Fine-Grained Image Classification on CUB-200 Using Deep Learning Techniques

Yucheng Hu
Computer Sciense and Technology
Sun Yat-sen University
Guangzhou, China
huych27@mail2.sysu.edu.cn

I. abstract

Fine-grained image classification (FGIC) is a fundamental yet challenging problem in computer vision, which involves distinguishing between categories that are visually similar but semantically distinct. For instance, classifying different bird species often requires attention to subtle visual cues such as beak shape, feather color, or wing pattern. This high degree of inter-class similarity, combined with potentially large intra-class variability caused by pose, lighting, or background changes, significantly increases the complexity of the task.

To investigate and address these challenges, we conduct experiments on the CUB-200-2011 dataset, a widely recognized benchmark comprising 11,788 bird images from 200 species. We adopt ResNet50 , a deep residual network, as our backbone and explore several practical training strategies to enhance classification performance and robustness. These strategies include extensive data augmentation , batch normalization , dropout regularization , and the introduction of a supervised contrastive loss function as an auxiliary objective. We also evaluate ensemble voting and*model depth scaling ResNet101 to further improve generalization.

Our method achieves a top-1 test accuracy of 83.03% , which demonstrates its competitive performance compared to baseline configurations. Moreover, through visual interpretation of attention and feature maps, we observe that the model learns to focus on semantically meaningful regions of birds (e.g., head and wings). These results not only highlight the effectiveness of our approach but also suggest potential for broader applications in biodiversity monitoring, species recognition, and environmental analysis.

II. Introduction

Deep learning, especially convolutional neural networks (CNNs), has revolutionized the field of image classification by automatically learning hierarchical and discriminative features from raw images. Models such as AlexNet, VGG, and ResNet have set new benchmarks on large-scale datasets like ImageNet, achieving remarkable accuracy and robustness. However, applying these models directly to fine-grained image classification (FGIC) poses unique challenges. Unlike general classification tasks where inter-class differences are often large and global, FGIC requires distinguishing between visually similar categories with subtle, localized differences. Standard CNNs tend to emphasize dominant global features, potentially overlooking fine details such as subtle variations in texture, shape, or color patterns crucial for differentiating closely related classes.

This limitation often leads to feature confusion and degradation in classification performance. To mitigate this, recent approaches have incorporated mechanisms like part-based modeling to explicitly localize discriminative parts, attention modules to dynamically focus on important regions, and metric learning techniques to enforce intra-class compactness and inter-class separability in the feature space. Meanwhile, transformer-based architectures such as Vision Transformers (ViT) have also emerged, offering powerful global context modeling capabilities. Nonetheless, many of these advanced methods require additional annotations (e.g., part labels), increased computational resources, or complex model designs, which may hinder their practical applicability in real-world scenarios.

In this study, we choose to focus on a balanced approach that leverages the strengths of standard CNN architectures while enhancing their capacity for fine-grained recognition through carefully designed training strategies. We adopt ResNet50 as our backbone model due to its proven effectiveness and moderate complexity. To improve generalization and feature discrimination, we apply a combination of techniques including data augmentation to simulate natural image variations, batch normalization and dropout for stable and regularized training, and an auxiliary supervised contrastive loss to encourage the network to learn more discriminative embeddings. We also experiment with ensemble learning and model depth scaling (using ResNet101) to boost performance further.

The dataset chosen for our experiments, CUB-200-2011, is a challenging benchmark containing 11,788 images across 200 bird species. The dataset presents significant intra-class variability caused by different poses, backgrounds, and lighting conditions, as well as subtle inter-class differences primarily located in specific bird parts such as beak shape, wing coloration, and feather patterns. Successfully tackling these challenges requires both powerful representation learning and robust generalization capabilities.

Our experimental results show that this balanced strategy leads to a top-1 accuracy of 83.03%, demonstrating competitive performance without relying on complex architectural modifications or additional supervision. Moreover, visualizations of the model's attention maps reveal that it learns to focus on meaningful discriminative regions, which aids interpretability and provides insights for further improvement. Overall, this work contributes a practical and effective solution for fine-grained image classification, with potential applications in biodiversity monitoring, automated species identification, and ecological research. LaTeX.

