Insect Pest Specifies Identification

Team Halo

Liran Fang (z5400642)
Zhikai Lai(z5412820)
Zihang Cheng (z5502944)
Yitong Li (z5504073)
Jinghan Wang (z5286124)

UNSW SYDNEY

Motivation

- **Pest Classification:**
  - Identifying and categorizing various pest species accurately.
- **Crucial for Pest Control:**
  - Effective pest management and agricultural productivity depend on accurate pest identification.
- **Challenges with IP102 Dataset:**
  - Significant data imbalance with underrepresented categories.
  - Potential for biased models that underperform on minority classes.
- **Solution:**
  - Implement targeted data augmentation.
  - Enhance model performance and robustness.

Problem Statement

-
- Insect Pest Classification :
- Insect pest classification is critical for agriculture, pest control, and ecological research. Quick and accurate identification is essential for effective pest management, early detection of invasive species, and maintaining crop quality. Manual classification is often slow, error-prone, and difficult, especially in wild environments.
-
- This project seeks to automate insect identification using advanced algorithms, aiming for high accuracy and robustness in various conditions, facilitating real-time monitoring and efficient pest control.
-

Literature Review

- [1]. X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng and J. Yang, "IP102: A Large-Scale Benchmark Dataset for Insect Pest Recognition," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 8779-8788, doi: 10.1109/CVPR.2019.00899. https://ieeexplore.ieee.org/document/8954351 https://openaccess.thecvf.com/content_CVPR_2019/papers/Wu_IP102_A_Large_Scale_Benchmark_Dataset_for_Insect_Pest_Recognition_CVPR_2019_paper.pdf
- [2]. A. Setiawan, N Yudistira, and R.C. Wihandika, "Large scale pest classification using efficient Convolutional Neural Network with augmentation and regularizers", Computers and Electronics in Agriculture, Vol. 200, Sept 2022. https://www.sciencedirect.com/science/article/pii/S0168169922005191

Literature Review

- [3]. W. Linfeng, L. Yong, L. Jiayao, W. Yunsheng, and X. Shipu, "Based on the multi-scale information sharing network of fine-grained attention for agricultural pest detection", PLOS ONE 18(10):e0286732. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0286732
- [4]. An J, Du Y, Hong P, Zhang L, Weng X., "Insect recognition based on complementary features from multiple views", Scientific Reports. 2023 Feb;13(1):2966. DOI: 10.1038/s41598-023-29600-1. https://europepmc.org/article/pmc/pmc9940688

By reviewing these articles and understanding their approaches and methods, we further refined our own solution with new data cleaning techniques, which effectively increased the accuracy.

Dataset(s)

Table 1. Training/validation/testing (denoted as Train/Val/Test) set split and imbalance ratio (IR) of the IP102 dataset on different class levels. The 'Class' indicates the *sub-class* number of the corresponding *super-class*. The 'FC' and 'EC' denote the field and economic crops, respectively.

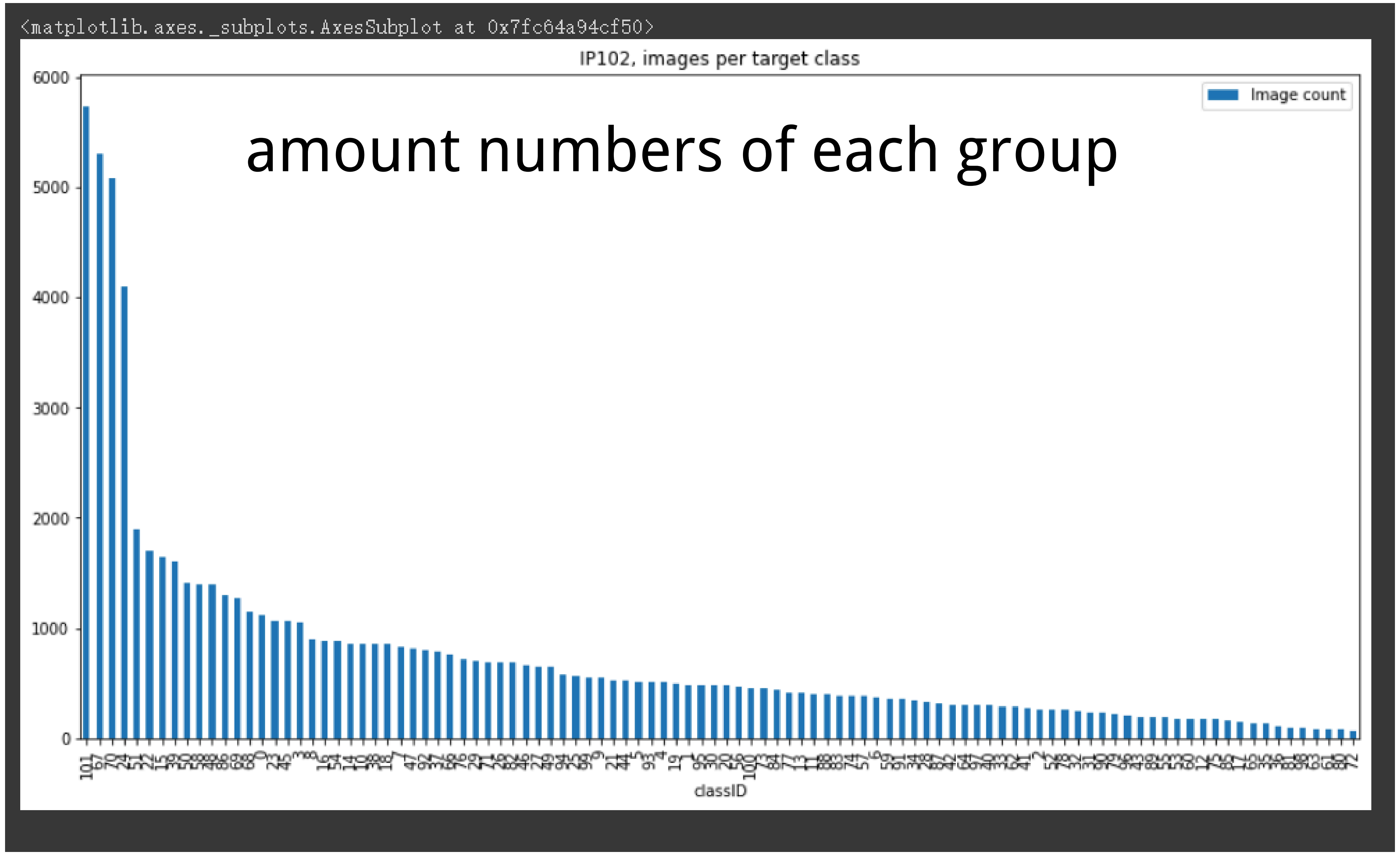
	Super-Class	Class	Train	Val	Test	IR
FC	Rice	14	5,043	843	2,531	6.4
	Corn	13	8,404	1,399	4,212	27.9
	Wheat	9	2,048	340	1,030	5.2
	Beet	8	2,649	441	1,330	15.4
	Alfalfa	13	6,230	1,037	3,123	10.7
EC	Vitis	16	10,525	1,752	5,274	74.8
	Citrus	19	4,356	725	2,192	17.6
	Mango	10	5,840	971	2,927	61.7
IP102	FC	57	24,602	4,098	12,341	39.4
	EC	45	20,721	3,448	10,393	80.8
	IP102	102	45,095	7,508	22,619	80.8

- The IP102 dataset has a hierarchical classification system. They exhibit the structure shown in Figure. Each pest is assigned a high-level category (hereinafter referred to as a superclass) based on the crop it is affected by. In other words, each pest is a subordinate class of some superclass (represented as a subclass below).

https://openaccess.thecvf.com/content_CVPR_2019/papers/Wu_IP102_A_Large-Scale_Benchmark_Dataset_for_Insect_Pest_Recognition_CVPR_2019_paper.pdf

Dataset(s)

	classID	Image count
0	101	5740
1	67	5310
2	70	5081
3	24	4094
4	51	1897
...	...	...
97	98	93
98	63	84
99	61	79
100	80	74
101	72	71



The IP102 dataset contains 75,222 images and 102 pest classes.

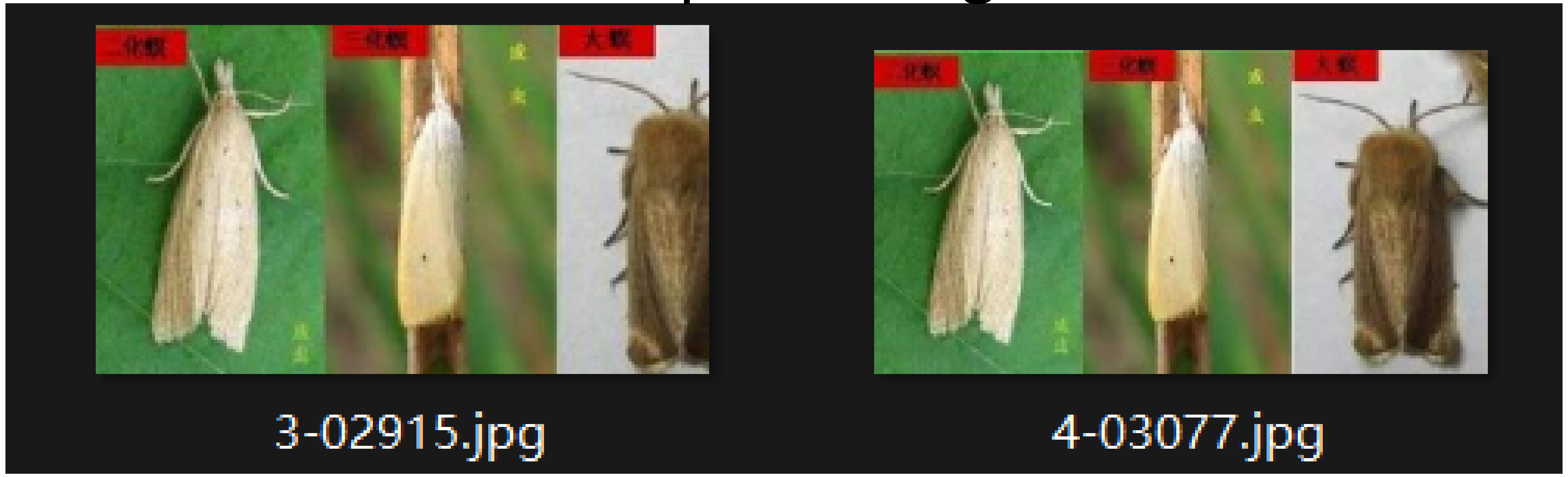
On the task of insect classification:

Train: Test: Validation=6:3:1 (45095/22619/7508)

https://openaccess.thecvf.com/content_CVPR_2019/papers/Wu_IP102_A_Large-Scale_Benchmark_Dataset_for_Insect_Pest_Recognition_CVPR_2019_paper.pdf

Data Analysis

Before processing the data, we discovered some problems with IP102



Same images in different category

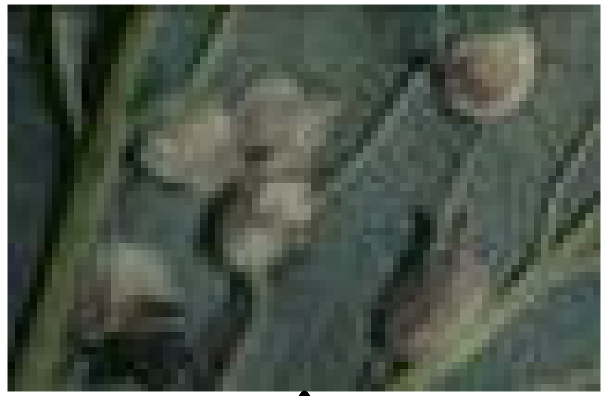
Some of the sources of the images, which we looked up, found that they were from news reports that simply showed experts working in the fields

Data Analysis

IP074000409.jpg

IP075000031.jpg

IP074* and IP075* are actually images of the same class Panonchus citri McGregor (Citrus red spider), but IP074* is listed as Papilio xuthus (citrus papilio). The IP074* label is incorrect.



Ultra-low pixel pic -->1kb

In the aphid classification, after we grabbed the features, we found that this was more biased towards the classification of ants

Data Analysis

Due to the chaotic nature of the data, there is a risk of overfitting and confusion in feature values, leading to a decrease in training accuracy. Therefore, we need to perform data cleaning.

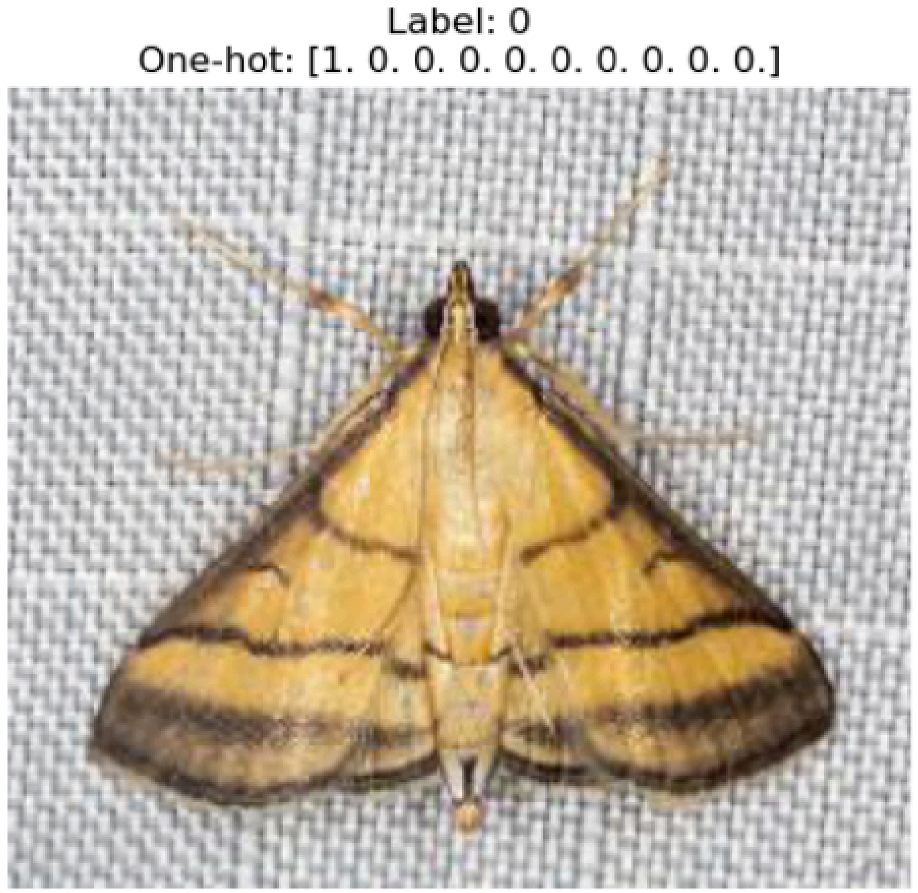
Data Analysis

Considering the advancements in imaging technology and the use of Attention Pyramid Networks (APN) to extract local features.

We pre-filter and organize overly abstract, low-resolution, and meaningless results. We re-cleaned and organized the images based on two rules:
if images are hashed repeatedly within the same category, we combine them into one; if they span multiple categories, we remove them all.
These processed images are then placed into the 'dropped' folder according to their categories for manual error checking

Data Analysis

```
In [14]: # show pic
plt.imshow(image)
plt.title(f'Label: {label}\nOne-hot: {one_hot_label}')
plt.axis('off')
plt.show()

# print one-hot
print('One-hot label:', one_hot_label)
```

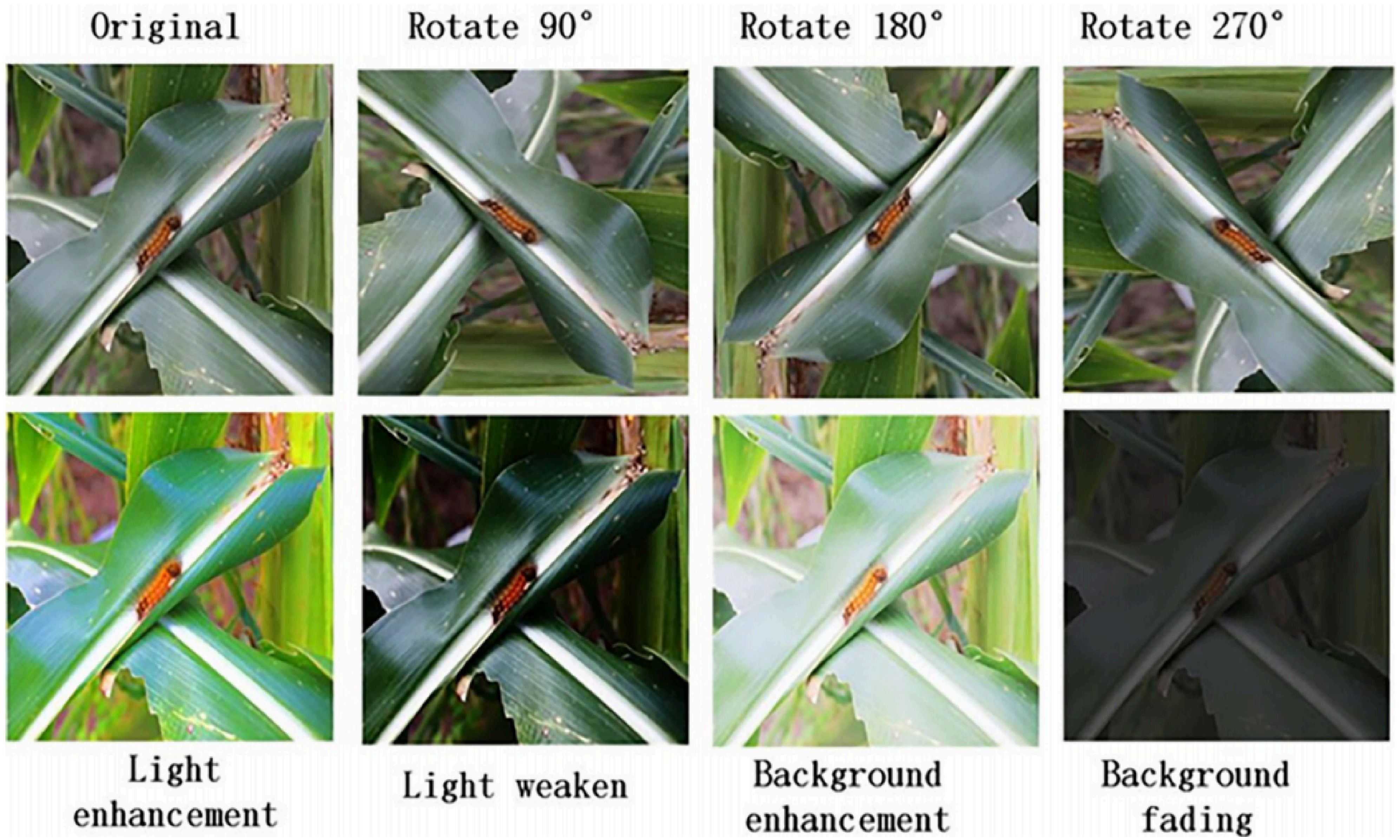



```
One-hot label: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

Data enhancement

