

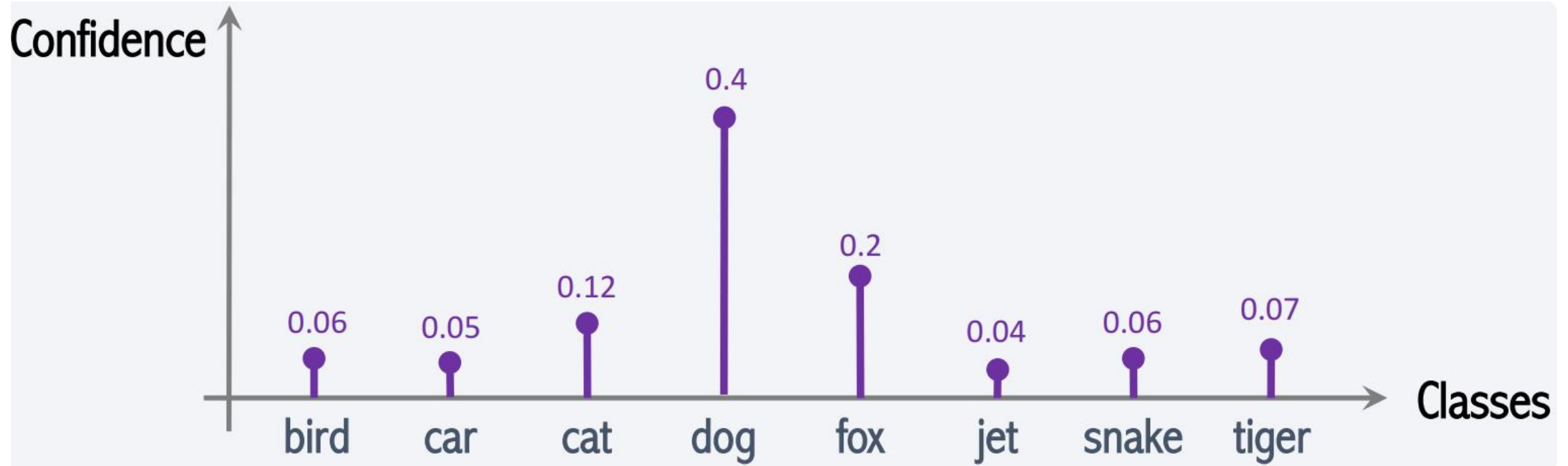
# **Vision Transformer (ViT)**



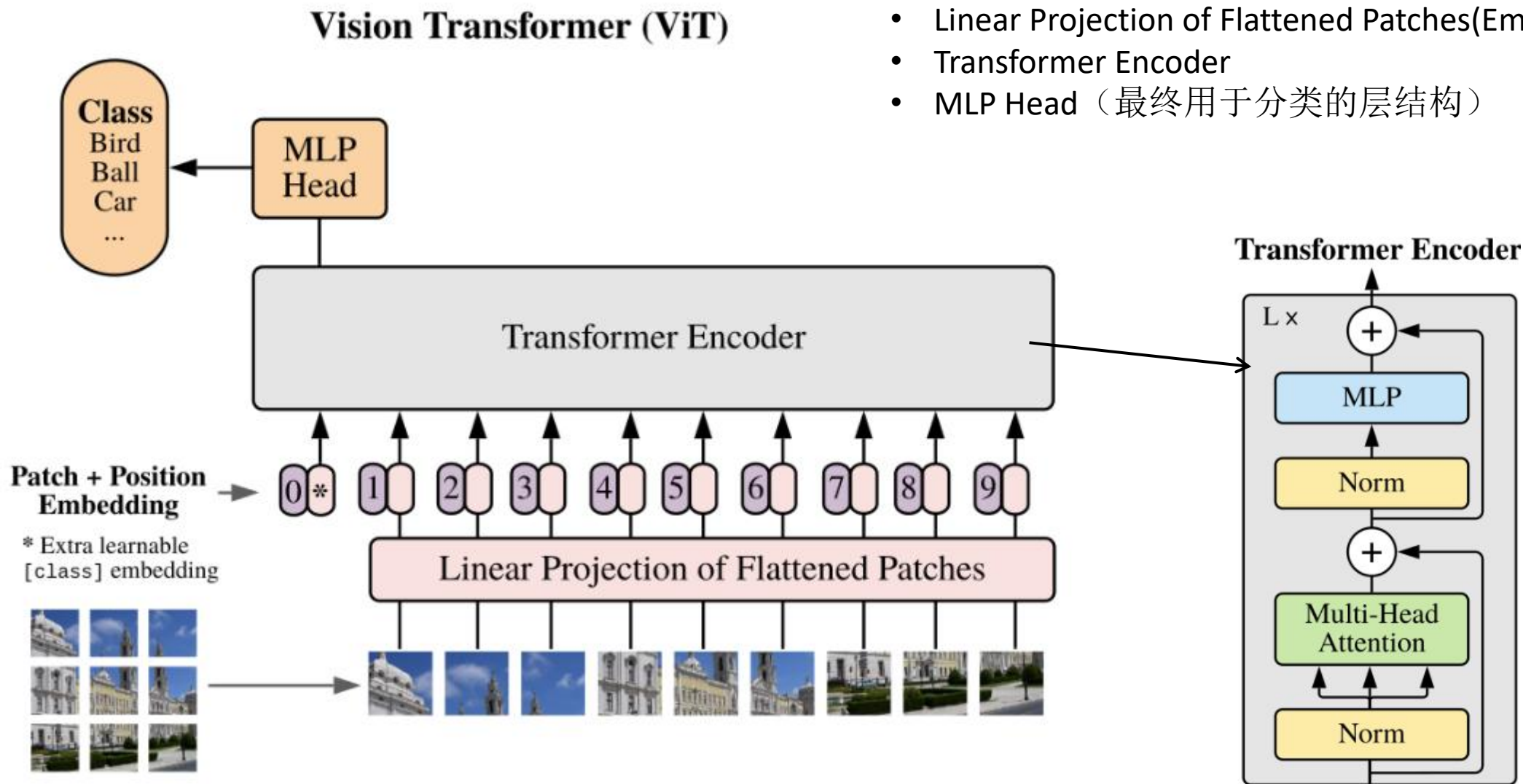
Neural  
Network



P



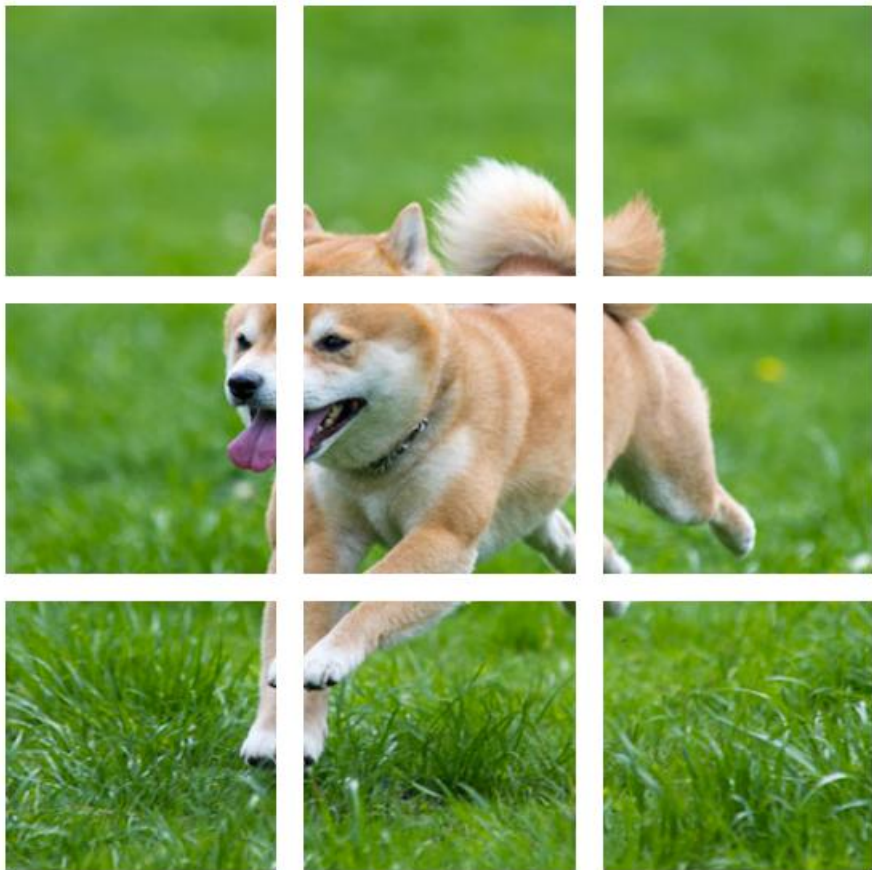
下图是原论文中给出的关于Vision Transformer(ViT)的模型框架。



模型由三个模块组成:

- Linear Projection of Flattened Patches(Embedding层)