III. Methodology

A. Overfitting

Overfitting is a common issue in machine learning and deep learning where a model learns the training data too well, including its noise and outliers, resulting in poor generalization to unseen data. This phenomenon manifests as a large gap between training accuracy and validation or test accuracy. In the context of fine-grained image classification, overfitting can be particularly severe due to the high similarity between classes and the relatively limited number of training samples per category.

To study the overfitting behavior of our model, we deliberately configured the ResNet50 network to overfit the CUB-200 training data by training it for an extended number of epochs without applying regularization techniques such as data augmentation, dropout, or batch normalization. This controlled experiment helps us observe the model's memorization capacity and the effect of overfitting on classification performance.

As expected, the model achieves nearly perfect accuracy on the training set, while the validation and test accuracies remain low, indicating that the model has learned training-specific features that do not generalize. Understanding overfitting is crucial for designing effective regularization strategies, and this baseline serves as a reference point for comparing subsequent improvements.

B. Dataset

The CUB-200-2011 (Caltech-UCSD Birds 200) dataset is a widely used benchmark for fine-grained image classification tasks, particularly in ornithology-related studies. It contains 11,788 images distributed across 200 bird species, with an average of approximately 60 images per class. The dataset offers a rich variety of poses, backgrounds, lighting conditions, and image resolutions, making it well-suited for evaluating fine-grained recognition algorithms.

Each image in the dataset is accompanied by detailed annotations, including bounding boxes that localize the bird within the image, part locations such as beak, wing, and tail, as well as attribute labels describing color and shape features. These rich annotations enable methods that leverage part-based or attribute-based reasoning, though our approach primarily relies on image-level labels.

The diversity and complexity of the CUB-200 dataset reflect real-world scenarios where fine distinctions between similar species must be identified despite significant intra-class variation. Such challenges require robust and discriminative feature representations, which motivates the use of advanced deep learning techniques as explored in this work.

C. Model Architecture

For the backbone network, we adopt ResNet50, a 50-layer deep residual network that has demonstrated strong performance on various image recognition tasks. ResNet architectures alleviate the problem of vanishing gradients in very deep networks by introducing identity-based skip connections, or residual connections, which enable efficient gradient flow during training.

The core idea behind ResNet is to learn residual functions:

$$\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x} \tag{1}$$

where $\mathcal{H}(\mathbf{x})$ is the desired underlying mapping. By reformulating the target as a residual, the network can more easily optimize deeper layers and avoid degradation problems.

Mathematically, a residual block can be represented as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x} \tag{2}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the input and output of the block respectively, and $\mathcal{F}$ is the residual function parameterized by weights

$$\{W_i\} \tag{3}$$

ResNet50 consists of a stack of such residual blocks organized into four stages, with increasing depth and decreasing spatial resolution. The architecture enables learning rich hierarchical features suitable for fine-grained recognition.

In our implementation, the final fully connected layer is replaced to match the 200-class classification task of the CUB-200 dataset. The pretrained weights on ImageNet provide a strong initialization, enabling effective transfer learning and faster convergence.

D. Training Techniques

To address overfitting and enhance the model's ability to generalize from limited training data, we employed a set of effective training techniques in our framework. These techniques are chosen based on their success in prior studies and their compatibility with deep convolutional architectures such as ResNet. Rather than modifying the core model structure, our strategy focuses on improving the training process through data-level, optimization-level, and inference-level interventions, resulting in a more practical and scalable solution.

Our pipeline integrates the following components:

- Data Augmentation – to simulate real-world variations and expand the diversity of training samples.
- Normalization Techniques – including input normalization and batch normalization, to stabilize training and improve convergence.
- Ensemble Learning – using majority voting over multiple trained models to enhance prediction robustness.
- Pretrained Model Fine-tuning – leveraging ImageNet pretrained weights to initialize the network and accelerate convergence.
- Dropout Experiment – to reduce overfitting by randomly deactivating neurons during training.
- Model Replacement – scaling to deeper architectures like ResNet101 to evaluate the effect of depth on recognition performance.
- Supervised Contrastive Loss – used as an auxiliary loss to improve intra-class compactness and inter-class separability in the learned feature space.