```
from PIL import Image, ImageEnhance
import matplotlib.pyplot as plt

#
image_path = './images/00000.jpg'

# load
image = Image.open(image_path)

# sharpen
enhancer = ImageEnhance.Sharpness(image)
sharpened_image = enhancer.enhance(10.0)  # 2.0

#
fig, ax = plt.subplots(1, 2, figsize=(12, 6))

ax[0].imshow(image)
ax[0].set_title('image')
ax[0].axis('off')

ax[1].imshow(sharpened_image)
ax[1].set_title('Sharpened Image')
ax[1].axis('off')

plt.show()
```

Data Analysis



Additionally, since the overall distribution of the IP102 dataset is uneven, with some categories having particularly few samples, we provide extra augmentation for these categories to compensate for the imbalance.

Data enhancement

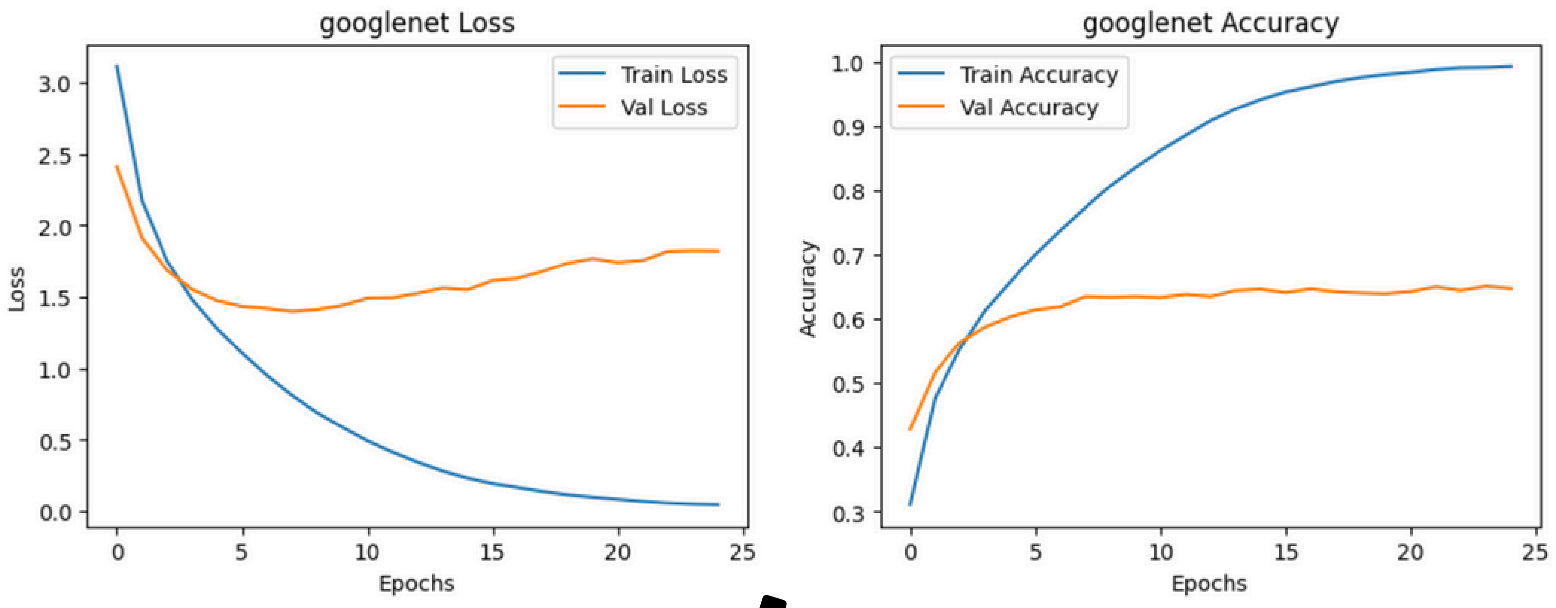
Methods--Different Pre-trained Model

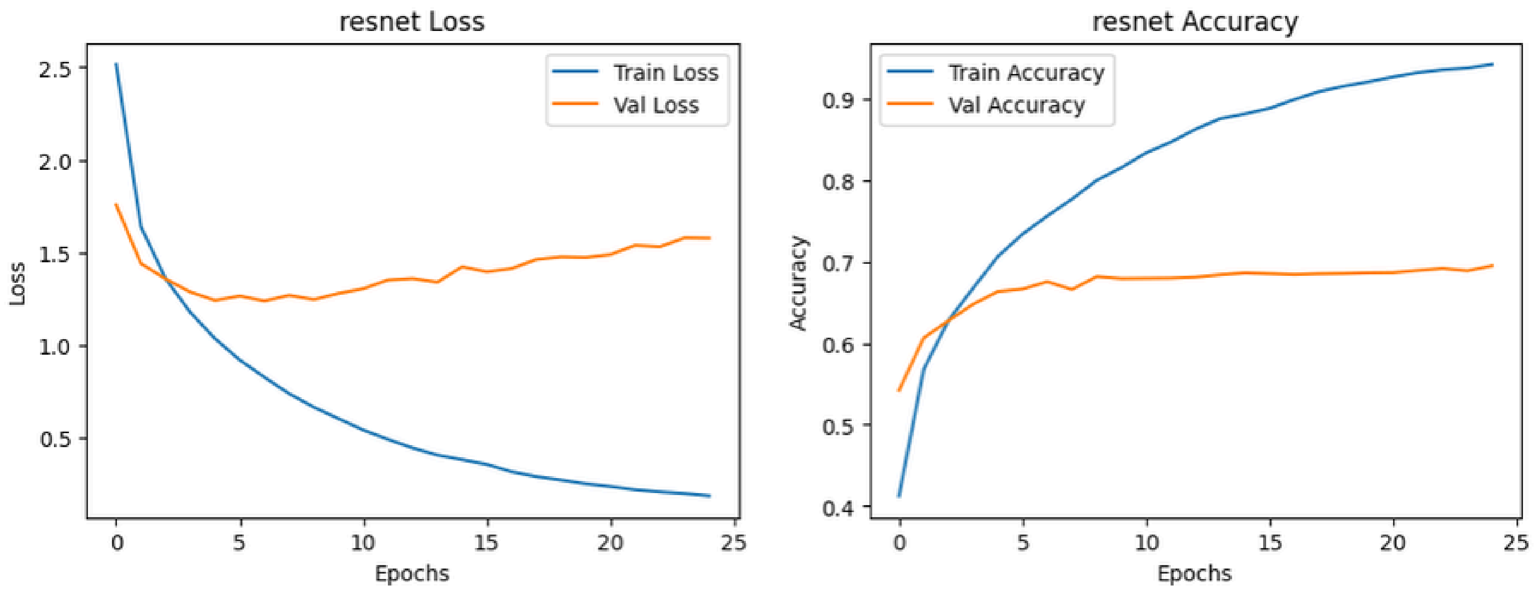
Model	Characteristics	Suitable Scenarios	Principles	Advantages	Disadvantages
AlexNet	Simpler architecture with 8 layers	**Small** to medium-scale datasets	Stacked convolutional layers followed by fully connected layers	**Fast** training and inference; Good for benchmarking and prototyping	**Limited** feature extraction **capacity** ; Prone to overfitting on complex datasets
