Title：研究问题（细粒度分类问题）和方法

Abstract：

* 细粒度图像分类的定义和重要性（动机）
* 当前的研究背景和挑战
* （本文采用了什么方法和结构，新颖性，结果）本文做了什么工作、结果

Introduction：半张纸

* 介绍细粒度图像分类任务（还是动机、研究问题）
* 介绍传统的方法以及存在问题（CNN、ResNet、ViT等等其他方法）
* （自己的方法，好处）介绍本文要做的实验
* 简要概括实验结果

Related work：（细粒度图像分类的主要方法和进展）半张到一张纸

划分类型介绍

介绍不同类型的几篇论文（每篇都用一/二句话介绍观点，独特性优缺点）

（和自己的方法比较区别）

Method：

（

1. 描述方法或想法，解释为什么能解决问题
2. 正式描述模型或方法，用数学公式和易于理解的数字
3. 将模型分为几个部分一个个介绍模型架构
4. 解释为什么模型表现更好

）

* 1. 逐个说明工作过程：尝试迁移学习ResNet、ViT模型、通过Gan数据增广、（VLM）
  2. 描述方法（为什么能解决问题5%）、模型原理（数学公式、易于定位的数字5%），分部分解释模型架构（15%），解释方法的作用和优势（10%）

Experiment：

* 实验装置：数据集（数量，大小），预处理（resize，正则化）
* 训练技巧：优化器、损失函数、batch\_size、epoches、lr、训练策略、系数、正则化等
* 轻量化模型：(Optional 5%)
* 评价程序和指标：交叉验证、准确率、recall
* 方法有效性：在不同数据集上比较与其他方法的性能（使用图表）
* 通用性：在不同模型与其他方法比较
* 鲁棒性：方法对超参数不敏感
* 消融实验：删除某个组件显示下降性能
* 可视化：可解释性

Conclusion：

* 总结目的和想法
* （讨论方法的优缺点）总结工作过程各个方法的优缺点
* 根据实验结果，提出未来的研究和应用

2. \*\*ViT模型\*\*

ViT（Vision Transformer）模型通过将图像划分为固定大小的块（patch），并将这些块作为输入序列，利用Transformer模型的自注意力机制来捕捉图像中的细粒度特征。Transformers最开始用于自然语言处理领域，通过将图像分块和展平，映射到高维特征空间，使每个图像块与自然语言处理中的单词一样，同时加入位置编码，为每个图像块添加位置信息，保留了图像块的空间特征，使得自注意模型能对图像进行处理。这种方法通过在全局范围内建模序列数据的关系，克服了传统卷积神经网络和循环神经网络在捕捉长距离依赖时的局限性，能够有效处理图像分类任务中的局部和全局信息，很好地处理细粒度图像分类任务。

\*\*模型原理\*\*

ViT模型的通过将图像转换为一个序列，输入到Transformer模型进行处理，主要分为以下步骤：

- 图像划分为块：输入大小为(H, W, C)的图像x，将图像划分为固定大小(P X P)的块，一共分得N=(HW)/P2个块，每个图像块展平后维度为（P2，C）

- 块嵌入：将每个块展平并通过一个线性变换映射到一个高维空间，\[ \text{z}\_0^i = \text{Linear}(\text{Flatten}(x^i)) \]，其中 \( x^i \) 是第 \( i \) 个块，\(\text{z}\_0^i\) 是其对应的嵌入向量 ，如此便将三维图像转换成二维输入