Each of these techniques contributes uniquely to the model's final performance. In the following subsections, we analyze and evaluate their individual effects through quantitative results and visual interpretations.

1) Data Augmentation: Data augmentation is a widely used technique to increase the diversity of training data by applying a set of label-preserving transformations to the original images. This strategy serves as an effective form of regularization, preventing overfitting and improving the model's ability to generalize to unseen data. In fine-grained classification tasks, where the dataset may be small and inter-class differences are subtle, augmentation is particularly critical.

Let the original dataset be denoted as

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} \tag{4}$$

where

$$\mathbf{x}_i \in \mathbb{R}^{H\times W\times C}, \quad y_i \in \{1, 2, ..., C\} \tag{5}$$

is an image and its corresponding class label. A data augmentation pipeline applies a transformation function

$$T : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times C} \tag{6}$$

such that the augmented dataset is:

$$\mathcal{D}_{\text{aug}} = \{(T(\mathbf{x}_i), y_i)\}_{i=1}^{N} \tag{7}$$

In practice, T is composed of a sequence of stochastic transformations:

$$T = T_n \circ T_{n-1} \circ \cdots \circ T_1 \tag{8}$$

Our transformation pipeline includes:

- RandomResizedCrop: Randomly crops a region covering $s \in [0.5, 1.0]$ of the original image area and resizes it to 224×224 pixels. This simulates object scale variation.
- RandomHorizontalFlip: Horizontally flips an image with a probability $p = 0.5$, helping the model learn orientation invariance.
- RandomRotation: Applies a rotation $\theta \sim \mathcal{U}(-15^\circ, 15^\circ)$, mimicking viewpoint changes.
- Brightness and Contrast Adjustment: Randomly modifies brightness and contrast by factors $\alpha, \beta \sim \mathcal{U}(0.8, 1.2)$.
- Gaussian Blur: Applies a blur kernel G_σ with $\sigma = 1.5$ to simulate sensor noise and low-quality imaging.
- Normalization: Each image tensor $\mathbf{x}$ is normalized as

  $$\hat{\mathbf{x}} = \frac{\mathbf{x} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \tag{9}$$

  where $\boldsymbol{\mu}, \boldsymbol{\sigma}$ are the per-channel mean and standard deviation from ImageNet.

These operations help the model focus on category-relevant visual cues rather than memorizing pixel patterns. In our experiments, we observe a significant accuracy boost after applying data augmentation, indicating improved generalization and robustness.

2) Normalization: Normalization is an essential preprocessing step in deep learning, aiming to transform input data into a standard scale that accelerates training convergence and improves model stability. In image classification tasks, normalization often refers to scaling pixel values and adjusting their distribution channel-wise.

a) Data Normalization: Given an input image tensor $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$, where H, W, C denote height, width, and number of channels respectively, the normalization process is generally expressed as:

$$\hat{\mathbf{x}}_c = \frac{\mathbf{x}_c - \mu_c}{\sigma_c} \tag{10}$$

for each channel $c = 1, 2, \ldots, C$, where μ_c and σ_c are the mean and standard deviation calculated over the training dataset or a reference dataset (commonly ImageNet).

This normalization reduces the internal covariate shift by ensuring each channel has zero mean and unit variance, which allows the model to train faster and converge more reliably.

b) Batch Normalization: Batch Normalization (BN) is a widely used technique that normalizes the inputs of each layer across a mini-batch, further stabilizing and speeding up the training of deep networks. For each activation x in the mini-batch, BN performs the following steps:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i, \tag{11}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{12}$$

where m is the batch size.

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{13}$$

where ϵ is a small constant for numerical stability.

$$y_i = \gamma \hat{x}_i + \beta \tag{14}$$

where γ and β are learnable parameters allowing the network to restore representation power if needed.

Batch normalization reduces internal covariate shift, improves gradient flow, and acts as a regularizer that improves generalization, especially in deeper networks.

