Image Classification using ResNeSt50d and SpinalNet

● Introduction

The primary task addressed in this project is image classification, specifically classifying images into 100 distinct categories. The core idea of the implemented method is to leverage a powerful convolutional backbone (ResNeSt50d) integrated with an advanced classification head (SpinalNet). The SpinalNet architecture helps enhance feature representation by using multiple parallel neural network layers, thereby improving the overall robustness and generalization capability of the model.

Github: https://github.com/reson320/VRDL_Homework/tree/main/Homework1

● Method

Data Preprocessing

The dataset is preprocessed using different image augmentations and normalization strategies:

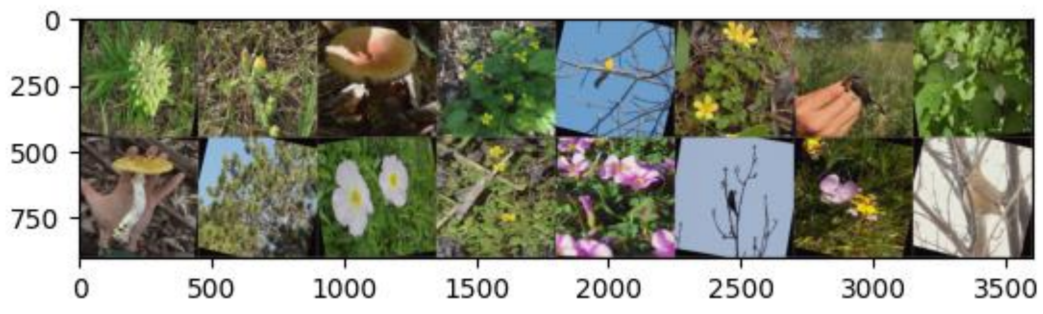


Figure1 - train image when batch size =16

- **Training:** Images are resized to 460×460, randomly cropped to 448×448, randomly rotated (±15°), horizontally flipped, converted to tensors, mixup, and normalized with mean [0.507, 0.487, 0.441] and standard deviation [0.267, 0.256, 0.276].
- **Validation and Testing:** Images are resized to 448×448, directly converted to tensors, and normalized similarly to the training data.

Model Architecture

- **Backbone:** ResNeSt50d from timm library, pretrained on ImageNet.

- **Classification Head:** SpinalNet, designed to extract and integrate complex features by splitting features into halves and concatenating outputs from multiple fully connected layers.
- **Optimizer and Scheduler:** Stochastic Gradient Descent (SGD) optimizer with momentum (0.9) and weight decay (1e-4), combined with a Cosine Annealing learning rate scheduler.
- **Loss Function:** Soft Target Cross-Entropy with label smoothing (0.1) to improve generalization.
- **Regularization Techniques:** Mixup data augmentation with decreasing alpha value across training stages to gradually fine-tune learning dynamics.

Hyperparameters

- Batch Size: 16
- Initial Learning Rates: First stage 0.01, second stage 0.001
- Number of Epochs: 40 epochs (stage 1), 20 epochs (stage 2)
- Early Stopping Patience: 7 epochs
- Mixup alpha: 1.0 (no reduce with training stage)

Results

The model performance was evaluated based on validation accuracy, confusion matrix, and top-1/top-5 accuracy:

- **Training Curves:** Displayed a clear decrease in loss and improvement in accuracy, demonstrating effective learning. In stage 1, which serves as the main training phase, both accuracy and loss show significant improvements. Due to the use of mixup, the training accuracy is limited to around 0.5. Meanwhile, the validation accuracy plateaus near the 0.9 threshold during this stage. In stage 2, the goal was to further refine the model, so the learning rate was reduced. After lowering the learning rate, the validation accuracy slightly improved to around 0.92 in the early epochs but then stagnated.