VGG16	Deep network with 16 layers, using 3x3 convolutions throughout	**grained** classification tasks	Consistent use of 3x3 convolutional layers for simplicity and effectiveness	Strong feature extraction capabilities ; Well-suited for **detailed** image analysis	Large number of parameters ; **High computational load** ; Tendency to overfit on imbalanced datasets
GoogLeNe t	Inception modules capture multi-scale features	**Moderate**-scale datasets	Inception modules perform convolutions with multiple kernel sizes in parallel	**Efficient** computation ; Fewer parameters compared to other deep networks	Not as deep as ResNet50; May **underperform on** very **large** and complex datasets
ResNet50	Deep residual network with 50 layers	**Complex and large**-scale datasets	Utilizes residual connections to ease training of deep networks	Effective for **complex** feature extraction; Excellent transfer learning performance	High computational complexity ; **Longer** training and inference **time**

Parameters of Different Pre-trained Model

Model	Learing Rate	momentum	Weight Decay	Dropout
AlexNet	0.0001	0.9	5e-4	0.7
VGG16	0.0001	0.9	5e-4	0.5
GoogLeNet	0.0001	0.9	5e-4	0.4
ResNet50	0.0001	0.9	5e-4	0.5

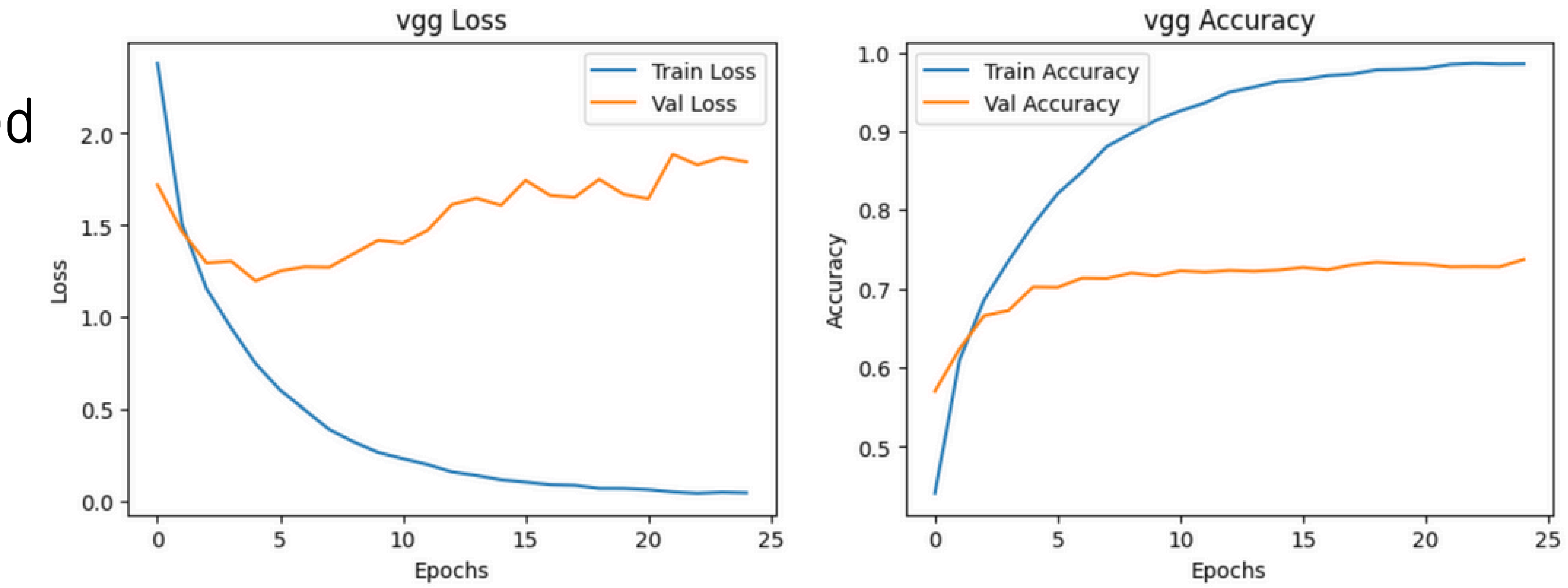
Results with Different Parameters(Overfitting)





Reason: Dropout layers are not added

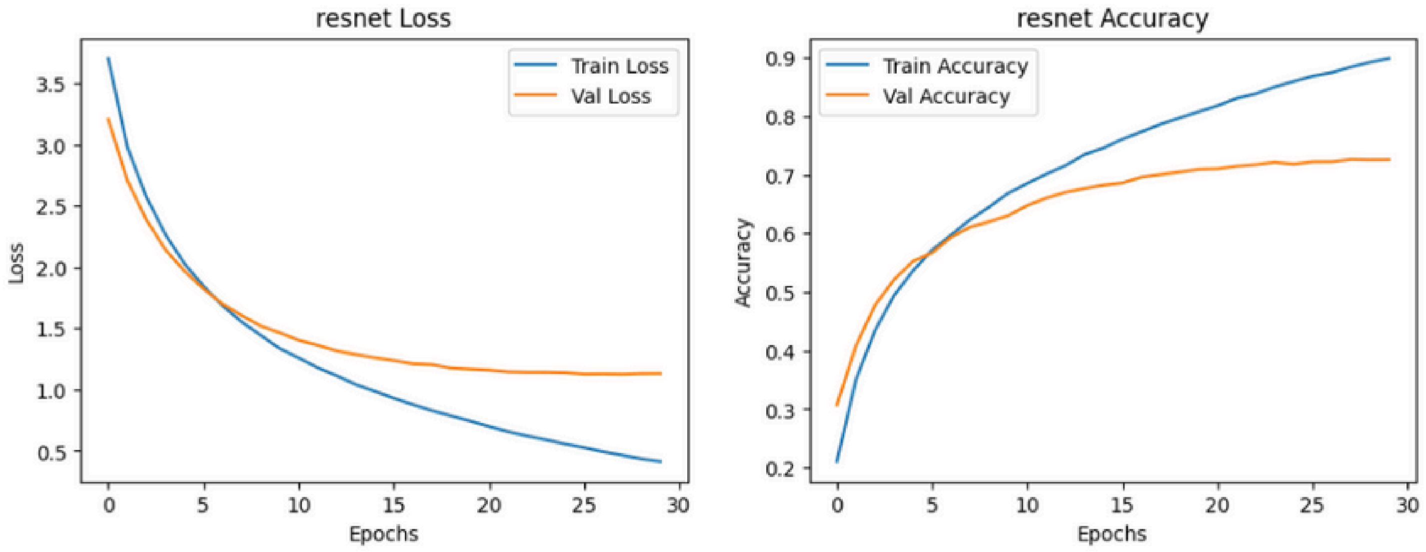
