AN IMAGE IS WORTH 16X16 WORDS:TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

摘要

虽然 Transformer 架构已经成为处理自然语言处理任务的首选模型，但其在计算机视觉领域的应用仍然有限。在视觉方面，注意力要么与卷积网络结合使用，要么用于替换卷积网络中的某些组件，同时保持其整体结构不变。本文表明，进行图像处理时可以不用依赖 CNN，纯 Transformer 可以直接应用于图像 patch 序列，并且在图像分类任务上表现非常好。当在大量数据上进行预训练，并转移到多个中等或小型的图像识别基准测试（ImageNet、CIFAR-100、VTAB 等）时，Vision Transformer（ViT）与最先进的卷积网络相比取得了优异的结果，同时训练所需的计算资源大大减少。本文复现了 ViT 模型的架构，通过执行花卉识别任务，把它与经典的 Resnet 网络进行对比，充分论证了 ViT 模型的优越性。

关键词： Transformer；ViT；图像分类

1 引言

基于自注意力机制的架构，特别是 Transformers [6]，已经成为自然语言处理 (NLP) 的首选模型，其主要是先在大型文本语料库上进行预训练，然后在较小的任务特定数据集上进行微调 [2]。由于 Transformers 的计算效率高、可扩展性强，它可以训练前所未有的模型，随着模型和数据集的增长，仍然没有出现性能饱和的迹象。

在计算机视觉领域中，有许多工作尝试将类似 CNN 的架构与自注意力机制相结合 [8]，而有些工作完全取代了卷积操作 [4]。将自注意力机制完全取代卷积的方法在理论上虽然是有效的，但由于使用了特定的的注意力模式，该方法还没有在现代硬件加速器上有效地扩展。因此，在大规模图像识别中，经典的类 resnet 架构仍然是最先进的。

受到 Transformer 在自然语言处理领域的成功启发，作者尝试将标准 Transformer 直接应用于图像，并尽量不做任何修改。作者将图像分割成多个 patch，再将这些 patch 的线性嵌入序列作为输入提供给 Transformer，也就是说每个 patch 被线性嵌入到某个向量空间中，然后这些嵌入的序列被用作 Transformer 的输入，在这一处理过程中，图像的这些 patch 被当作 NLP 应用中的 token 一样处理，并且作者使用监督学习的方式来训练这个模型进行图像分类。由于 Transformer 缺乏 CNN 固有的一些归纳偏置，例如平移等变性和局部性，当在 ImageNet 这样的中等规模数据集上进行训练而不使用正则化时，这些 Transformer 模型的准确率会比同等规模的 ResNets 低几个百分点，因此在数据不足的情况下泛化能力不佳。

然而，如果模型在更大的数据集上训练，情况会有所不同。作者发现大规模的训练优于归纳偏差。作者所提出的 Vision Transformer(ViT) 在大规模的数据集上进行预训练后，再在较小数据集上进行迁移学习后，能获得出色的表现。当在 ImageNet-21k 公共数据集或 JFT-300M 私有数据集上预训练时，ViT 在多个图像识别基准测试中达到甚至超过了当前最好的模型的水平。在 ImageNet 数据集上，最佳模型准确率达到了 88.55%，在 ImageNet-ReaL 数据集上，准确率为 90.72%，在 CIFAR-100 数据集上，准确率为 94.55%，在 VTAB(19 个任务) 套件上，准确率为 77.63%。

2 相关工作

2.1 VISION TRANSFORMER(ViT)

Transformer 的输入应该是一维标记嵌入序列，为了处理二维图像，我们将图像 $\mathbf{x}\epsilon\mathcal{R}^{H\times W\times C}$ 分割成一个个二维 patch，$\mathbf{x}_p\epsilon\mathcal{R}^{N\times\left(P^2\cdot C\right)}$，其中 (H,W) 是原始图像的分辨率，C 是通道数，(P,P) 是每个图像 patch 的分辨率，N=HW/P^2 是 patch 的数量，同时也是 Transformer 的有效输入序列的长度。将每一个 Patch 展开成一个一维向量，其大小为 D, 这就是 Patch 嵌入操作 (Eq. 1)。