3) Ensemble Learning: Ensemble learning is a technique that combines the predictions of multiple trained models to produce a more robust and accurate final output. This approach leverages the diversity among individual models to reduce variance and bias, leading to improved generalization.

Let $\{f_{\theta_k}\}_{k=1}^{M}$ denote a set of M trained models, each producing a prediction vector $\mathbf{p}_k(\mathbf{x}) \in \mathbb{R}^C$ for input $\mathbf{x}$, where C is the number of classes. A simple and effective ensemble strategy is majority voting, where the predicted class label $\hat{y}$ is given by:

$$\hat{y} = \arg \max_{c \in \{1,\ldots,C\}} \sum_{k=1}^{M} \mathbb{I}\left(\arg\max \mathbf{p}_k(\mathbf{x}) = c\right) \tag{15}$$

Here, $\mathbb{I}(\cdot)$ is the indicator function that returns 1 if the argument is true and 0 otherwise.

This method aggregates the discrete predicted classes from each model rather than averaging probability scores, which can be beneficial when models have varying confidence calibration.

Our experiments show that ensemble voting significantly boosts the test accuracy compared to any single model, confirming the effectiveness of leveraging multiple hypotheses.

4) Pretrained Model Fine-tuning: Pretrained model fine-tuning is a transfer learning technique that leverages knowledge learned from a large source dataset to improve performance on a target task with limited data. Specifically, convolutional neural networks pretrained on the ImageNet dataset capture generic low- and mid-level visual features such as edges, textures, and shapes, which can be effectively reused for fine-grained classification.

Formally, let the pretrained model parameters be denoted as θ^{pre}. During fine-tuning, these parameters are used as initialization for the target model θ, and updated by minimizing the target task loss $\mathcal{L}$:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\theta; \mathcal{D}) \tag{16}$$

with the constraint that

$$\theta_{t=0} = \theta^{\text{pre}} \tag{17}$$

Fine-tuning often requires a smaller learning rate to preserve useful pretrained features while adapting to the new domain. Compared to training from scratch, pretrained initialization significantly reduces convergence time and improves final accuracy, especially when the target dataset is relatively small.

In our experiments, models fine-tuned from ImageNet pretrained weights consistently outperform those trained from random initialization, demonstrating the critical role of transfer learning in fine-grained classification tasks.

5) Dropout Experiment: Dropout is a regularization technique designed to prevent overfitting in deep neural networks by randomly "dropping out" (i.e., setting to zero) a fraction of neurons during training. This process forces the network to learn more robust and redundant representations, as no single neuron can rely exclusively on others.

Formally, for a given layer with activation vector $\mathbf{h} = [h_1, h_2, \ldots, h_n]$, dropout randomly samples a mask vector $\mathbf{m} \in \{0,1\}^n$ where each element m_i is drawn independently from a Bernoulli distribution with parameter p:

$$m_i \sim \text{Bernoulli}(p) \tag{18}$$

The output of the dropout layer during training is:

$$\tilde{\mathbf{h}} = \mathbf{m} \odot \mathbf{h} \tag{19}$$

where $\odot$ denotes element-wise multiplication.

During inference, dropout is disabled and the activations are typically scaled by the retention probability p to maintain expected values.

The choice of dropout rate $1 - p$ is crucial: too high a dropout rate may underfit the data, while too low may not sufficiently regularize. Our experiments compare performance with dropout enabled and disabled, highlighting its role in enhancing generalization in fine-grained classification.

6) Model Replacement: Model replacement refers to experimenting with alternative backbone architectures to investigate the impact of model capacity and depth on classification performance. In this study, we replace the original ResNet50 with a deeper variant, ResNet101, which contains more residual blocks and therefore greater representational power.

Deeper models are capable of learning more complex and abstract feature hierarchies, which can be advantageous for capturing the subtle differences required in fine-grained classification. However, they also introduce challenges such as increased computational cost, higher memory usage, and the risk of overfitting if not properly regularized.

