pre-trained weights enabled competitive performance despite severe

data limitations, requiring only 20-30 frozen epochs plus 10 fine-tuning epochs.

5.2. Key Design Decisions

Architecture. ConvNeXt-Small's modern design (7×7 depthwise convolutions, Layer Normalization) enabled superior fine-grained discrimination. The simple fixed-depth MLP ($768 \rightarrow 512 \rightarrow 256 \rightarrow 20$) proved most effective, suggesting architectural complexity offers diminishing returns with limited data.

Data augmentation. Random flips, rotations (±15-30°), and color jitter improved generalization. We deliberately avoided Mixup, CutMix, and Random Erasing, which risk creating ambiguous samples or removing discriminative features in fine-grained tasks. Test-time augmentation provided consistent gains.

Interpretability. Grad-CAM analysis revealed both models generally focus correctly on birds, but the three-head ensemble shows occasional background sensitivity. Without bounding box annotations, aggressive cropping risks removing discriminative details, so we retained full images.

5.3. Challenges and Limitations

- **Data scarcity**: Only 54 images per class necessitated aggressive regularization and limited architectural exploration.
- **Fine-grained difficulty**: Visual similarity between species creates a natural performance ceiling; the 18% error rate likely includes inherently ambiguous cases.
- **Background influence**: Models sometimes attend to habitat cues, introducing potential dataset-specific biases.
- **Validation reliability**: Small validation folds (148 images) introduce variance, as evidenced by validation-test discrepancies.

5.4. Lessons Learned

1. **Simplicity often beats complexity**: Simple architectures with strong regularization outperformed sophisticated alternatives in our data-limited regime.
2. **Interpretability is crucial**: Grad-CAM revealed failure modes invisible to accuracy metrics alone.
3. **Task-appropriate choices matter**: Understanding fine-grained requirements guided augmentation decisions and prevented naive application of unsuitable techniques.
4. **Validation requires scrutiny**: Multiple evaluation approaches (accuracy, interpretability, cross-validation) provide more reliable assessment than any single metric.

5.5. Future Directions

Promising improvements include: weakly-supervised localization using Grad-CAM for automatic bird-focused cropping; semi-supervised learning to leverage unlabeled data; attention mechanisms (CBAM, Squeeze-and-Excitation) to emphasize discriminative regions; part-based models for localized features; and extended interpretability analysis (integrated gradients, attention rollout) to identify systematic failure modes.

5.6. Final Remarks

This project demonstrates that modern CNNs with transfer learning, task-appropriate augmentation, and careful regularization achieve strong fine-grained classification despite severe data limitations. The fixed-depth MLP's success over complex alternatives reinforces a fundamental principle: with limited data, simplicity and regularization trump architectural sophistication.

Grad-CAM analysis proved essential, revealing background sensitivity issues and validating that interpretability tools complement numerical metrics. Our systematic approach—combining ConvNeXt's architectural advantages, stratified cross-validation, hyperparameter optimization, and interpretability analysis—represents best practices for data-constrained fine-grained recognition tasks. The 82.5% test accuracy demonstrates respectable performance, though remaining challenges highlight opportunities for weakly-supervised methods, attention mechanisms, and expanded training data.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[2] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[4] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, pages 1137–1145. Montreal, Canada, 1995. 1

[5] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 1, 2, 3, 4

[6] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 1

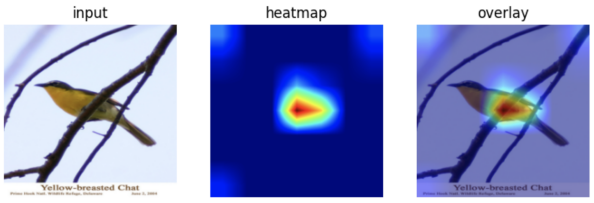
[7] Divya Shanmugam, Davis Blalock, Guha Balakrishnan, and John Guttag. Better aggregation in test-time augmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1214–1223, 2021. 1, 3, 5