与 BERT 的 [class]token 类似，作者将一个可学习的参数嵌入到图像的 patch 序列中 ($\mathbf{z}_0^0=\mathbf{x}_{class}$), 经过 Transformer Encoder 后, 输出到 $\mathbf{z}_L^0$，将 $\mathbf{z}_L^0$ 作为图像 y 的表示 (Eq. 4)。在预训练和微调期间，分类头都附加到 $\mathbf{z}_L^0$。分类头在预训练时由一个隐藏层的 MLP 实现，在微调时由一个线性层实现。

将 Position Embedding 添加到 Patch Embedding 中来保留图片的位置信息。对于 Position Embedding，作者做了一系列对比实验，在源码中默认使用的是 1D Pos.Emb.，对比不使用 Position Embedding 的结果，准确率提升了大概 3 个点，和 2D Pos.Emb. 比起来没太大差别，最后将得到的嵌入向量序列作为编码器的输入。

Transformer 编码器由多头自注意 (MSA) 和 MLP block(Eq 2, 3) 的交替层组成。在使用每个块之前要进行一次 Layer Norm 操作，并在每个块之间使用残差连接 [1] [7]。

$$\mathbf{z_0}=\left[\mathbf{x}_{class};\mathbf{x}_p^1\mathbf{E};\mathbf{x}_p^2\mathbf{E};...;\mathbf{x}_p^N\mathbf{E};\right]+\mathbf{E}_{pos},\mathbf{E}\epsilon\mathcal{R}^{\left(P^2\cdot C\right)\times D},\mathbf{E}_{pos}\epsilon\mathcal{R}^{(N+1)\times D} \tag{1}$$

$$\mathbf{z'}_l=MSA\left(LN\left(\mathbf{z}_{l-1}\right)\right)+\mathbf{z}_{l-1},l=1...L \tag{2}$$

$$\mathbf{z'}_l=MLP\left(LN\left(\mathbf{z'}_l\right)\right)+\mathbf{z'}_l,l=1...L \tag{3}$$

$$\mathbf{y}=LN(\mathbf{z}_L^0) \tag{4}$$

2.2 归纳偏置

相比 CNN 模型，ViT 模型具有较少的图像特定归纳偏置。在 CNN 中，局部性、二维邻域结构和平移等方差在每一层中都是内置的，而在 ViT 中，MLP 层具有局部性和平移等变性，自注意力层是全局的。二维邻域结构仅在模型开始时用于切割图像，并在微调阶段用于

调整不同分辨率图像的位置嵌入，除此之外，位置嵌入在初始化时不包含有关 Patch 二维位置的信息，所有 Patch 之间的空间关系都必须从数据中学习。

2.3 微调

通常，我们会对 ViT 进行大规模数据集的预训练，然后将其微调到较小的下游任务。为此，作者移除预训练的预测头并附加一个零初始化的 D×K 前馈层，其中 K 是下游类别的数量。通常在那些分辨率比预训练更高的数据集下进行微调可以提高模型的性能 [5] [3]，当输入更高分辨率的图像时，我们保持相同的 Patch 大小，从而会导致更长的有效序列长度。Vison Transformer 可以处理任意长度的序列（受内存限制），但是预训练的位置嵌入可能不再有意义。因此，作者根据原始图像中的位置对预训练的位置嵌入进行二维插值。

3 本文方法

3.1 本文方法概述

ViT 的整体结构如图 1所示。与寻常的分类网络类似，整个 Vision Transformer 可以分为两部分，一部分是特征提取部分，另一部分是分类部分。

在特征提取部分，VIT 所做的工作是提取特征，这一部分主要包括 Patch+Position Embedding 和 Transformer Encoder。Patch+Position Embedding 的作用主要是对输入进来的图片进行分块处理，每隔一定的区域大小划分图片块，然后将划分后的图片块组合成序列。在获得序列信息后，传入 Transformer Encoder 进行特征提取，Transformer Encoder 中有 Transformer 中特有的 Multi-head Self-attention 结构，通过自注意力机制来关注每个图片块的重要程度。

在分类部分，VIT 所做的工作是利用提取到的特征进行分类，在进行特征提取的时候，会在图片序列中添加上 [class]token，该 token 会作为一个单位的序列信息一起进行特征提取，提取的过程中，该 [class]token 会与其它的特征进行特征交互，融合其它图片序列的特征。最终，利用 Multi-head Self-attention 结构提取特征后的 [class]token 进行全连接分类。