- Transformer Encoder
- MLP Head (最终用于分类的层结构)

# Split Image into Patches

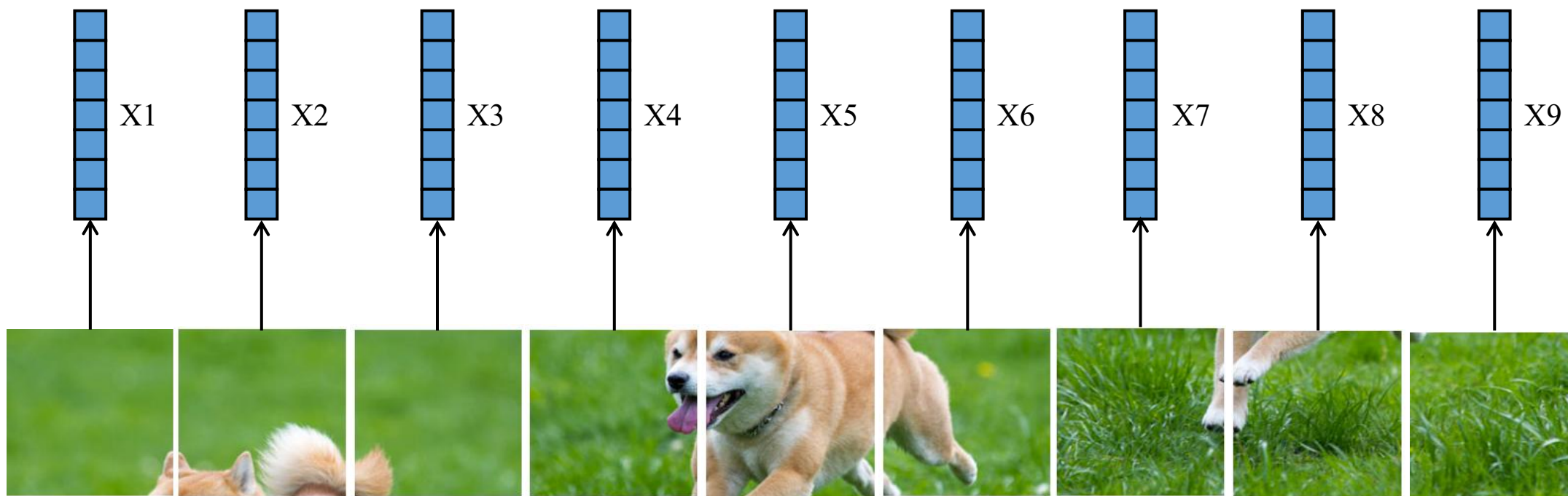


- Here, the patches do not overlap.
- The patches can overlap.
- User specifies:
  - **patch size**, e.g.,  $16 \times 16$ ;
  - **stride**, e.g.,  $16 \times 16$ .