[8] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019. 1

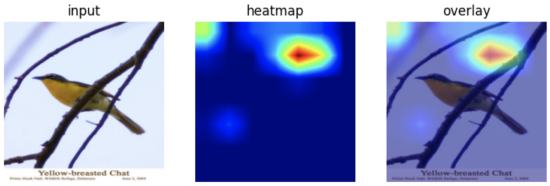
[9] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 1, 2

[10] Xiu-Shen Wei, Chen-Wei Xie, Jianxin Wu, and Chunhua Shen. Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognition*, 76:704–714, 2018. 1

6. Appendix

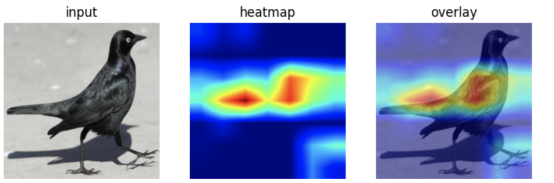


(a) Fixed-head model

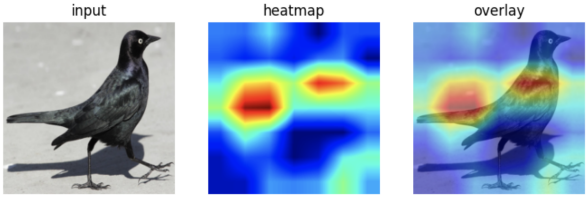


(b) Three-head model

Figure 4. Grad-CAM visualizations for a sample where the three-head model shows increased attention to background regions compared to the fixed-head model.

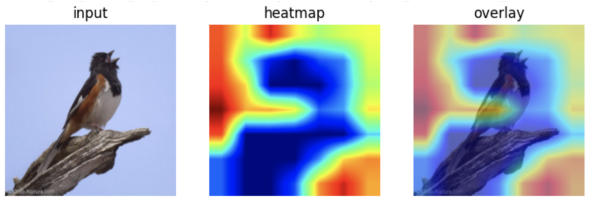


(a) Fixed-head model

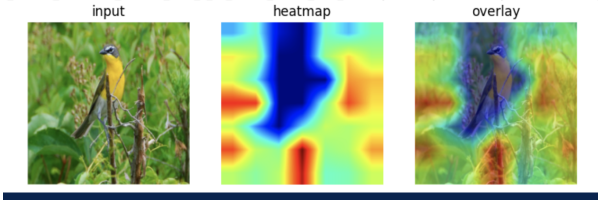


(b) Three-head model

Figure 5. Grad-CAM visualizations for a correctly classified sample, where both the fixed-head and the three-head models focus on the bird region.



(a) Fixed-head model



(b) Three-head model

Figure 6. Grad-CAM qualitative comparison showing that both the fixed-head and the three-head models may occasionally attend to background regions, suggesting reliance on contextual cues in fine-grained classification.

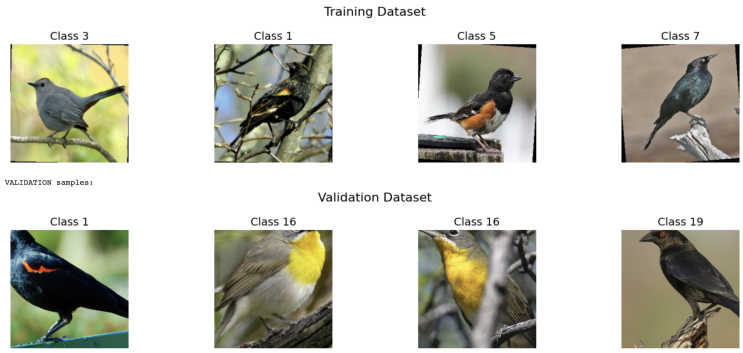


Figure 7. Example of random cropping, showing how discriminative bird features may be partially removed in the absence of precise bounding-box annotations.