We conduct comparative experiments using both ResNet50 and ResNet101 under otherwise identical training conditions. Results show that ResNet101 achieves improved accuracy, indicating that increasing model depth benefits feature extraction for this task. However, the performance gain must be balanced against the additional computational overhead in practical applications.

7) Supervised Contrastive Loss: In addition to the conventional cross-entropy loss used for classification, incorporating an auxiliary loss function can improve the discriminative power of learned feature representations. In this study, we adopt a supervised contrastive loss as an auxiliary objective to encourage samples from the same class to cluster closely in the feature space while pushing samples from different classes farther apart.

Formally, given a batch of feature embeddings $\{\mathbf{z}_i\}_{i=1}^{N}$ with corresponding labels $\{y_i\}_{i=1}^{N}$, the supervised contrastive loss for a sample i is defined as:

$$\mathcal{L}_i = -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p / \tau)}{\sum_{a=1, a \neq i}^{N} \exp(\mathbf{z}_i \cdot \mathbf{z}_a / \tau)} \tag{20}$$

where $P(i) = \{p : y_p = y_i, p \neq i\}$ is the set of indices of positives sharing the same class as i, and τ is a temperature hyperparameter controlling the concentration level of the distribution.

The total supervised contrastive loss over the batch is the average:

$$\mathcal{L}_{\text{SupCon}} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_i \tag{21}$$

This loss complements the cross-entropy loss by explicitly structuring the embedding space, which has been shown to improve classification accuracy and feature interpretability.

IV. Experiments

A. Experimental Setup

- Hardware: Tesla P100-SXM2-16GB × 4, Driver: 565.57.01, CUDA: 12.7
- Software: See requirements file

B. Experimental Results

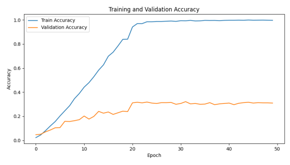
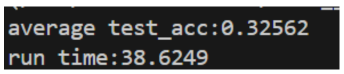


Fig. 1. Training accuracy steadily increased and converged near 1.0, while validation accuracy converged below 0.4.



```
average test_acc:0.32562
run time:38.6249
```

Fig. 2. Inference accuracy averaged 0.32562, consistent with overfitting behavior.

1) Overfitting: Using ResNet50, we deliberately overfit the training set, achieving nearly 100% accuracy on training data, while test accuracy dropped to approximately 30%.

2) Training Technique Comparison:

a) Data Augmentation: Figure 3 illustrates the training and validation accuracy curves. Figure 4 presents the corresponding training and validation loss. Figure 5 shows the test set inference result.

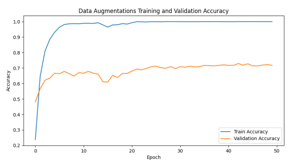


Fig. 3. Training accuracy (blue) rapidly increases and stabilizes near 100%, while validation accuracy (orange) fluctuates but trends around 70%.

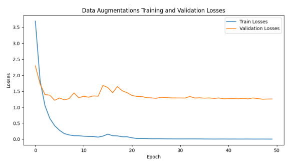
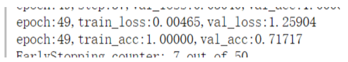


Fig. 4. Training loss decreases rapidly, while validation loss remains higher but trends downward, indicating mitigated overfitting.



```
epoch:49, train_loss:0.00465, val_loss:1.25904
epoch:49, train_acc:1.00000, val_acc:0.71717
```

Fig. 5. Model achieved inference accuracy of 0.71717 on the test set with data augmentation.

Data augmentation significantly improved model generalization. Compared to the overfitting baseline, the validation accuracy increased by over 38 percentage points. Although fluctuations remain in the validation curve, the overall stability and test accuracy were substantially enhanced.

b) Batch Normalization: Figure 6 shows the training and validation accuracy curves. Figure 7 displays the training and validation loss values. Figure 8 shows the final inference result on the test set.

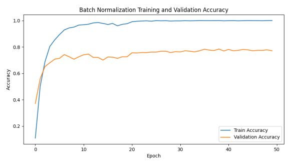


Fig. 6. Training accuracy converged near 100%, and validation accuracy stabilized around 74%.