# Vectorization

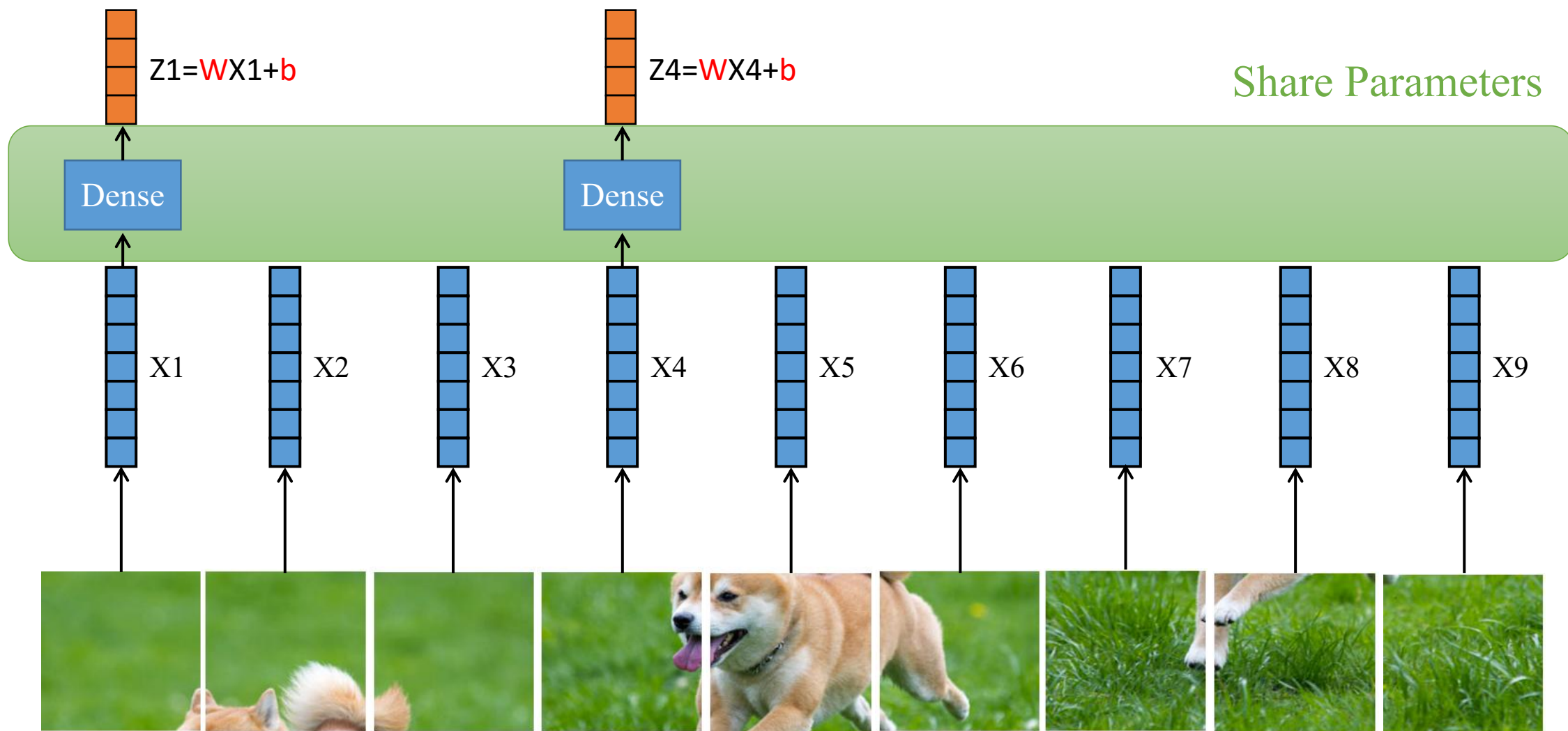


# Vectorization

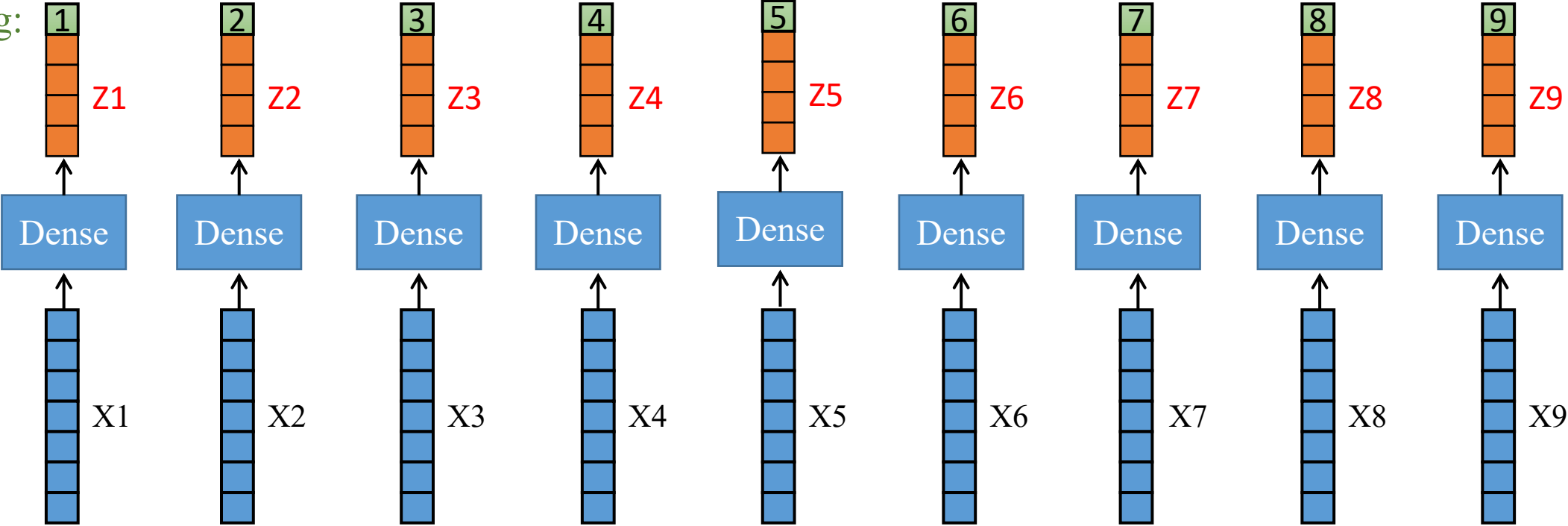