Reason: Learning Rate is set to 0.001



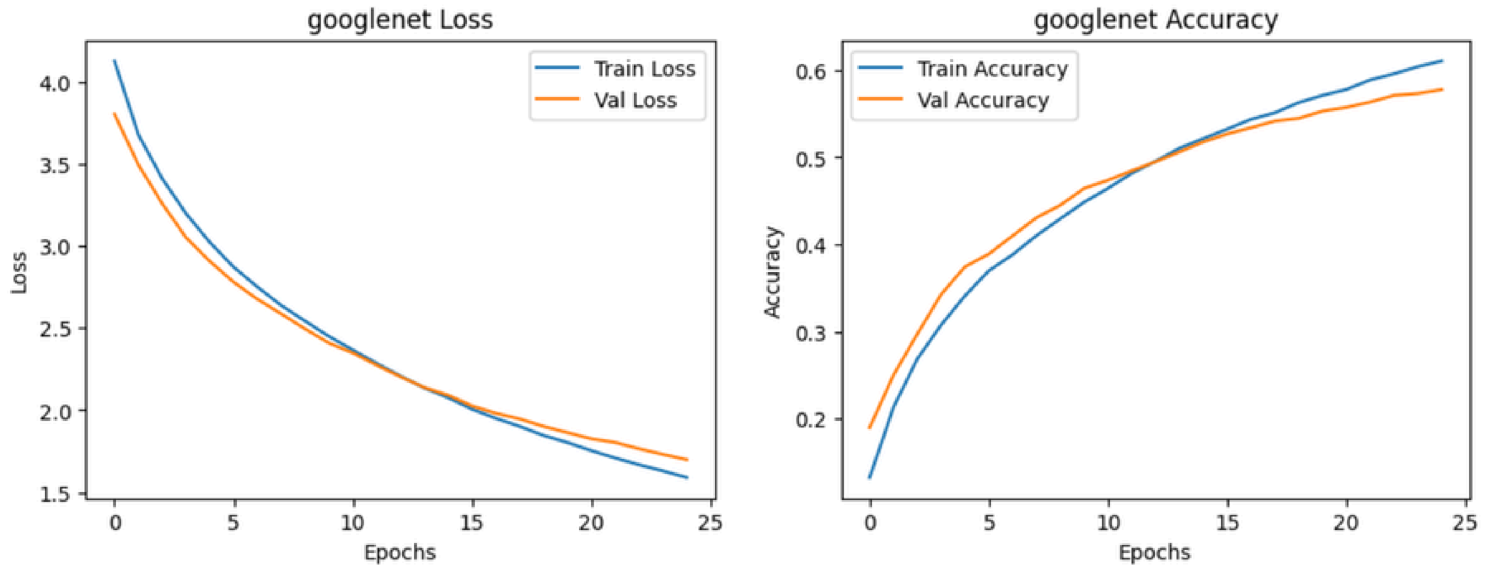
Reason: Model tendency to overfit on imbalanced datasets

Results with Different Parameters

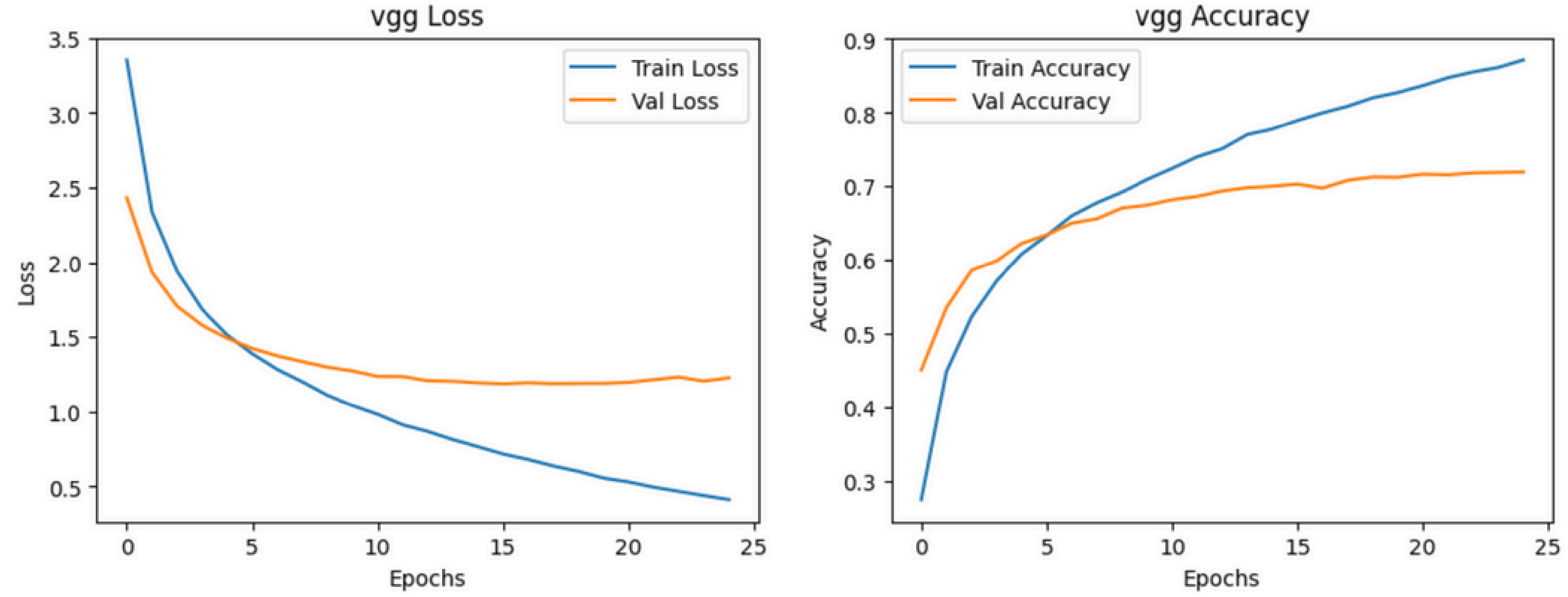
Test Acc: 0.7213



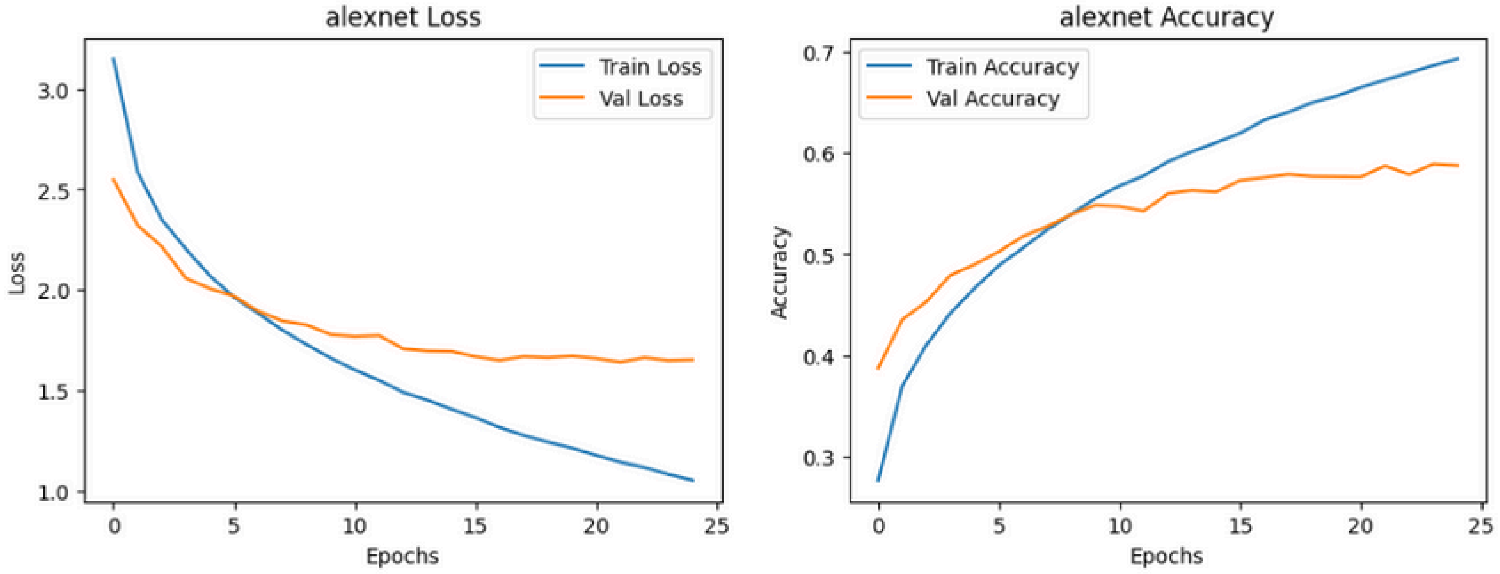
Test Acc: 0.5930



Test Acc: 0.7176



Test Acc: 0.5932



Multi-Scale Attention ResNet (MSA-ResNet)

[3]. W. Linfeng, L. Yong, L. Jiayao, W. Yunsheng, and X. Shipu, “Based on the multi-scale information sharing network of fine-grained attention for agricultural pest detection”, PLOS ONE 18(10):e0286732.
https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0286732

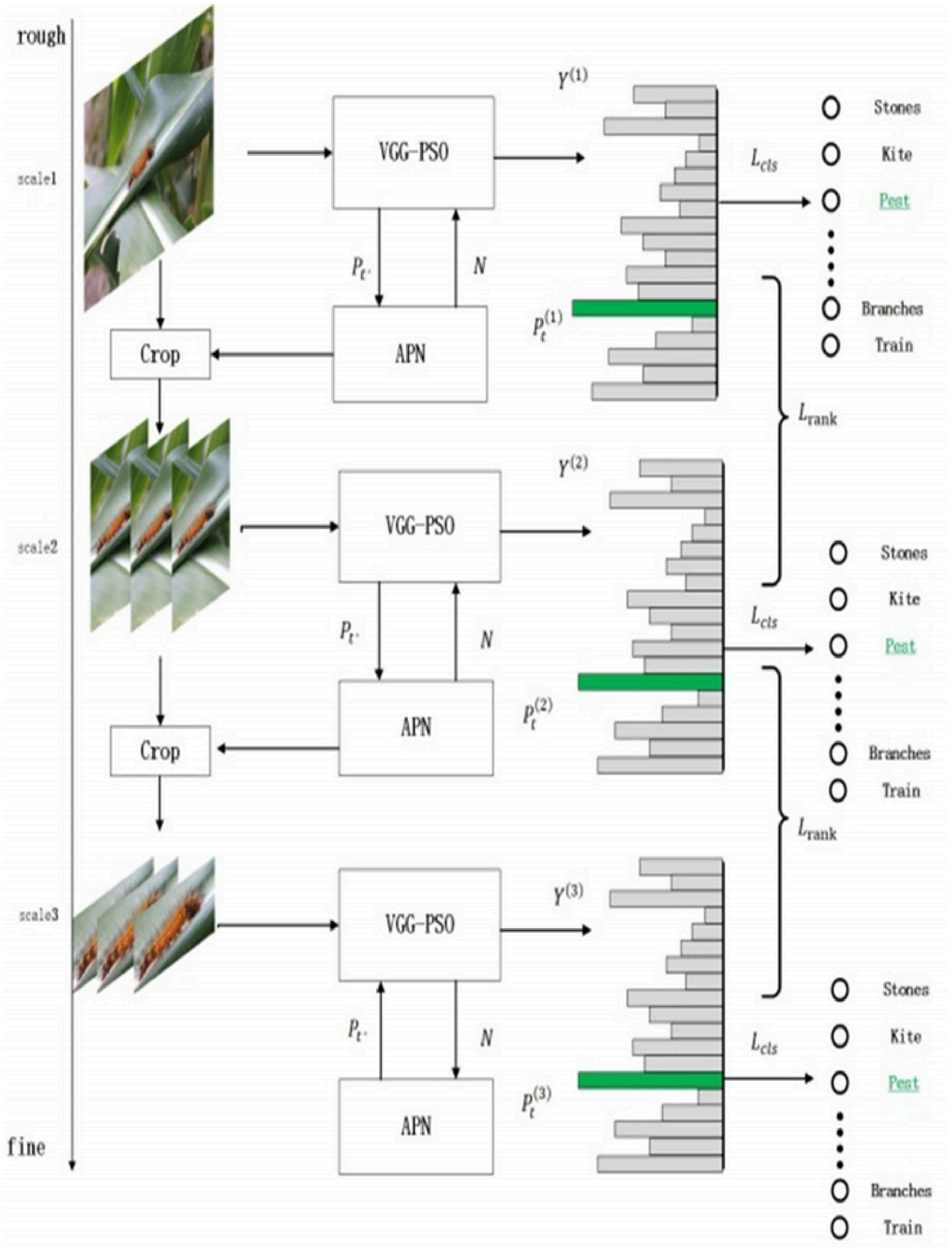


Fig 1. Multi-scale cascading network （Linfeng et al., 2023） .

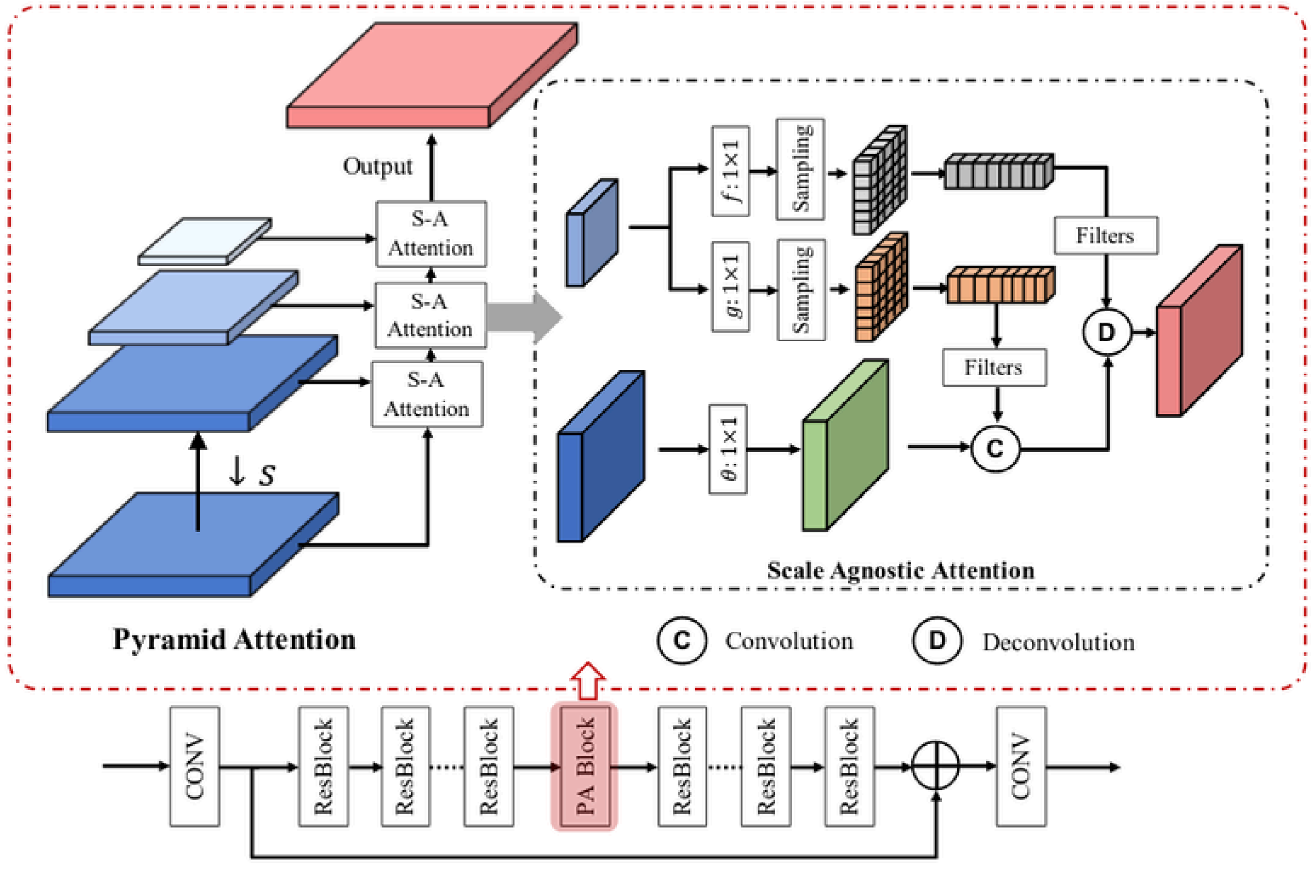


Fig 2. Attention Pyramid Network
https://www.bing.com/images/search?q=Multi-scale+Feature+Extraction&form=HDRSC2&first=1

ChannelAttention:
channel may contain different types of information, such as color, texture, or edge.

SpatialAttention:
helps the model better understand the spatial relationships in the image by emphasizing the important positions in the feature map.

Multi-Scale Attention ResNet (MSA-ResNet)

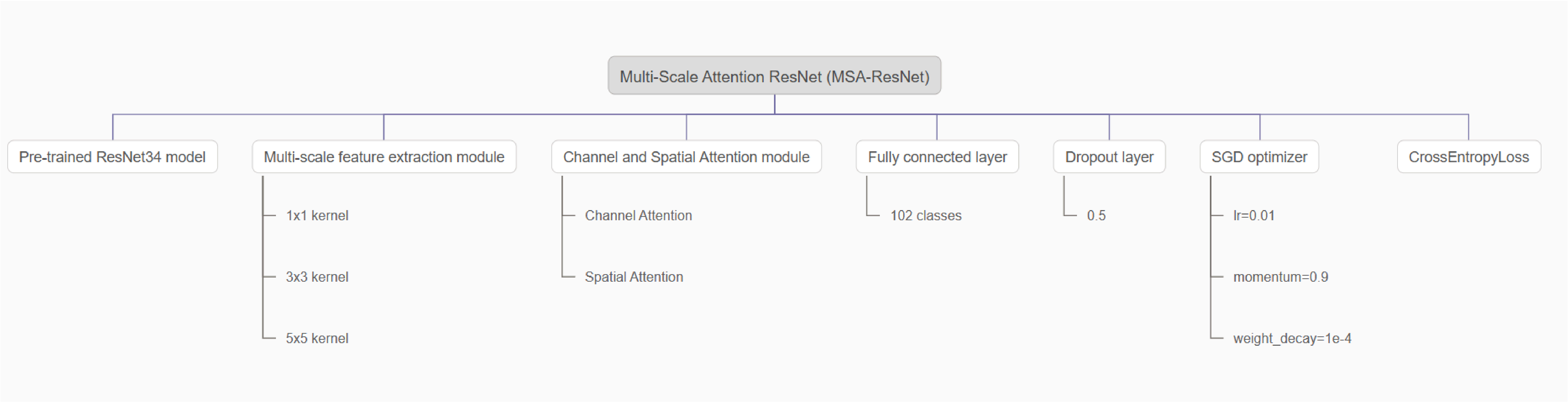


Fig 3. MSA-ResNet model