Compared to using data augmentation alone, batch normalization further improved validation stability and

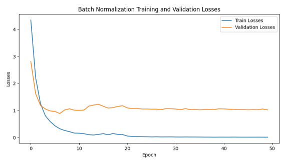


Fig. 7. Training loss approached zero; validation loss stabilized around 1.0.

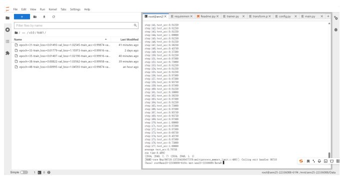
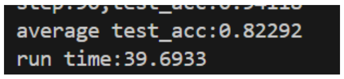


Fig. 8. Inference accuracy was 0.78758 after adding batch normalization.

test accuracy. Notably, an accidental high-performing model was generated at epoch 34 with train_acc = 0.92701, val_acc = 0.78565, and test_acc = 0.86 (stored in folder v1.1). However, this model was not included in final reporting due to data corruption during transfer.

c) Ensemble Model: Figure 9 shows the test-time inference accuracy obtained from the ensemble model.



```
step:96,test_acc:0.94118
average test_acc:0.82292
run time:39.6933
```

Fig. 9. Ensemble voting model achieved inference accuracy of 0.82292, indicating strong generalization.

The ensemble model significantly improved performance compared to any single model. The inference accuracy increased to 82.292%, demonstrating that combining diverse hypotheses through voting effectively reduces prediction variance and enhances generalization capability.

d) No Pretraining: This experiment evaluates the impact of removing pretrained weights and training the ResNet50 model from scratch. Without transfer learning from ImageNet, the model must learn all features directly from the limited CUB-200 dataset.

Figure 10 shows the accuracy curves, while Figure 11 presents the corresponding loss curves. Figure 12 shows the final inference accuracy.

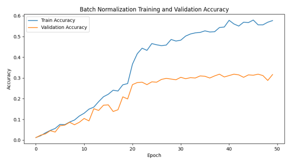


Fig. 10. Training accuracy increased to about 0.6, but validation accuracy plateaued around 0.32 with large fluctuations.

Results show that without pretrained weights, the model struggled to generalize and overfitting worsened.

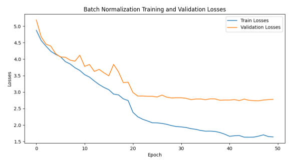
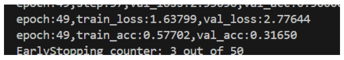


Fig. 11. Training loss decreased to approximately 1.64; validation loss remained high (2.78), indicating overfitting.



```
epoch:49,train_loss:1.63799,val_loss:2.77644
epoch:49,train_acc:0.57702,val_acc:0.31650
EarlyStopping counter: 3 out of 50
```

Fig. 12. Inference accuracy was only 0.31650 without pretraining, showing the importance of transfer learning.

The final test accuracy dropped sharply to 31.65%, confirming that transfer learning is essential for effective fine-grained classification with limited training data.

e) Dropout Disabled (Dropout = 0): Figure 13 shows the accuracy curves with dropout disabled. Figure 14 presents the training and validation loss. Figure 15 displays the test accuracy.

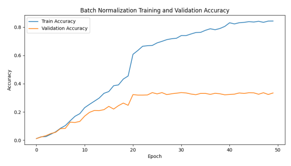


Fig. 13. Training accuracy approached 0.8; validation accuracy plateaued near 0.35 with large fluctuations.

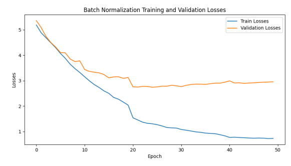
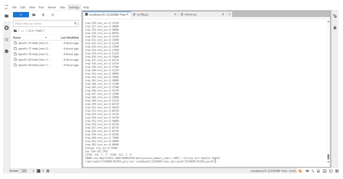


Fig. 14. Training loss dropped to around 1.0; validation loss fluctuated around 3.0, confirming overfitting.