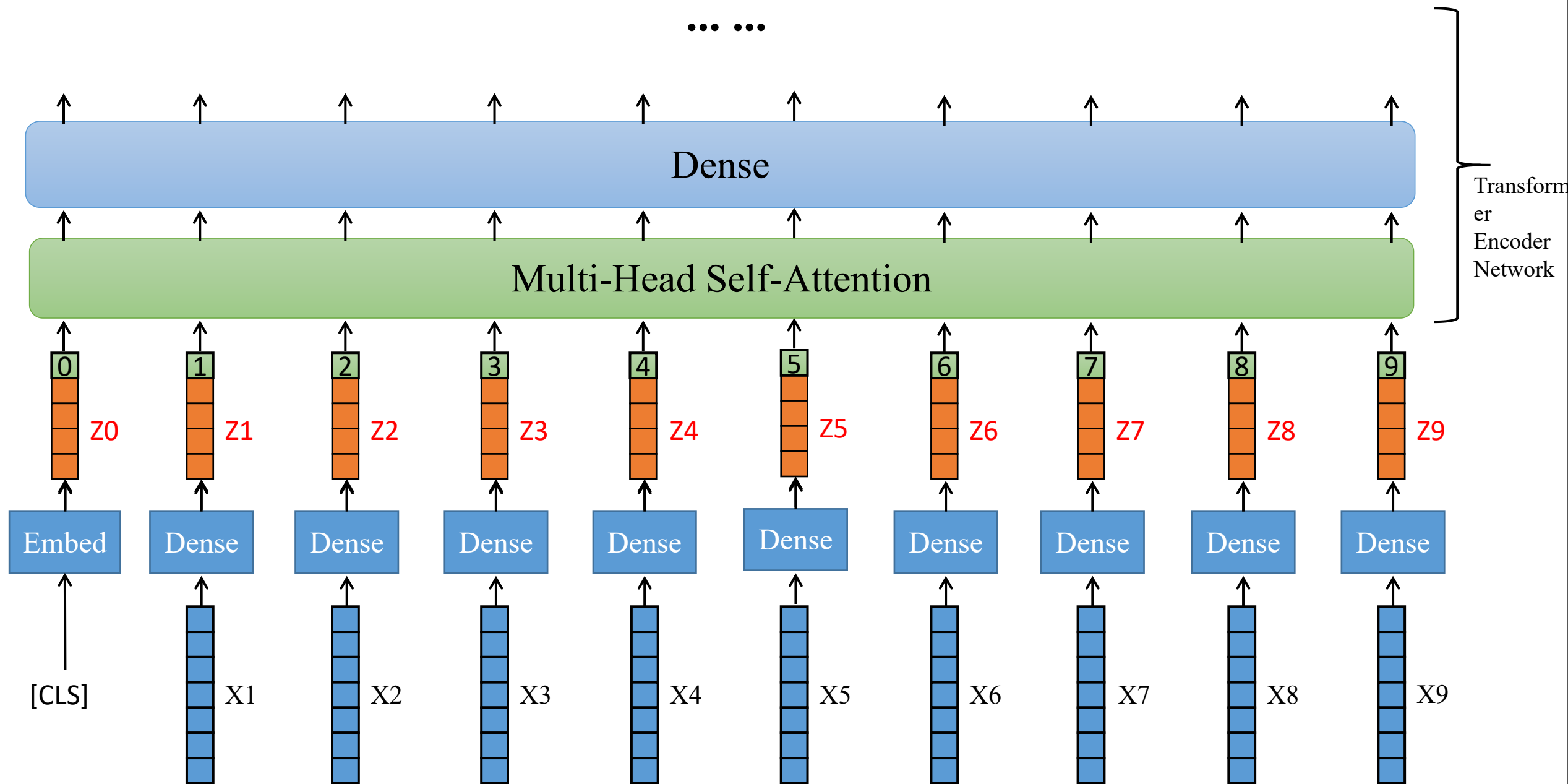
Share Parameters

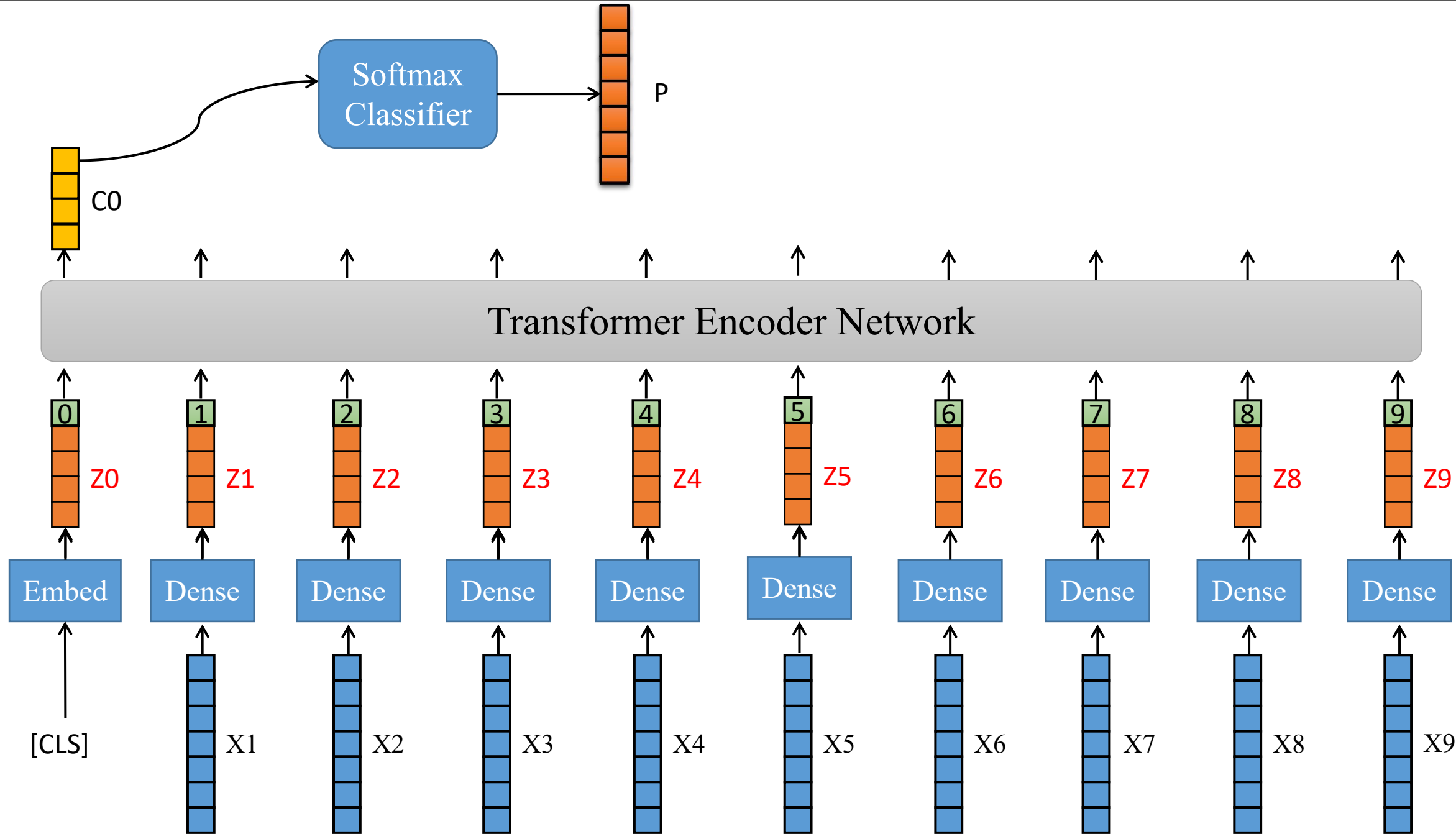


Positional  
Encoding:







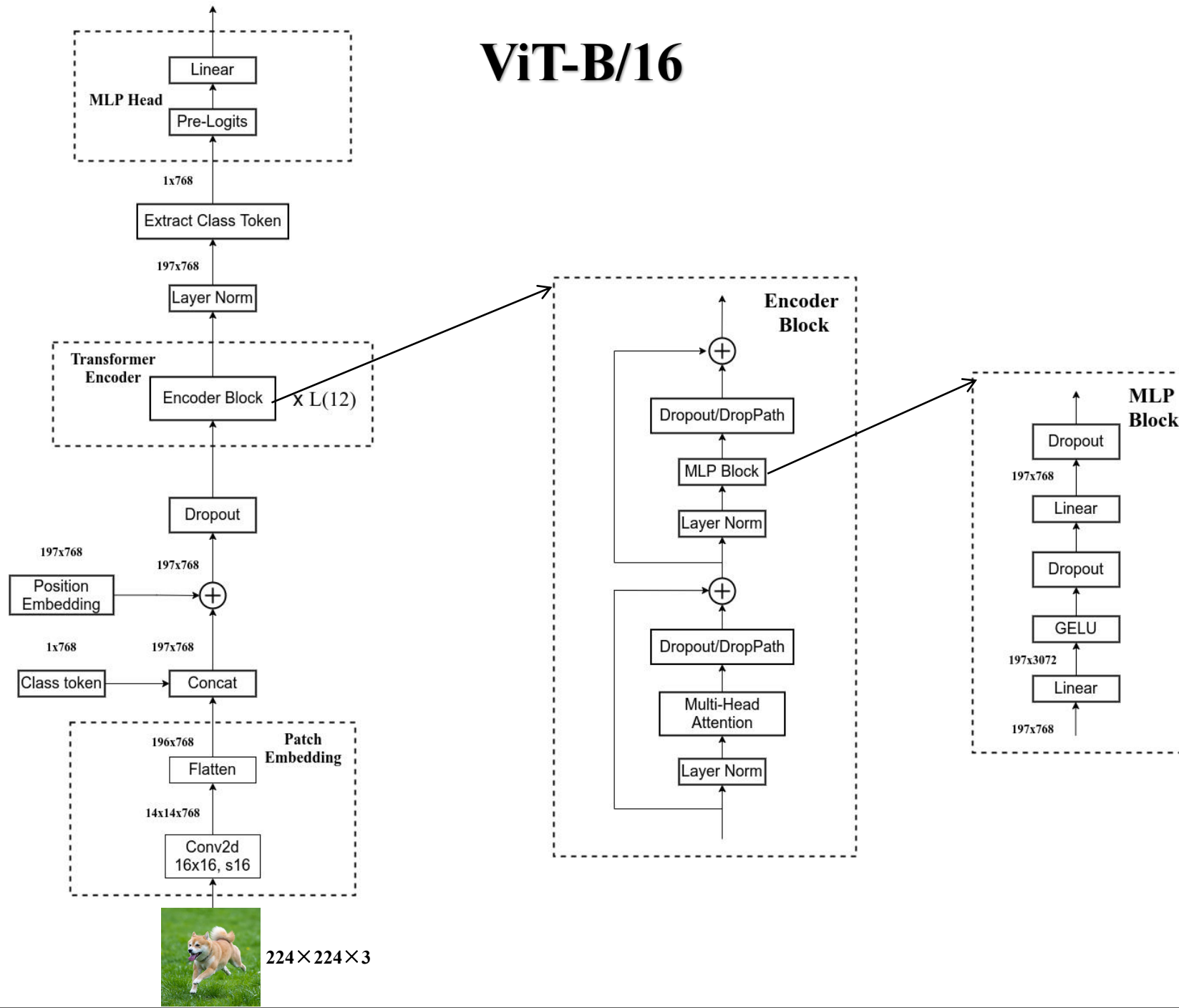


# ViT-B/16

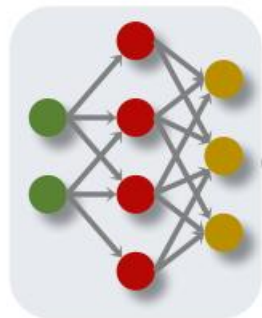
Part3

Part2

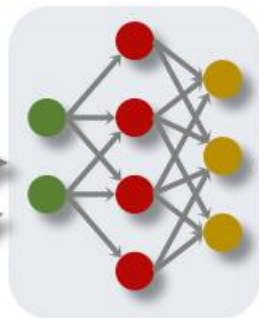
Part1



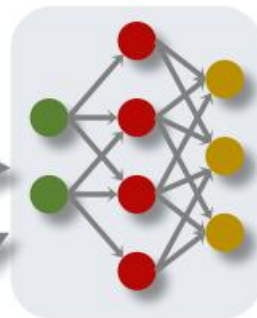
Randomly  
Initialized



Pretrained



Fine-tuned

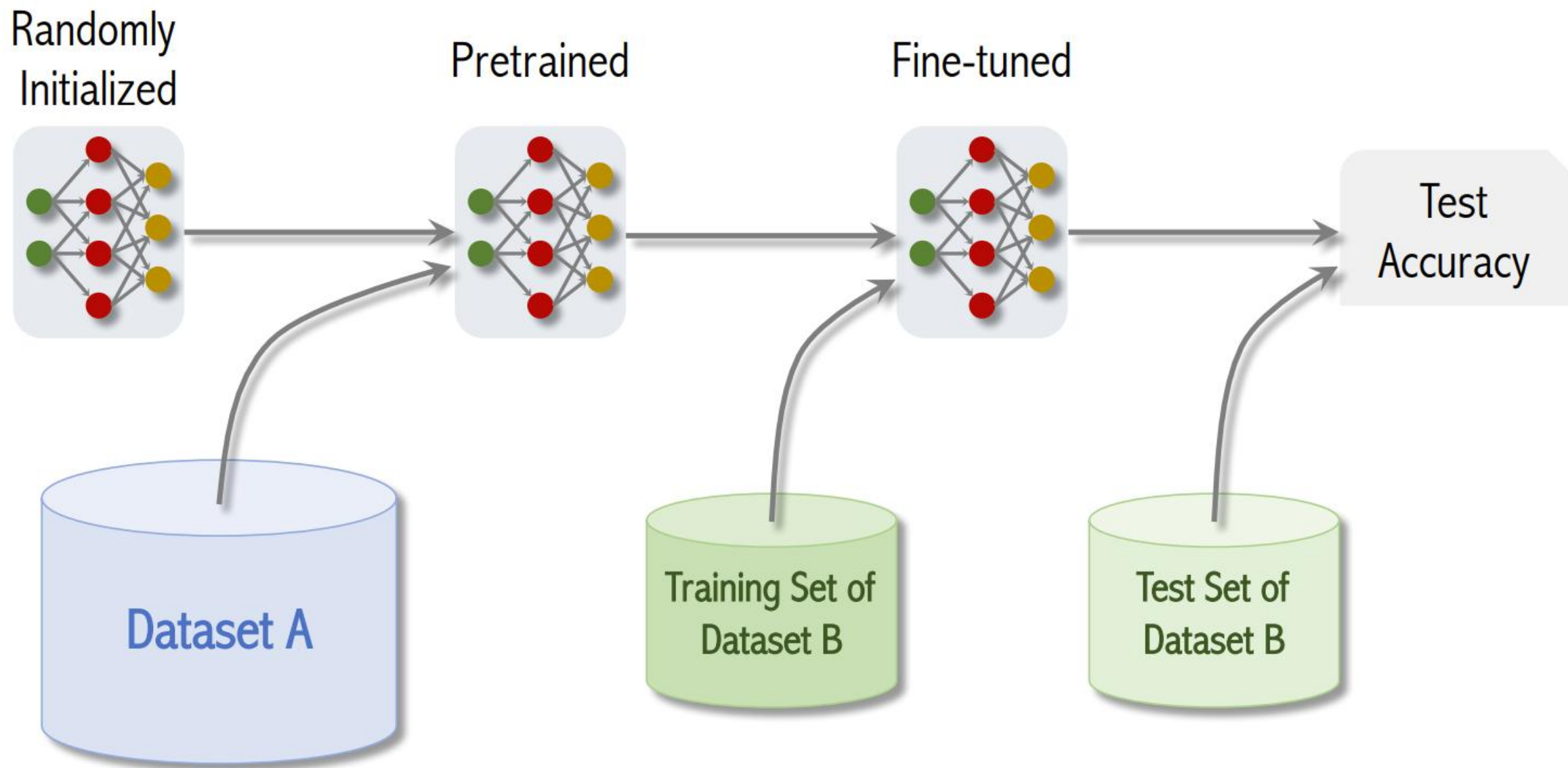


Test  
Accuracy

Dataset A

Training Set of  
Dataset B

Test Set of  
Dataset B



1

资源管理器

CODE

> \_\_pycache\_\_

> .idea

> flower\_photos

> runs

> weights

class\_indices.json

flops.py

my\_dataset.py

predict.py

train.py

utils.py

vit\_base\_patch16\_224.pth

vit\_model.py

train.py

vit\_model.py

train.py > ...

```
115
116 if __name__ == '__main__':
117     parser = argparse.ArgumentParser()
118     parser.add_argument('--num_classes', type=int, default=5)
119     parser.add_argument('--epochs', type=int, default=10)
120     parser.add_argument('--batch-size', type=int, default=8)
121     parser.add_argument('--lr', type=float, default=0.001)
122     parser.add_argument('--lrf', type=float, default=0.01)
123
124     # 数据集所在根目录
125     # https://storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz
126     parser.add_argument('--data-path', type=str,
127                         |         default="C:/Users/86198/Desktop/vit/code/flower_photos/flower_photos")
128     parser.add_argument('--model-name', default='vit_base_patch16_224', help='create model name')