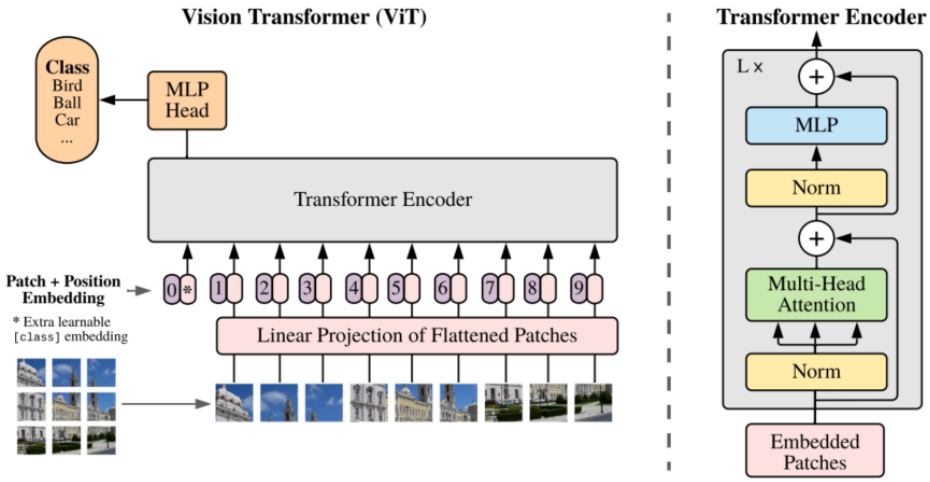

图 1. ViT 结构图

3.2 Position Embedding

位置编码（Position Encoding）是一种用于 Transformer 模型的编码系统，用于标识序列中每个单词或实体的位置。在自然语言处理中，Transformer 模型通过关注词与词之间的关系来工作，但仅知道词与词之间的关系是不够的，还需要知道每个词在序列中的位置信息。

位置编码为每个位置分配一个唯一的表示，最简单的方法是使用索引值来表示位置。然而，对于长序列，索引值会变得很大，这会导致很多问题。因此，位置编码将每个位置/索引都映射到一个向量。这样，位置编码层的输出是一个矩阵，其中矩阵中的每一行是序列中的编码字与其位置信息的和。

位置编码有多种演变方法，包括用整型值、[0,1] 范围内的值、二进制向量、正弦和余弦函数等来表示位置。在 Transformer 模型中，通常使用正弦和余弦函数来表示位置编码，因为这样可以确保无论序列长度如何，都可以被两者表示出来。

在 Vision Transformer（ViT）模型中，位置编码起着至关重要的作用。它可以帮助模型对图像中每个位置的特征信息进行建模，并且可以处理长距离依赖关系。通过使用位置编码，ViT 模型可以更好地理解图像，提高模型的表达能力和理解能力。

对于标准的 Transformer 模块，要求输入的是 token（向量）序列，即二维矩阵 [num_token, token_dim]，如图 1所示，token0-9 对应的都是向量，以 ViT-B/16 为例，每个 token 向量长度为 768。

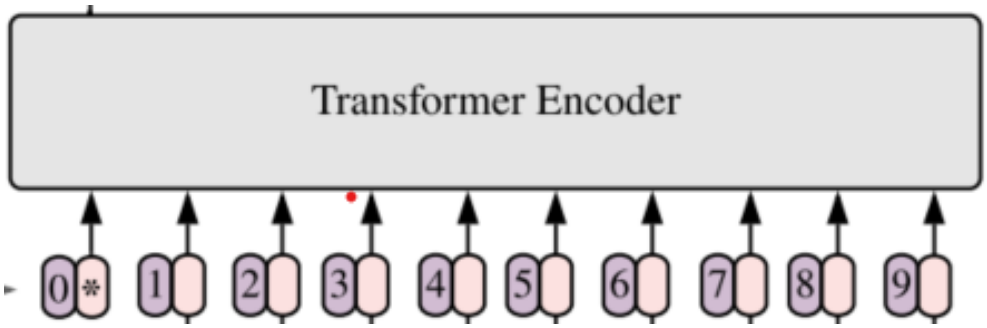

图 2. Transformer Encoder 输入

对于图像数据而言，其数据格式为 [H, W, C]，这是一个三维矩阵，不能直接将其输入到 Transformer 中，所以需要先通过一个 Embedding 层来对数据做个变换。如图 3所示，首先将一张图片按给定大小分成一堆 Patches。以 ViT-B/16 为例，将输入图片 (224x224) 按照 16x16 大小的 Patch 进行划分，划分后会得到 (224/16)2=196 个 Patches。接着通过线性映射将每个 Patch 映射到一维向量中，以 ViT-B/16 为例，每个 Patche 数据 shape 为 [16, 16, 3] 通过映射得到一个长度为 768 的向量（[16, 16, 3] -> [768]）。