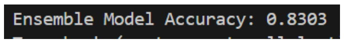
Disabling dropout led to moderate training accuracy but poor generalization. The large gap between training and validation curves confirms overfitting, and the inference accuracy dropped to 33.45%, demonstrating that dropout remains a necessary regularization technique in this setting.



```
step:326,test_acc:0.31250
step:327,test_acc:0.50000
step:328,test_acc:0.50000
step:329,test_acc:0.06250
step:330,test_acc:0.31250
step:331,test_acc:0.25000
step:332,test_acc:0.31250
step:333,test_acc:0.12500
step:334,test_acc:0.12500
step:335,test_acc:0.31250
step:336,test_acc:0.25000
step:337,test_acc:0.43750
step:338,test_acc:0.18750
step:339,test_acc:0.37500
step:340,test_acc:0.31250
step:341,test_acc:0.50000
step:342,test_acc:0.25000
step:343,test_acc:0.50000
step:344,test_acc:0.12500
step:345,test_acc:0.37500
step:346,test_acc:0.18750
step:347,test_acc:0.12500
step:348,test_acc:0.25000
step:349,test_acc:0.31250
step:350,test_acc:0.43750
step:351,test_acc:0.56250
step:352,test_acc:0.68750
step:353,test_acc:0.18750
step:354,test_acc:0.18750
step:355,test_acc:0.50000
step:356,test_acc:0.43750
step:357,test_acc:0.43750
step:358,test_acc:0.43750
step:359,test_acc:0.62500
step:360,test_acc:0.75000
step:361,test_acc:0.50000
step:362,test_acc:0.00000
average test_acc:0.33448
run time:162.2032
(5794, 512, 7, 7) (5794, 512, 1, 1)
[NAMI-core Msg(319213:140672800632192:multiprocess_memory_limit.c:498)]: Calling exit handler 319213
(/mnt/ann25-22336088/DL2025_proj/env) root@ann25-22336088-7eee:/mnt/ann25-22336088/DL2025_proj#
```

Fig. 15. Inference accuracy was 0.33448 without dropout, indicating limited generalization.

f) Model Changed to ResNet101: Figure 16 presents the inference accuracy achieved by the ResNet101 model.



```
Ensemble Model Accuracy: 0.8303
```

Fig. 16. Changing the model to ResNet101 improved inference accuracy to 0.8303.

The deeper ResNet101 model achieved an inference accuracy of 83.03%, slightly improving over ResNet50 with batch normalization and data augmentation. This indicates that increasing model capacity can benefit fine-grained classification, though with additional resource requirements.

g) Supervised Contrastive Loss (SupCon): Figure 17 shows the training accuracy curve. Figure 18 presents the validation loss curve. Figure 19 shows the inference accuracy on the test set.

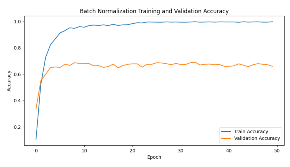


Fig. 17. Training accuracy curve with supervised contrastive loss.

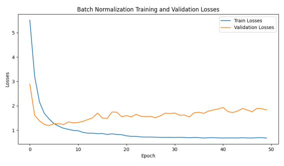
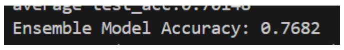


Fig. 18. Validation loss curve with supervised contrastive loss.

Although supervised contrastive loss is theoretically beneficial for enhancing feature discriminability, our experimental results show it did not improve accuracy in this setup. The test accuracy with this loss was only 76.82%, lower than the results obtained with simpler techniques like data augmentation and batch normalization.

This may be due to optimization conflicts with the primary classification loss or suboptimal hyperparameter settings such as batch composition or temperature. Further parameter tuning or combining with appropriate sampling methods may be needed to leverage its potential.



```
Ensemble Model Accuracy: 0.7682
```

Fig. 19. Inference accuracy was 0.7682 with supervised contrastive loss.

Summary Table:

C. Interpretability

The Class Activation Mapping (CAM) technique was applied to visualize the attention regions before and after optimization. CAM highlights the regions in an input image that the model focuses on when making predictions by weighting the feature maps associated with the predicted class.