```
Epoch 1/30
----------
train Loss: 2.9591 Acc: 0.3381
val Loss: 1.9225 Acc: 0.5273
Epoch 2/30
----------
train Loss: 1.6243 Acc: 0.5772
val Loss: 1.4710 Acc: 0.6176
Epoch 3/30
----------
train Loss: 1.1724 Acc: 0.6781
val Loss: 1.3225 Acc: 0.6529
Epoch 4/30
----------
train Loss: 0.8834 Acc: 0.7442
val Loss: 1.2653 Acc: 0.6716
Epoch 5/30
----------
train Loss: 0.7082 Acc: 0.7876
val Loss: 1.3131 Acc: 0.6701
Epoch 6/30
----------
train Loss: 0.5494 Acc: 0.8305
val Loss: 1.3906 Acc: 0.6712
Epoch 7/30
----------
train Loss: 0.4481 Acc: 0.8614
val Loss: 1.4374 Acc: 0.6867
Epoch 8/30
----------
train Loss: 0.2520 Acc: 0.9232
val Loss: 1.2602 Acc: 0.7321
Epoch 9/30
----------
train Loss: 0.1841 Acc: 0.9452
val Loss: 1.3074 Acc: 0.7372
Epoch 10/30
----------
train Loss: 0.1533 Acc: 0.9520
val Loss: 1.3334 Acc: 0.7351
Epoch 11/30
----------
train Loss: 0.1394 Acc: 0.9588
val Loss: 1.3789 Acc: 0.7380
Epoch 12/30
----------
train Loss: 0.1136 Acc: 0.9676
val Loss: 1.4107 Acc: 0.7391
Epoch 13/30
----------