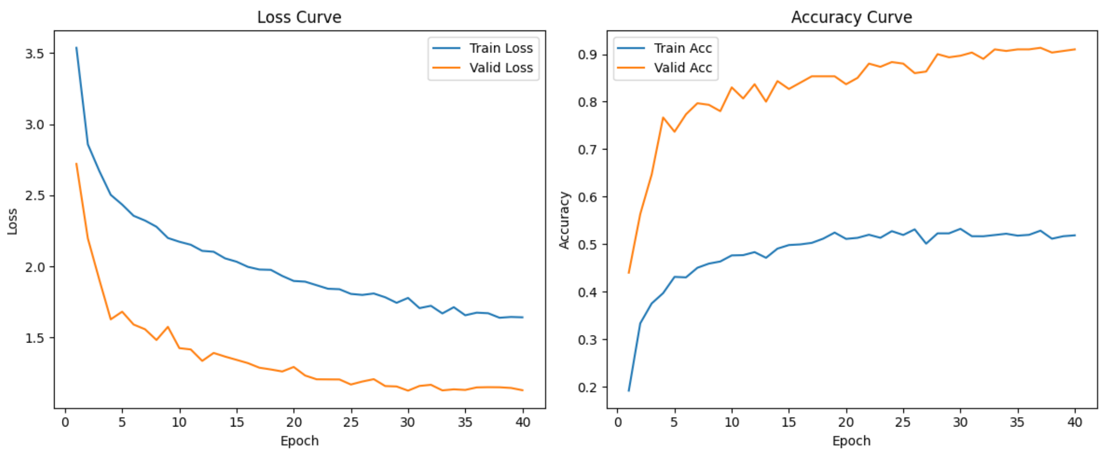


Figure2-training curve at stage 1

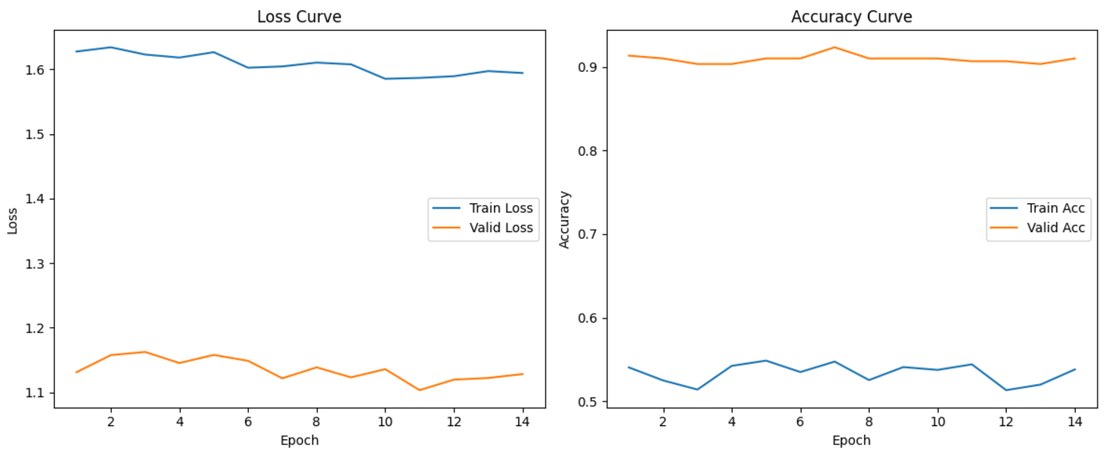
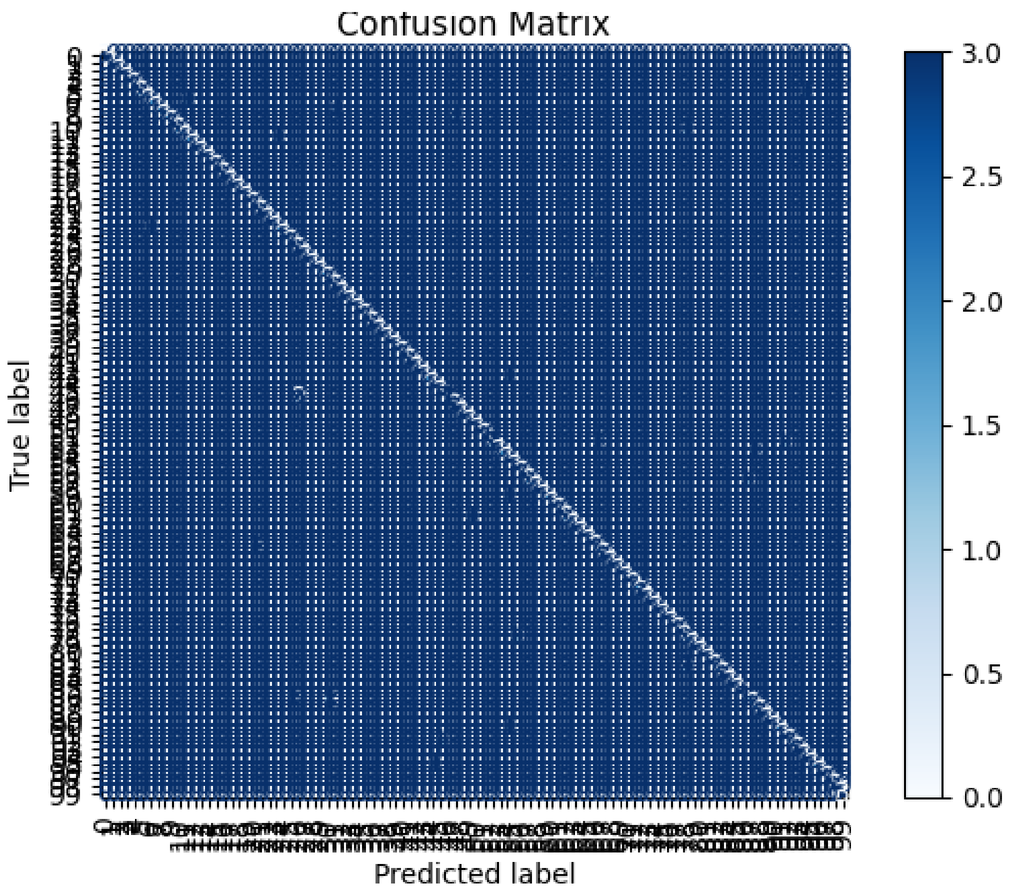