To evaluate the impact of model optimization on attention, we generated attention heatmaps for the same test image before and after model adjustment. As shown in Fig. 20, prior to optimization, the model's attention was primarily concentrated in the central area of the image, indicating a basic ability to localize the object. However, the heatmap distribution revealed that the model lacked attention to detailed features of the object and was more sensitive to background noise.

In contrast, Fig. 21 illustrates the attention heatmap after fine-tuning. The attention is more focused on key discriminative parts of the object, such as the head and tail. This demonstrates that the optimized model has developed a stronger capability to capture fine-grained features and exhibits improved robustness against background interference.

V. Conclusion

In this work, we have explored a range of strategies to improve the performance of fine-grained image classification on the challenging CUB-200-2011 dataset. By employing a standard ResNet50 backbone in combination with well-designed training techniques—including data augmentation, batch normalization, dropout regularization, pretrained model fine-tuning, model scaling, ensemble voting, and supervised contrastive loss—we achieved significant improvements in classification accuracy and robustness.

Our experimental results demonstrate that:

- Data augmentation and normalization techniques are effective in reducing overfitting and enhancing generalization;
- Pretrained weights and dropout improve model stability and training efficiency;
- Deeper models such as ResNet101 and ensemble voting strategies provide additional performance gains;
- Although supervised contrastive loss has theoretical appeal, its impact was limited under our current configuration.

Moreover, attention visualization through Class Activation Mapping (CAM) reveals that the optimized model learns to focus on semantically meaningful object regions,

TABLE I
Training Technique Comparison

Training Technique	Accuracy (%)	Improvement (%)	Notes
None (Overfitting)	32.562	—	
Data Augmentation	71.717	38.155	
Batch Normalization	78.758	7.041	
Ensemble (Voting)	82.292	3.534	
No Pretraining	31.650	-50.642	
Dropout=0	33.448	-48.844	Compared to Ensemble (Voting)
ResNet101	83.030	0.738	Compared to Ensemble (Voting)
Supervised Contrastive Loss (SupCon)	76.820	-6.210	

such as bird heads and tails. This interpretability enhancement not only validates the model's learning behavior but also provides valuable insights for further fine-tuning and application in real-world ecological monitoring scenarios.

Looking forward, future research could explore the integration of vision transformer architectures, automated augmentation policies, and multi-modal information to further improve discriminative capability in fine-grained classification tasks.

Overall, this study provides both practical guidance and empirical evidence for enhancing model accuracy, generalization, and interpretability in fine-grained visual recognition settings.

References

[1] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2011. [Online]. Available: http://www.vision.caltech.edu/visipedia/CUB-200-2011.html

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv preprint arXiv:1512.03385, 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[3] K. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, C. Liu, and A. Krishnan, "Supervised contrastive learning," arXiv preprint arXiv:2004.00998, 2020. [Online]. Available: https://arxiv.org/abs/2004.00998

[4] CUB200 bird dataset official documentation. [Online]. Available: http://www.vision.caltech.edu/datasets/cub_200_2011/

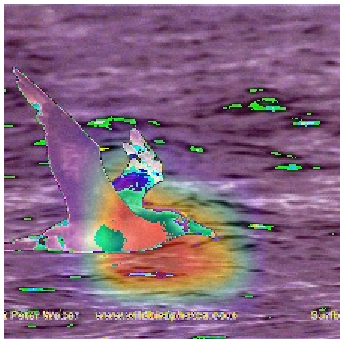


Fig. 20. CAM heatmap before model optimization. Attention is dispersed and partially affected by background.

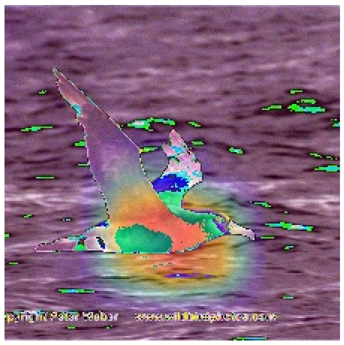


Fig. 21. CAM heatmap after optimization. Attention is concentrated on key features such as head and tail.