129
130     # 预训练权重路径
131     parser.add_argument('--weights', type=str, default='',
132                         |         help='initial weights path')
133     # 是否冻结权重
134     parser.add_argument('--freeze-layers', type=bool, default=True)
135     parser.add_argument('--device', default='cuda:0', help='device id (i.e. 0 or 0,1 or cpu)')
136
137     opt = parser.parse_args()
138
```

问题

输出

调试控制台

终端

[train epoch 4] loss: 1.315, acc: 0.453: 100%

[valid epoch 4] loss: 1.284, acc: 0.423: 100%

[train epoch 5] loss: 1.281, acc: 0.470: 100%

[valid epoch 5] loss: 1.310, acc: 0.450: 100%

[train epoch 6] loss: 1.231, acc: 0.485: 100%

[valid epoch 6] loss: 1.233, acc: 0.503: 100%

[train epoch 7] loss: 1.231, acc: 0.484: 100%

[valid epoch 7] loss: 1.196, acc: 0.495: 100%

[train epoch 8] loss: 1.201, acc: 0.504: 100%

[valid epoch 8] loss: 1.187, acc: 0.506: 100%

[train epoch 9] loss: 1.193, acc: 0.502: 100%

[valid epoch 9] loss: 1.173, acc: 0.529: 100%

368/368 [00:54<00:00, 6.78it/s]

92/92 [00:28<00:00, 3.19it/s]

368/368 [00:55<00:00, 6.64it/s]

92/92 [00:28<00:00, 3.19it/s]

368/368 [00:55<00:00, 6.62it/s]

92/92 [00:28<00:00, 3.17it/s]

368/368 [00:55<00:00, 6.64it/s]

92/92 [00:29<00:00, 3.17it/s]

368/368 [00:55<00:00, 6.59it/s]

92/92 [00:28<00:00, 3.18it/s]

368/368 [00:55<00:00, 6.60it/s]

92/92 [00:28<00:00, 3.18it/s]