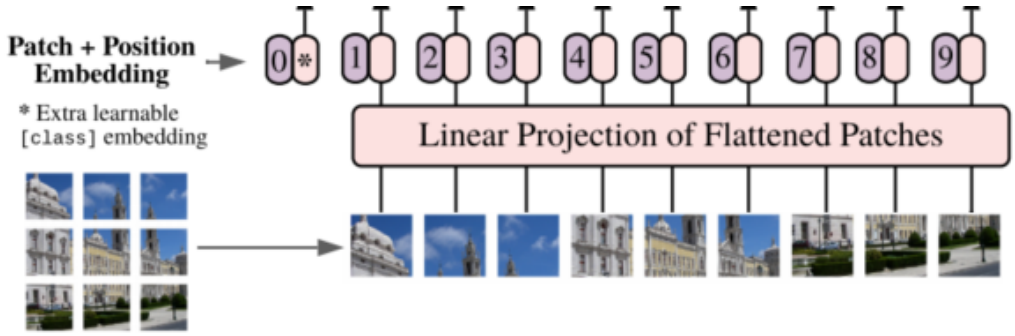

图 3. Patch+PositionEmbedding

在代码实现中，直接通过一个卷积层来实现。以 ViT-B/16 为例，直接使用一个卷积核大小为 16x16，步长为 16，卷积核个数为 768 的卷积来实现。通过卷积，维度从 [224, 224, 3] 变为 [14, 14, 768]，然后把 H 以及 W 两个维度展平即可，也就是将 [14, 14, 768] 变为 [196, 768]，此时正好变成了一个二维矩阵，可以输入到 Transformer 中。

在输入 Transformer Encoder 之前，还需要加上 [class]token 以及 Position Embedding。在原论文中，作者说参考 BERT，在刚刚得到的一堆 tokens 中插入一个专门用于分类的 [class]token，这个 [class]token 是一个可训练的参数，数据格式和其他 token 一样都是一个向量，以 ViT-B/16 为例，就是一个长度为 768 的向量，与之前从图片中生成的 tokens 拼接在一起，Cat([1, 768], [196, 768]) -> [197, 768]。关于 Position Embedding，就是 Transformer 中所涉及的 Positional Encoding，为所有特征添加上位置信息，网络才有区分不同区域的能力，这里的 Position Embedding 采用的是一个可训练的参数（1-D Pos.Emb.），是直接叠加在 tokens 上的（add），所以 shape 要一样。以 ViT-B/16 为例，刚刚拼接 [class]token 后 shape 是 [197, 768]，那么这里的 Position Embedding 的 shape 也是 [197, 768]。

3.3 注意力机制

从本质上理解，Attention 是从大量信息中有筛选出少量重要信息，并聚焦到这些重要信息上，忽略大多不重要的信息。权重越大越聚焦于其对应的 Value 值上，即权重代表了信息的重要性，而 Value 是其对应的信息。Attention 机制的具体计算过程可以归纳为两个过程：第一个过程是根据 Query 和 Key 计算权重系数，第二个过程根据权重系数对 Value 进行加权求和。而第一个过程又可以细分为两个阶段：第一个阶段根据 Query 和 Key 计算两者的相似性或者相关性；第二个阶段对第一阶段的原始分值进行归一化处理。

3.4 Transformer 编码器

Transformer Encoder 其实就是重复堆叠 Encoder Block L 次，主要由 Layer Norm、Muiti-Head Attention、Dropout/DropPath 以及 MLP Block 组成，如图 4所示。