train Loss: 0.1047 Acc: 0.9689
val Loss: 1.4436 Acc: 0.7380
Early stopping triggered. Best epoch: 8
```

Fig 5. Images of the training process

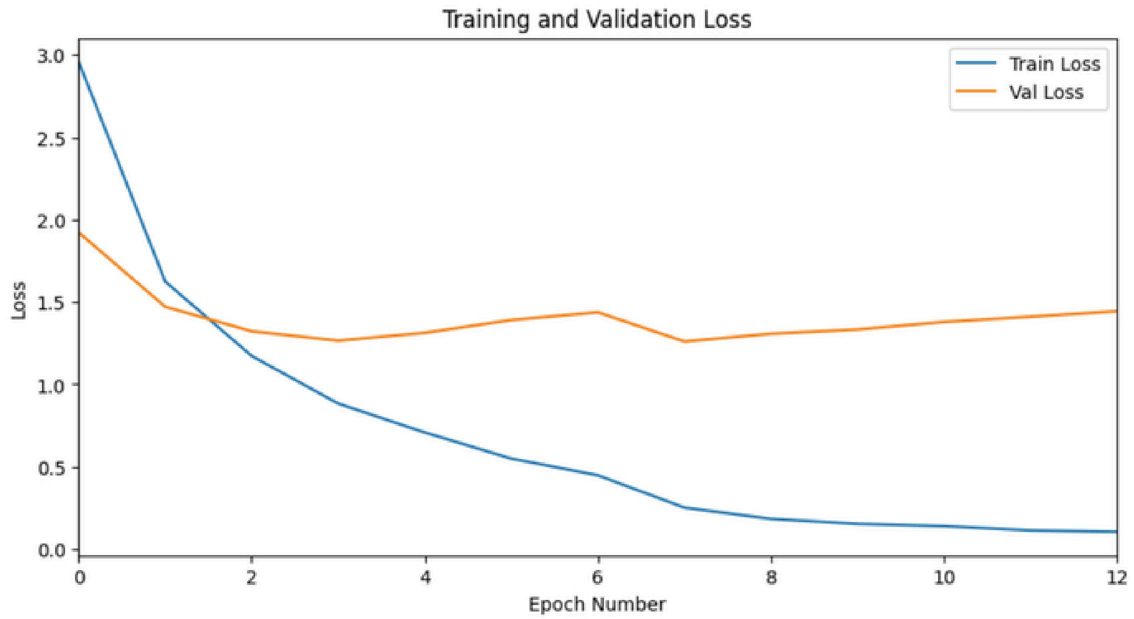


Fig 4. loss graph

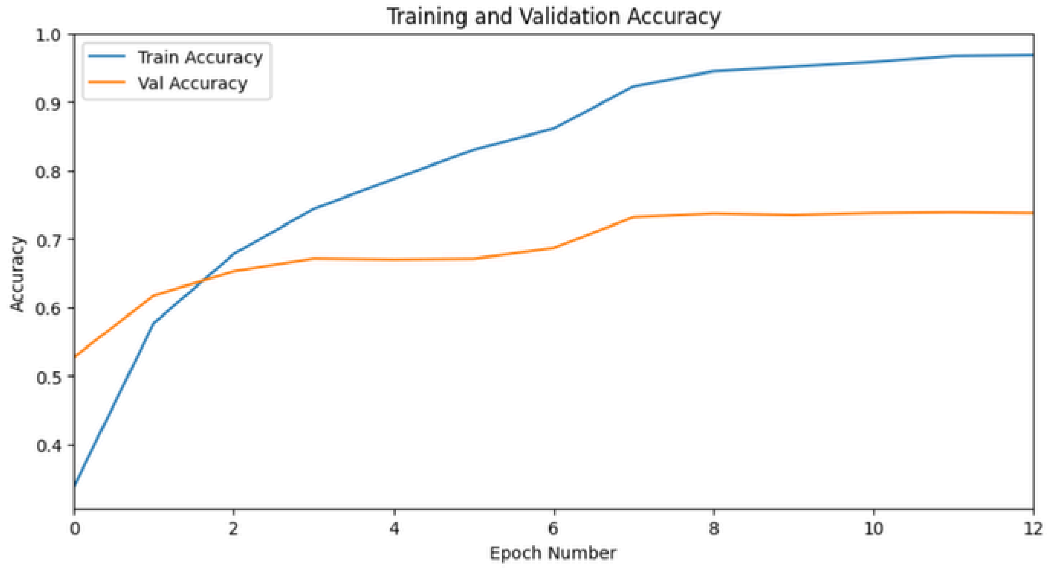


Fig 5. Accuracy graph

Presenter: Zhikai Lai(z5412820)

Multi-Scale Attention ResNet (MSA-ResNet)

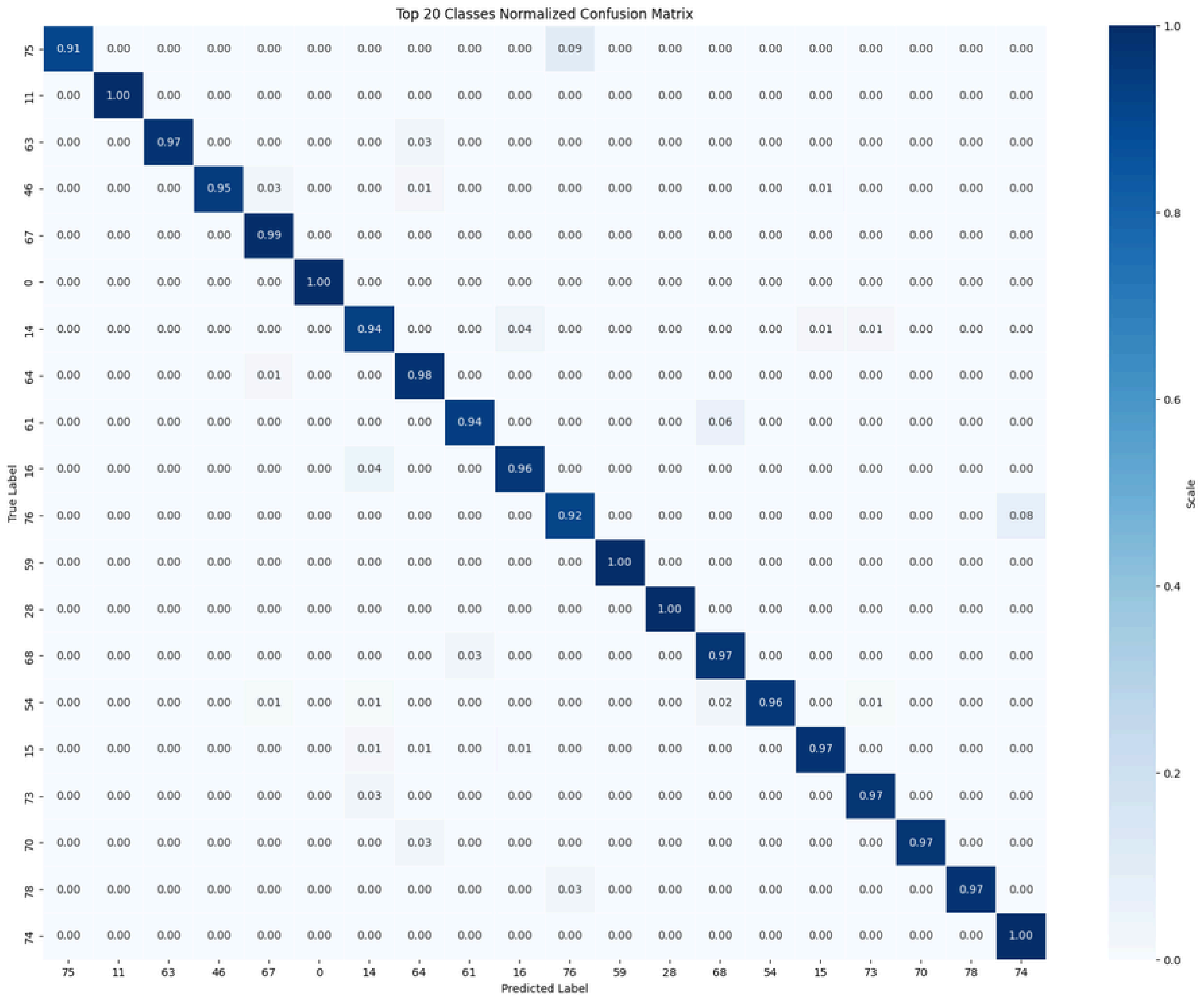


Fig 7. heatmap

Test Loss: 1.2637, Test Accuracy: 72.52%

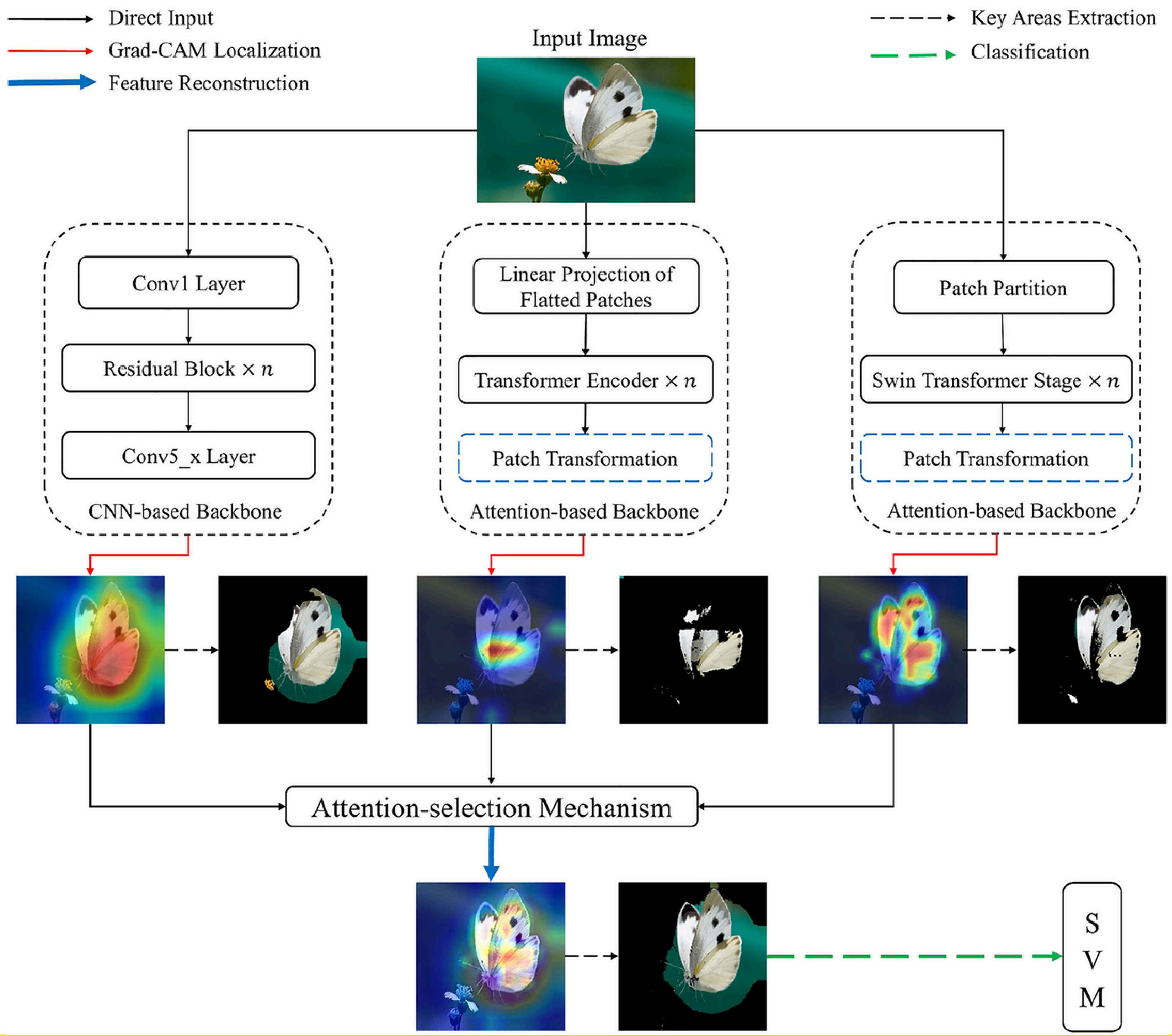
Fig 6. Test set result

Presenter: Zhikai Lai(z5412820)

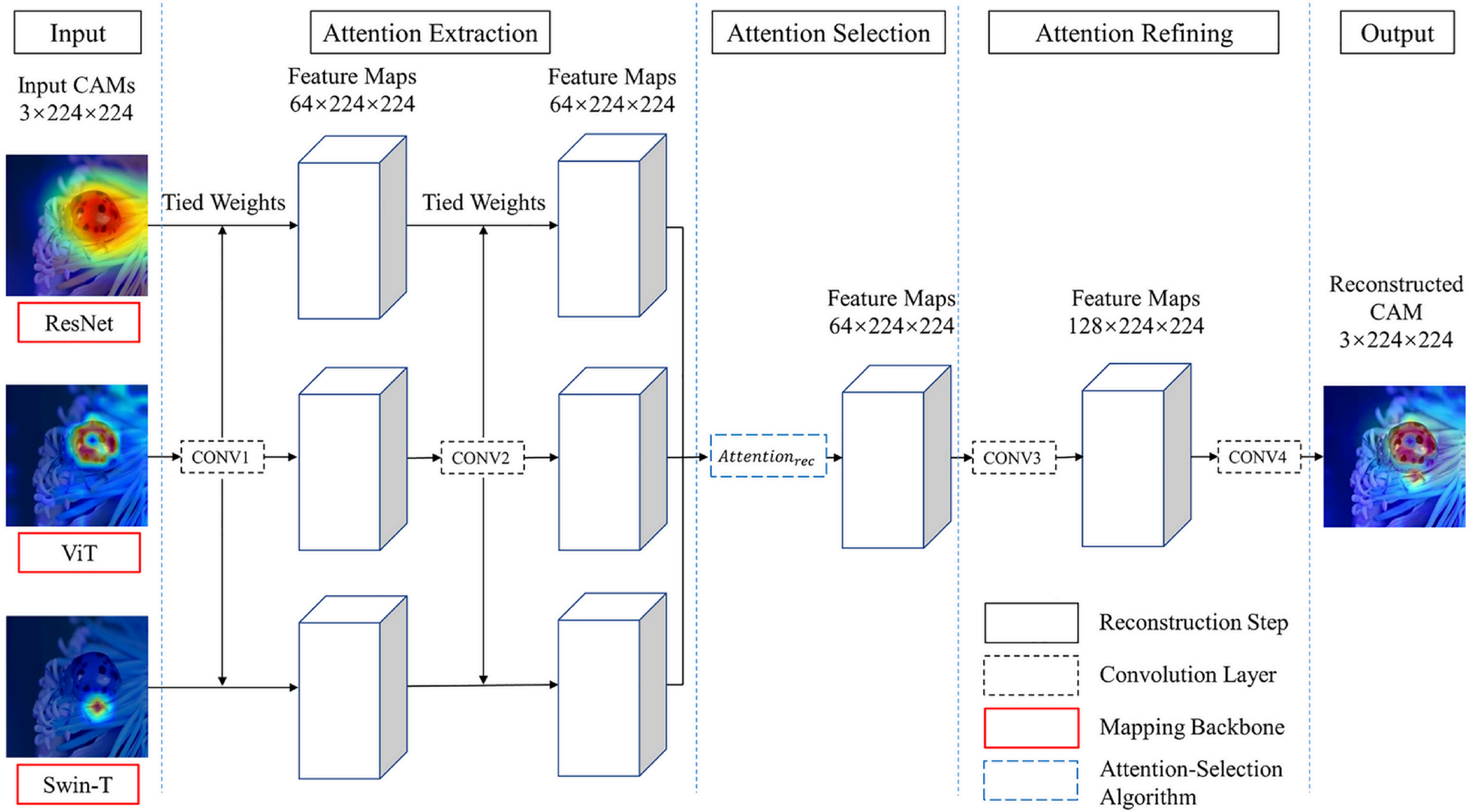
UNSW SYDNEY

Method(s)

- A feature recognition based on multi-model complementary extraction



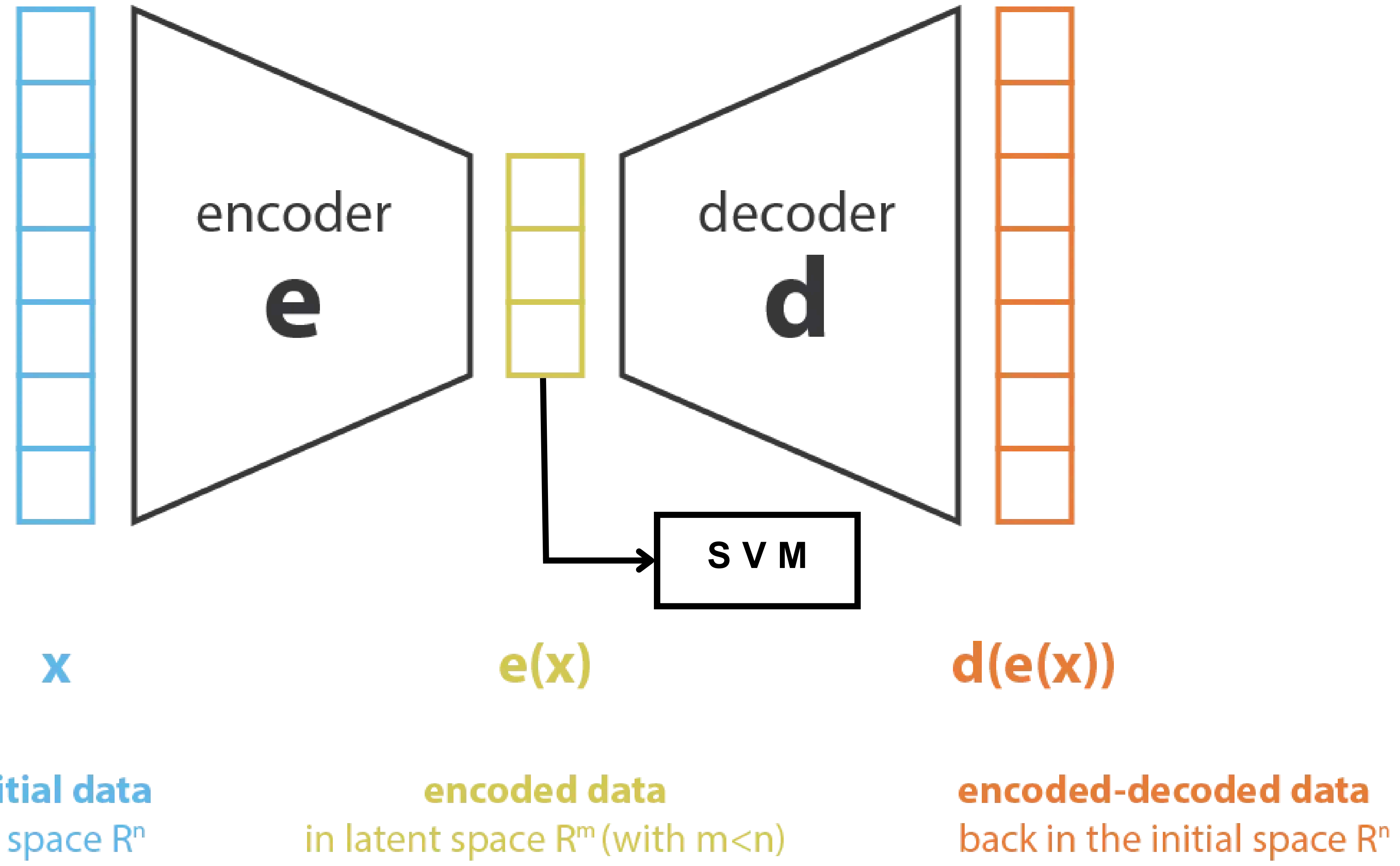
Method(s)



Input
Input CAMs
3×224×224
ResNet
ViT
Swin-T
Tied Weights
CONV1
Attention Extraction
Feature Maps
64×224×224
Tied Weights
CONV2
Feature Maps
64×224×224
Attention Selection
Attention$_{rec}$
Feature Maps
64×224×224
Attention Refining
CONV3
Feature Maps
128×224×224
CONV4
Output