PS C:\Users\86198\Desktop\vit\code>

行 130, 列 14 空格: 4 UTF-8 LF Python 3.8.16 ('torch': conda)

21:20 2023-06-05



资源管理器

CODE

> \_\_pycache\_\_

> .idea

> flower\_photos

> runs

> weights

{ } class\_indices.json

🔗 flops.py

🔗 my\_dataset.py

🔗 predict.py

🔗 train.py

🔗 utils.py

≡ vit\_base\_patch16\_224.pth

🔗 vit\_model.py

train.py

vit\_model.py

train.py > main

```
75
76     if args.freeze_layers:
77         for name, para in model.named_parameters():
78             # 除head, pre_logits外, 其他权重全部冻结
79             if "head" not in name and "pre_logits" not in name:
80                 para.requires_grad_(False)
81             else:
82                 print("training {}".format(name))
83
84     pg = [p for p in model.parameters() if p.requires_grad]
85     optimizer = optim.SGD(pg, lr=args.lr, momentum=0.9, weight_decay=5E-5)
86     # Scheduler https://arxiv.org/pdf/1812.01187.pdf
87     lf = lambda x: ((1 + math.cos(x * math.pi / args.epochs)) / 2) * (1 - args.lrf) + args.lrf # cosine
88     scheduler = lr_scheduler.LambdaLR(optimizer, lr_lambda=lf)
89
90     for epoch in range(args.epochs):
91         # train
92         train_loss, train_acc = train_one_epoch(model=model,
93                                                 optimizer=optimizer,
94                                                 data_loader=train_loader,
95                                                 device=device,
96                                                 epoch=epoch)
97
98         scheduler.step()
```

问题

输出

调试控制台

终端

Python

🗑

⌵

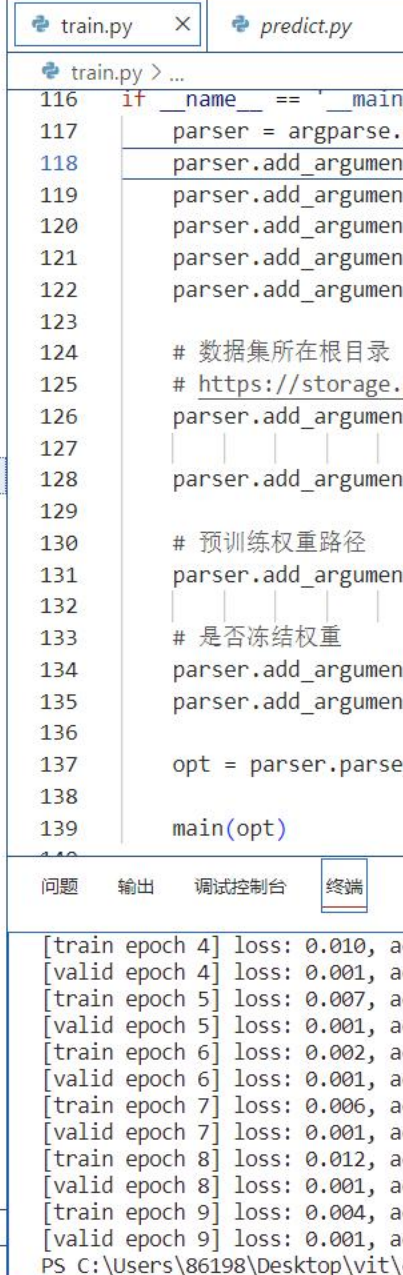
⌶

⌵

⌶

```
[train epoch 4] loss: 0.096, acc: 0.978: 100% | 368/368 [00:55<00:00, 6.61it/s]
[valid epoch 4] loss: 0.130, acc: 0.973: 100% | 92/92 [00:31<00:00, 2.92it/s]
[train epoch 5] loss: 0.077, acc: 0.980: 100% | 368/368 [00:57<00:00, 6.45it/s]
[valid epoch 5] loss: 0.102, acc: 0.977: 100% | 92/92 [00:29<00:00, 3.16it/s]
[train epoch 6] loss: 0.074, acc: 0.984: 100% | 368/368 [00:55<00:00, 6.65it/s]
[valid epoch 6] loss: 0.096, acc: 0.982: 100% | 92/92 [00:28<00:00, 3.18it/s]
[train epoch 7] loss: 0.045, acc: 0.988: 100% | 368/368 [00:55<00:00, 6.60it/s]
[valid epoch 7] loss: 0.099, acc: 0.982: 100% | 92/92 [00:29<00:00, 3.16it/s]
[train epoch 8] loss: 0.059, acc: 0.986: 100% | 368/368 [00:55<00:00, 6.66it/s]
[valid epoch 8] loss: 0.102, acc: 0.982: 100% | 92/92 [00:28<00:00, 3.18it/s]
[train epoch 9] loss: 0.049, acc: 0.987: 100% | 368/368 [00:56<00:00, 6.56it/s]
[valid epoch 9] loss: 0.102, acc: 0.982: 100% | 92/92 [00:29<00:00, 3.17it/s]
```

PS C:\Users\86198\Desktop\vit\code>



 *predict.py*

```

116 if __name__ == '__main__':
117     parser = argparse.ArgumentParser()
118     parser.add_argument('--num_classes', type=int, default=7)
119     parser.add_argument('--epochs', type=int, default=10)
120     parser.add_argument('--batch-size', type=int, default=8)
121     parser.add_argument('--lr', type=float, default=0.001)
122     parser.add_argument('--lrf', type=float, default=0.01)
123
124     # 数据集所在根目录
125     # https://storage.googleapis.com/download.tensorflow.org/example\_images/flower\_photos.tgz
126     parser.add_argument('--data-path', type=str,
127                         | default="C:/Users/86198/Desktop/vit/code/cifar-7")
128     parser.add_argument('--model-name', default='vit_base_patch16_224', help='create model name')
129
130     # 预训练权重路径
131     parser.add_argument('--weights', type=str, default='C:/Users/86198/Desktop/vit/code/vit_base_patch16_224.pth',
132                         | help='initial weights path')
133     # 是否冻结权重
134     parser.add_argument('--freeze-layers', type=bool, default=True)
135     parser.add_argument('--device', default='cuda:0', help='device id (i.e. 0 or 0,1 or cpu)')
136
137     opt = parser.parse_args()
138
139     main(opt)