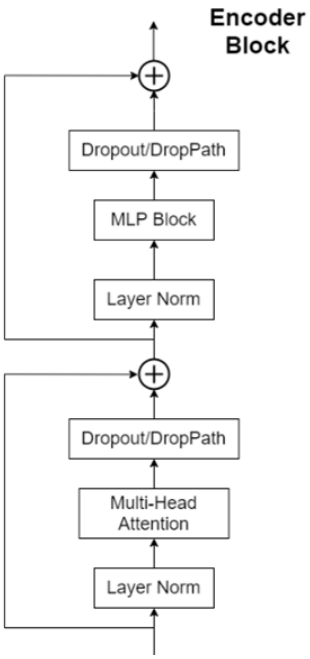

图 4. Encoder-Block

(1)Layer Norm。这种 Normalization 方法主要是针对 NLP 领域提出的，这里是对每个 token 进行 Norm 处理，可以帮助稳定训练过程并提高模型性能。

(2)Multi-Head Attention。Multi-Head Attention 是 Self-Attention 一种扩展，它将自注意力的操作并行运行多个 “头”，然后将它们的输出连接起来，可以有效地改善模型性能。

(3)Dropout/DropPath。在 ViT 模型的训练中，发现强正则化对于从头在 ImageNet 上训练模型是很关键的，可以用于增强模型泛化能力、防止过拟合。

(4)MLP Block。MLP Block 是由全连接层、GELU 激活函数以及 Dropout 构成的，其结构如图 5所示，需要注意的是第一个全连接层会把输入节点个数翻 4 倍，也就是 [197, 768] -> [197, 3072]，第二个全连接层会还原回原节点个数，也就是 [197, 3072] -> [197, 768]。

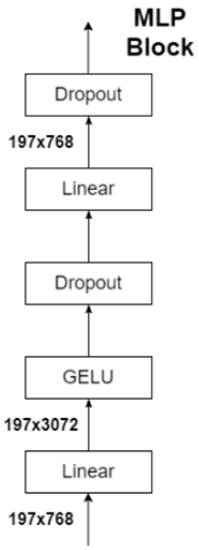

图 5. Encoder-Block

3.5 MLP Head

通过 Transformer Encoder 后输出的 shape 和输入的 shape 是保持不变的，以 ViT-B/16 为例，输入的是 [197, 768] 输出的还是 [197, 768]，从 Transformer Encoder 的输出结果中提取出 [class]token，即从 [197, 768] 中抽取出 [class]token 对应的 [1, 768]，最后将其送入到 MLP Head 得到我们最终的分类结果，如图 6所示。MLP Head 原论文中说在训练 ImageNet21K 时是由 Linear+tanh 激活函数 +Linear 组成，但是迁移到我自己所使用的数据上时，只用一个 Linear 就可以了。

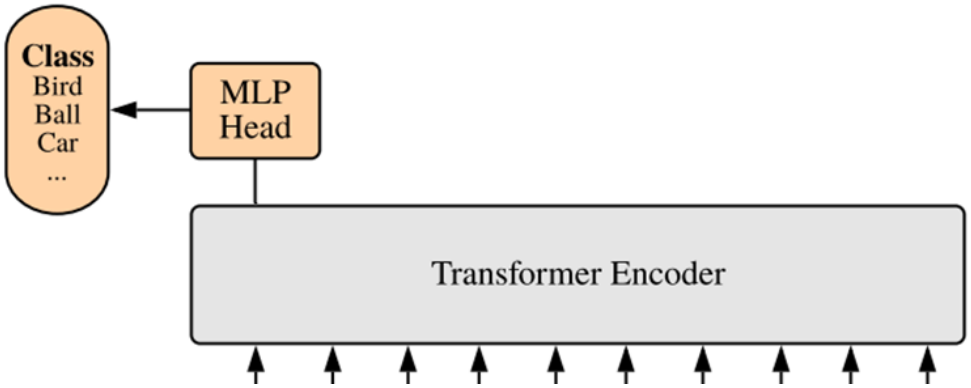

图 6. Encoder-Block

4 复现细节

4.1 与已有开源代码对比

在复现过程中，我并没有引用官方所给出的源代码，而是根据其思想用自己方法复现了出来，其中包括 Patch Embedding、Positional Embedding、自注意力机制等。使用的预训练权重是在网站 https://github.com/google-research/vision_transformer 下载的。

(1) 数据准备：从 tensorflow 的官方网站进行下载，其中的图片包括五种不同种类的花：雏菊、蒲公英、玫瑰、向日葵和郁金香，这些照片被存储在 flower_photo 文件夹的五个子文件夹中。