Reconstructed
CAM
3×224×224
Reconstruction Step
Convolution Layer
Mapping Backbone
Attention-Selection
Algorithm

Method(s)

- A method for extracting features based on image generation using Variational Autoencoder (VAE)

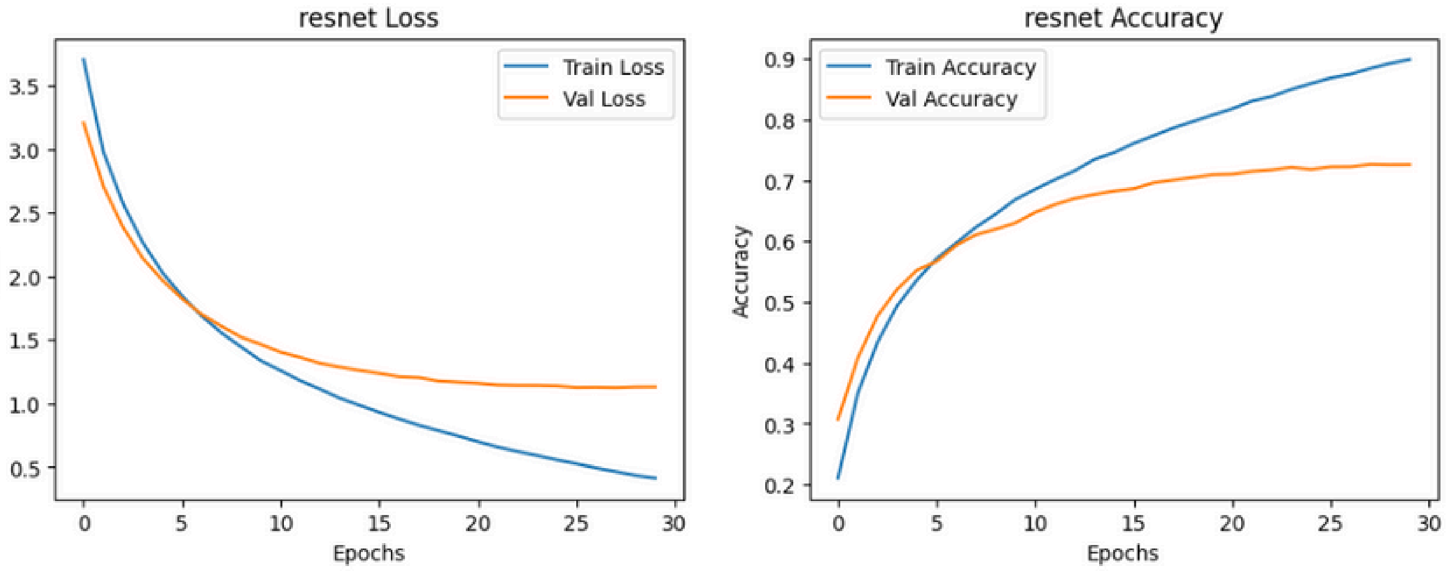


$x = d(e(x))$ ➡ **lossless encoding**
no information is lost when reducing the number of dimensions

$x \neq d(e(x))$ ➡ **lossy encoding**
some information is lost when reducing the number of dimensions and can't be recovered later

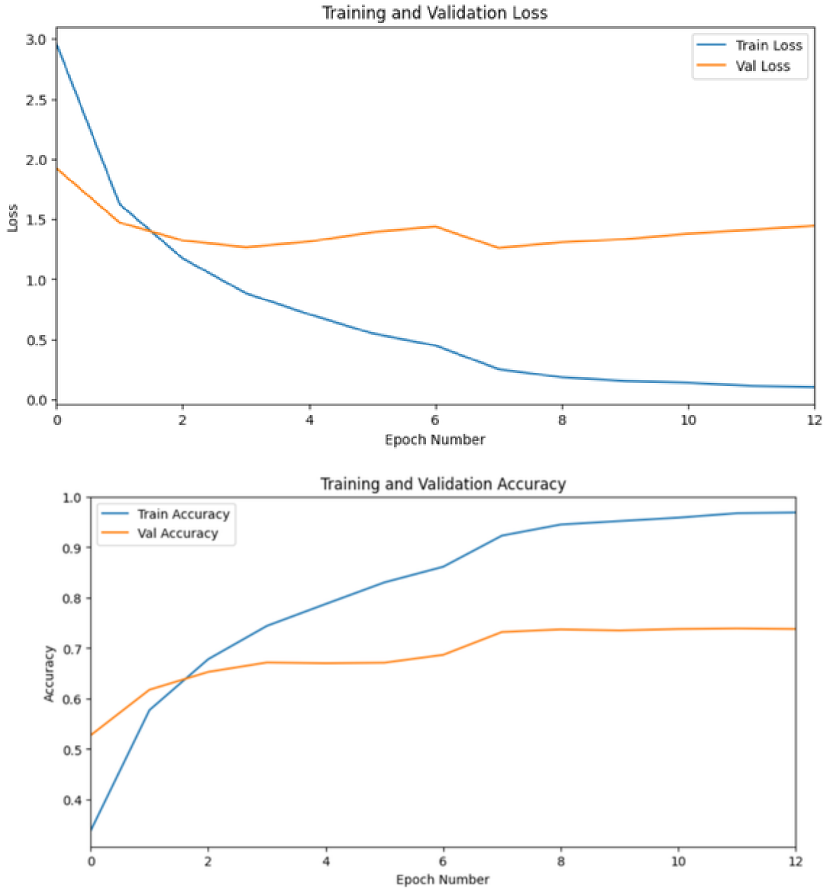
Result and Discussion

ResNet50



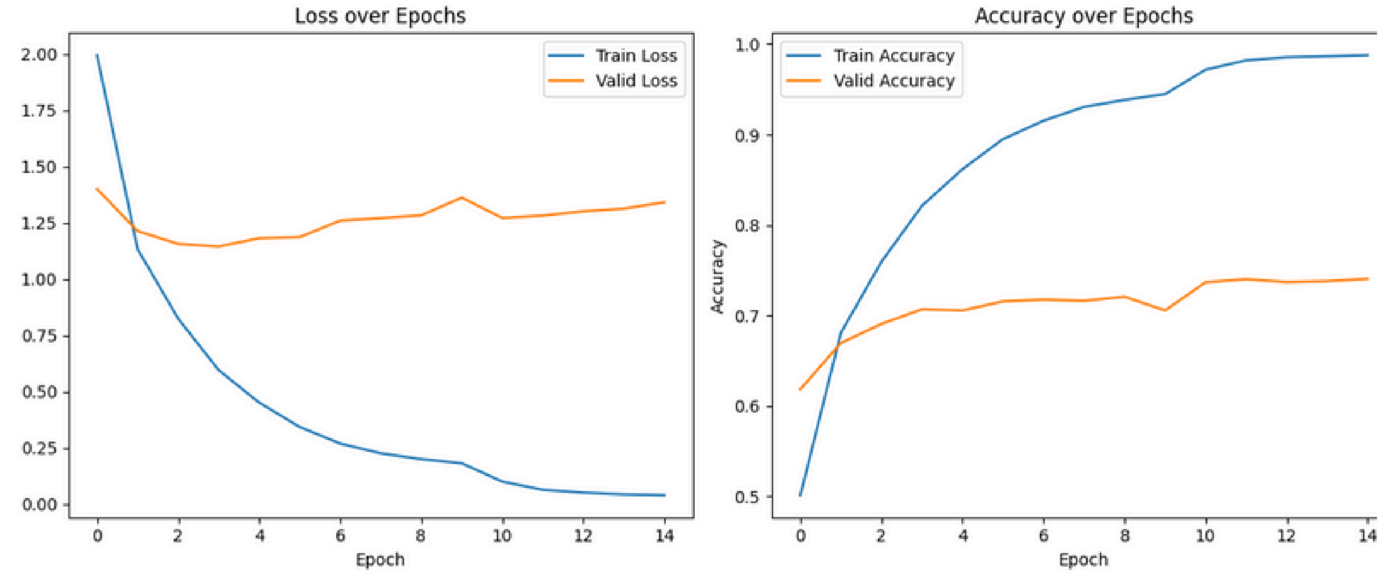
Test Accuracy: 0.72

MSA-ResNet



Test Accuracy: 0.7252

Swin-T



Test Accuracy: 0.744

Conclusion

Model Selection ,Optimization and Improvements

Adjusting the learning rate

Adding Dropout layer and

Introducing attention mechanism

Effect and Necessity of Improvements

Achieve better performance on large-scale

Complex datasets

Future Work

Explore hybrid models

Reduce overfitting


Questions