```

+ Python

```
[train epoch 4] loss: 0.010, acc: 0.996: 100% | 103/103 [00:29<00:00, 3.49it/s]
[valid epoch 4] loss: 0.001, acc: 1.000: 100% | 26/26 [00:22<00:00, 1.17it/s]
[train epoch 5] loss: 0.007, acc: 0.999: 100% | 103/103 [00:29<00:00, 3.53it/s]
[valid epoch 5] loss: 0.001, acc: 1.000: 100% | 26/26 [00:22<00:00, 1.17it/s]
[train epoch 6] loss: 0.002, acc: 0.999: 100% | 103/103 [00:29<00:00, 3.50it/s]
[valid epoch 6] loss: 0.001, acc: 1.000: 100% | 26/26 [00:22<00:00, 1.16it/s]
[train epoch 7] loss: 0.006, acc: 0.999: 100% | 103/103 [00:29<00:00, 3.51it/s]
[valid epoch 7] loss: 0.001, acc: 1.000: 100% | 26/26 [00:22<00:00, 1.14it/s]
[train epoch 8] loss: 0.012, acc: 0.995: 100% | 103/103 [00:29<00:00, 3.53it/s]
[valid epoch 8] loss: 0.001, acc: 1.000: 100% | 26/26 [00:22<00:00, 1.14it/s]
[train epoch 9] loss: 0.004, acc: 0.999: 100% | 103/103 [00:31<00:00, 3.27it/s]
[valid epoch 9] loss: 0.001, acc: 1.000: 100% | 26/26 [00:23<00:00, 1.13it/s]
PS C:\Users\86198\Desktop\vlt\code>
```



终端(T) 帮助(H) • 代码.ipynb - code - Visual Studio Code

train.py predict.py 代码.ipynb

实验环境: python=3.8、d2l=0.17.4、matplotlib=3.4.0、numpy=1.22.2、pandas=1.2.4、torch=1.10.2+cpu


代码 + Markdown 全部运行 清除所有输出 重启 变量 大纲

```
import PIL.Image

image_path = "C:/Users/86198/Desktop/2020113698顾宇航图片分类实验/pictures,
image = PIL.Image.open(image_path)
plt.imshow(image)
img = train_augs(image)
model = torch.load("Resnet.pth", map_location=torch.device('cpu'))
img = torch.reshape(img, (1, 3, 224, 224))
model.eval()
with torch.no_grad():
    output = model(img)
print(train_data_loader.dataset.classes[int(output.argmax(1))])
print(output)
```

[34] ✓ 1.2s

cup  
tensor([[ 0.0018, -0.0161, 0.0716, -0.0257, -0.0418, -0.0167, -0.0158]])



选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H) predict.py - code - Visual Studio Code

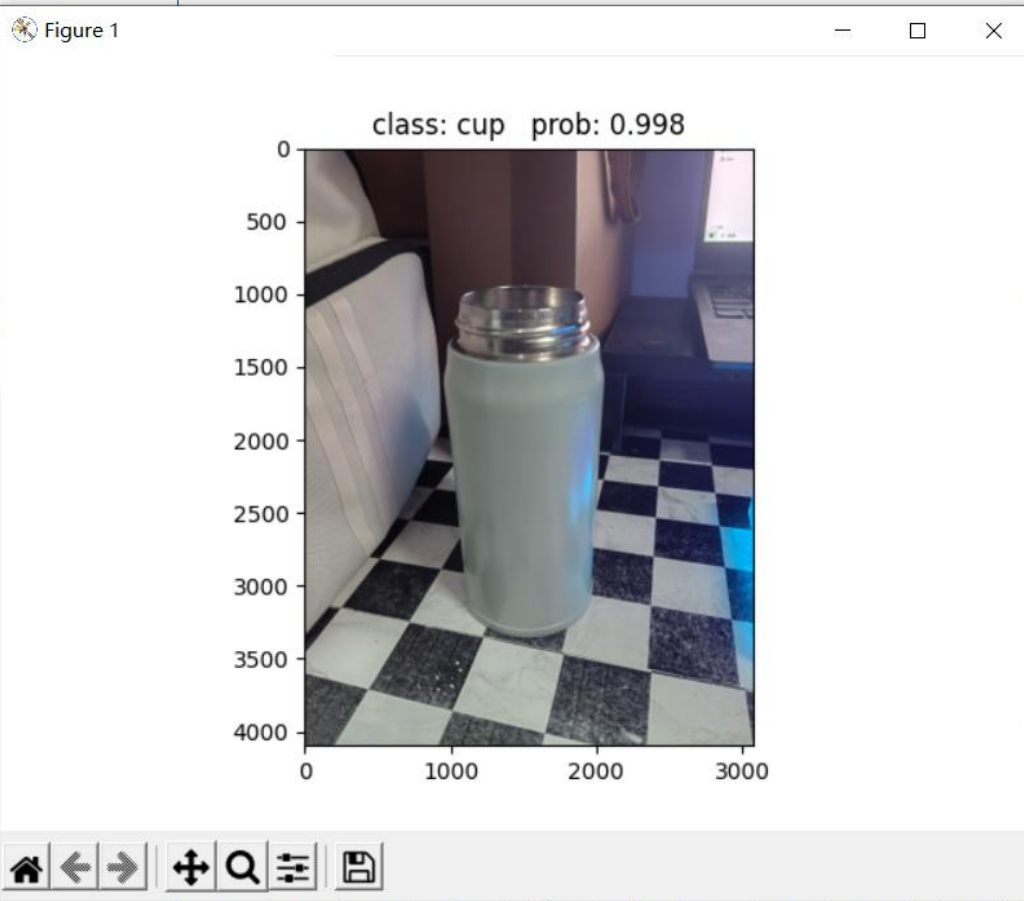
train.py predict.py 代码.ipynb

predict.py > main

Figure 1

otos  
ces.json  
et.py  
s.jpg  
atch16\_224.pth  
PY

class: cup prob: 0.998



PS C:\Users\86198\Desktop\vit\code> & C:/Users/86198/anaconda3/envs/torch/python.exe c:/U  
class: Faces prob: 0.00141  
class: airplanes prob: 0.000161  
class: cup prob: 0.998  
class: dolphin prob: 4.28e-06  
class: elephant prob: 5.49e-05  
class: ibis prob: 1.13e-06  
class: panda prob: 3.25e-05  
[]