(2) 搭建 ViT 模型：根据 ViT 论文中的结构，使用 PyTorch 深度学习框架构建 ViT 模型。

(3) 模型训练：在训练过程中监控损失函数和准确率等指标，以评估模型性能。

(4) 模型调优：根据训练结果，对模型的超参数进行调整，如学习率、批量大小、Transformer 层数等。

(5) 模型评估：使用 f1_score 等指标评估模型性能，并与 ResNet 模型做了对比实验。

4.2 实验环境搭建

实验使用的 python 版本为 3.10，Pytorch 版本为 2.1.0。具体硬件信息如表 1所示。训练过程中使用 NVIDIA GeForce RTX 3090 单卡训练，ViT 模型训练时长为 2.5 个小时，ResNet 模型训练时长为 2 个小时。

GPU	NVIDIA GeForce RTX 3090
显存	24GB
内存	251.5GB

表 1. 硬件信息

4.3 创新点

根据论文的思想对 ViT 模型进行了复现，其中，在 MLP Head 中只使用了一个全连接层，并在 flower 数据集上进行微调，并与当前比较经典的 ResNet 模型做了对比分析。

5 实验结果分析

5.1 实验设置

在论文中给出了 ViT 三个模型的参数（Base/ Large/ Huge），本文复现的是 ViT-Base，详细信息如表 2所示，其中的 Layers 就是 Transformer Encoder 中重复堆叠 Encoder Block 的次数，Hidden Size 就是对应通过 Embedding 层后每个 token 的 dim（向量的长度），MLP size 是 Transformer Encoder 中 MLP Block 第一个全连接的节点个数（是 Hidden Size 的四倍），Heads 代表 Transformer 中 Multi-Head Attention 的 heads 数。

Model	Patch Size	Layers	Hidden Size D	MLP Size	Head	Params
ViT-Base	16*16	12	768	3072	12	86M

表 2. 模型参数

数据集使用的是 flower_picture 数据集, 这是一个专门用于花卉图像分类的数据集，它包含了五种常见的花卉类别，分别是雏菊、蒲公英、玫瑰、向日葵和郁金香。每种类别的图片数量分别为 633 张、898 张、641 张、699 张和 799 张，总共收集了 3670 张图片。在训练模型时，随机将数据集的 20% 作为验证集，剩下的数据集作为训练集。