Figure3-training curve at stage 2

- **Confusion Matrix:** Though, this confusion matrix is not that clear. However we can still see that there are some common mis-predict spot in the matrix.



```
Saved confusion matrix to figures/confusion_matrix.png
Top-1 Accuracy: 0.9100
Top-5 Accuracy: 0.9767
Model saved as best_model_second.pth
```

Figure4-training curve at stage 2

- **Top-1 Accuracy:** 0.91
- **Top-5 Accuracy:** 0.9767

Additional Experiments to Explore Better Performance

Hypothesis

The original model used in this project was based on the standard ResNet-50 architecture. While effective, it may not be sufficient to capture complex patterns present in a 100-class image classification task. I hypothesize that switching to a more expressive and deeper backbone, such as ResNeSt-50d, and integrating it with a SpinalNet classifier head can improve the model's ability to learn rich hierarchical representations. The SpinalNet design facilitates gradual feature propagation and introduces redundancy-resilient layers, which may help improve classification accuracy and generalization performance.

Methodology

To evaluate this hypothesis, I implemented a custom model architecture that combines the ResNeSt-50d backbone with a SpinalNet head. Specifically:

- **Backbone:** I used resnest50d from the TIMM library as a feature extractor. The final classification layer of the backbone was removed by setting num_classes=0, and global average pooling was used to produce a fixed-length feature vector.
- **SpinalNet Head:** The feature vector was split in half and passed through four sequential fully connected layers. Each layer received the output from the previous one concatenated with a portion of the original feature vector, allowing for hierarchical feature refinement.
- **Final Layer:** A fully connected output layer produced logits for the 100 target classes.
- **Training Setup:** The model was trained using the same training pipeline, optimizer, and learning rate schedule as the baseline, ensuring a fair comparison. However, I also changed image input size to 448x448, this might also change the result, due to more details in the input images.

Results and Implications

	Top-1	**Public score**	**Private score**
Resnet50	**87.3%**	**0.890**	**0.880**
Resnest50d+spinalnet	**91.0%**	**0.941**	**0.929**

References

- Timm library by Ross Wightman:
  https://github.com/rwightman/pytorch-image-models
- Paperswithcode- Image Classification on ImageNet:
  https://paperswithcode.com/sota/image-classification-on-imagenet
- Kaggle-Spinal Transfer Learning on Bird270:

  https://www.kaggle.com/code/dipuk0506/spinal-transfer-learning-on-bird270?scriptVersionId=63998762
- Huggingface- Model card for s resnest50d.in1k:
  https://huggingface.co/timm/resnest50d.in1k
- SpinalNet: Deep Neural Network with Gradual Input:

  https://github.com/dipuk0506/SpinalNet