训练参数如下表 3所示。

epochs	batch-size	lr	lrf
200	256	0.001	0.01

表 3. 训练参数

5.2 实验结果

ViT 训练的损失函数如下图 7所示。

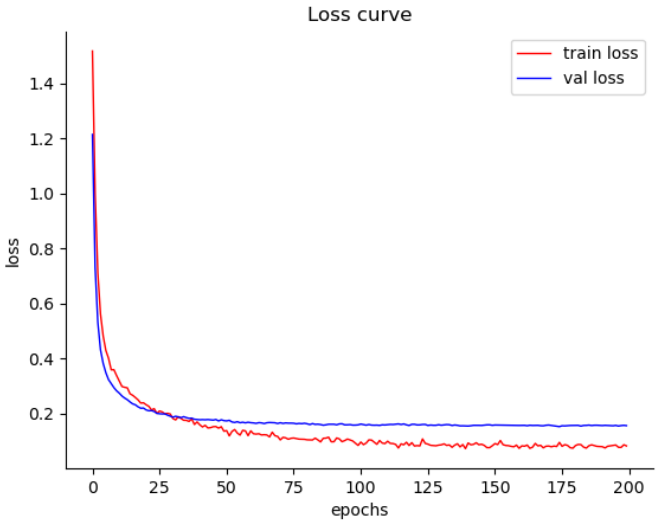

图 7. ViT 损失函数

ResNet 训练的损失函数如下图 8所示。模型大约训练 5 轮就收敛了。

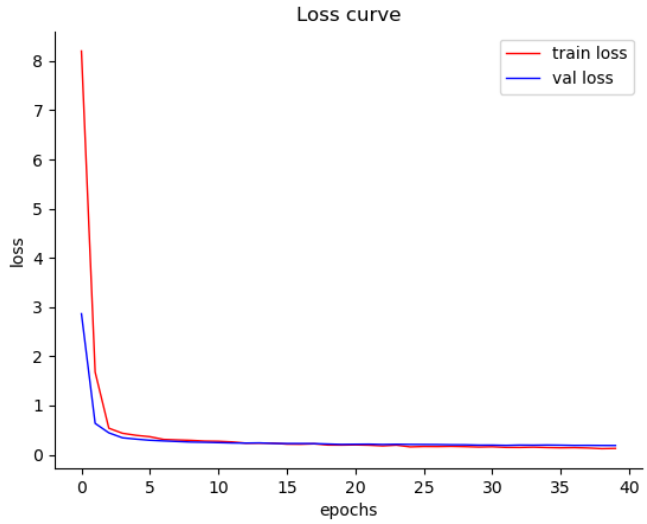

图 8. ResNet 损失函数

表 4为 ViT 的测试结果。从表中可以看出，该模型对 roses 类别的识别效果并不是很好，但是在识别其他种类的花时，其精度还是比较高的，这充分说明了预先在大型数据集上训练 ViT 模型，再在小型数据集上进行微调，可以提高模型的识别效果。后续考虑将 ViT 模型迁移到其他其他数据集上进行测试，并将 ViT 模型结合其他方向，比如目标检测、语义分割等。

表 5为 ResNet 的测试结果。从表中可以看出，和 ViT 模型相比，ResNet 识别的效果很不理想，猜测是在实验过程中学习率设置不当以及训练迭代次数不够等影响了模型的性能，对于每一种类别，其精度和召回率都特别低，在后续实验中，会针对这个问题展开详细的研究。

	precision	recall	f1-score
daisy	0.966	0.973	0.969
dandelion	0.963	0.976	0.969
roses	0.903	0.939	0.920
sunflowers	0.979	0.979	0.979
tulips	0.966	0.914	0.939
accuracy	0.956	0.956	0.956

表 4. ViT 模型实验结果

	precision	recall	f1-score
daisy	0.962	0.643	0.770
dandelion	0.493	0.988	0.657
roses	0.941	0.501	0.654
sunflowers	0.978	0.441	0.608
tulips	0.767	0.758	0.763
accuracy	0.689	0.689	0.689

表 5. ResNet 模型实验结果

6 总结与展望

ViT（Vision Transformer）模型是一种基于 Transformer 的视觉识别模型，它摒弃了传统的卷积神经网络（CNN）结构，将图像分割成多个小块，并将这些小块作为序列输入到 Transformer 中进行处理。这种方法使得模型能够捕捉到图像中的长距离依赖关系，从而提高了图像分类和目标检测等任务的性能。具体处理方式如下：在图像预处理方面，将图像分割成多个小块，使得 Transformer 能像在 NLP 应用中处理句子的 token 一样处理图片数据，并对每个小块进行位置编码，以便模型能够区分不同位置的像素，然后将预处理后的图像块序列输入到多层 Transformer 编码器中，每层编码器包含多个自注意力（Self-Attention）和前馈神经网络（Feed-Forward Neural Network）层，通过多层编码器的处理，模型能够逐渐提取图像的特征表示，最后将 Transformer 编码器的输出输入到一个全连接网络（Fully Connected Network），用于预测图像的类别标签。在复现过程中，我选择了 flower_photo 公开数据集，通过调节学习率等参数来避免训练过程中的梯度消失和爆炸问题，最后对模型进行性能的评估，并与经典的 ResNet 模型做了对比实验。

尽管 ViT 模型在图像识别任务上取得了显著的性能提升，但是目前的位置编码方法可能会导致一些信息的丢失，我认为可以尝试寻找更有效的位置编码方法来提高模型的性能，并且当前的 Transformer 结构在处理图像数据时可能会遇到计算和内存瓶颈等问题，我们可以通过设计更高效的 Transformer 结构来解决这个问题。由于 ViT 模型将整个图像划分为相同大小的小块进行处理，可能无法充分利用图像中的多尺度信息。可以尝试将不同尺度的特征融合在一起，以提高模型的性能。ViT 模型主要针对图像分类任务进行了优化，在之后的研究中，我们可以尝试将其应用于其他视觉任务，如目标检测、语义分割等，以验证其在更广

泛场景下的性能。

参考文献

[1] Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*, 2018.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[4] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. *Advances in neural information processing systems*, 32, 2019.

[5] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *Advances in neural information processing systems*, 32, 2019.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[7] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

[8] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.