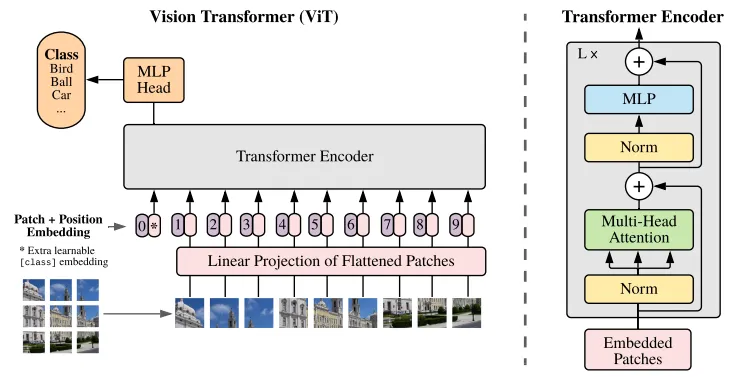
- 加入位置编码：由于Transformer模型对序列中的位置信息不敏感，需要加上位置编码 \( \text{E}\_{pos} \)：

\[ \text{z}\_0^i = \text{z}\_0^i + \text{E}\_{pos}^i \]

- 加入分类标记：在输入序列的开头加入一个特殊的分类标记 \( \text{z}\_0^0 = \text{class token} \)，这个标记具有独立的嵌入向量 \( \text{E}\_{\text{cls}} \) 和位置编码 \( \text{E}\_{\text{pos}}^{\text{cls}} \)，用于捕捉全局图像特征，经过Transformer编码器后用于分类任务。

- 通过Transformer：将块嵌入和位置编码输入到标准的Transformer编码器中，\[ \text{Z}\_l = \text{TransformerLayer}(\text{Z}\_{l-1}) \]其中 \( \text{Z}\_l \) 是第 \( l \) 层Transformer的输出。

- 通过分类头：最终分类结果通过一个线性层和softmax层得到：\[ \text{y} = \text{softmax}(\text{Linear}(\text{z}\_L^0)) \]



\*\*模型架构\*\*

图像预处理部分：将图像划分相同大小的块，并将块展平后形成块嵌入，输入到线性变换层

Linear Projection of Flattened Patches：通过线性变换将每个块嵌入到一个高维向量空间，形成输入序列

Position Embedding：将块的位置信息通过位置编码的形式加入输入序列，使Transformer能够利用图像的空间结构特征

Extra learnable [class] embedding：在序列开头加入分类标志，具有独立的嵌入向量和位置编码,用于分类任务

Transformer Encoder：其中经过多层EncoderBlock，每层EncoderBlock由两部分组成，第一部分是Norm层+多头注意力机制；第二部分是Norm层+MLP多层感知机，两个部分的前后都存在前馈神经网络连接

MLP Head：通过一个线性层将分类标记的输出映射到类别标签，并通过softmax层得到分类结果

\*\*模型的作用和优势\*\*

ViT通过自注意力机制有效捕捉图像中的局部和全局特征，在处理需要全局上下文的细粒度分类任务中非常有效，特别适用于细粒度图像分类任务。将图像作为序列处理，简化了图像中的位置信息处理。且Transformer架构具有高度的可扩展性，通过增加Transformer编码器层的数量、扩展每层的隐藏单元数以及增加注意力头的数量，可以构建更大的模型，且自注意力机制适合并行计算，Transformer中的每一层都可以并行处理所有输入块，从而大大提高了训练和推理的速度，使得ViT能够高效处理大规模数据和模型。

\*\*ViT与ResNets的区别\*\*

当训练数据集不够大的时候，ViT的表现相比ResNets要差，因为Transformer和CNN相比缺少归纳偏置（inductive bias），即一种先验知识，提前做好的假设。CNN通过卷积获取图像特征，具有两种归纳偏置，一种是局部性（locality/two-dimensional neighborhood structure），即图片上相邻的区域具有相似的特征；另一种是平移不变性（translation equivariance），f(g(x))=g(f(x))，其中g代表卷积操作，f代表平移操作。CNN具有这两种归纳偏置，需要相对少的数据就可以学习一个比较好的模型，而ViT的自注意机制需要大量的训练数据来获得可接受的视觉表示，从零开始学习小数据集的效果并不好，且由于内存限制，无法对整个ViT模型的参数进行训练，因此实验中都加载了预训练的ViT模型进行微调训练。

\*\*关于微调\*\*

ViT 模型的微调，能使模型适应更高的分辨率或不同的下游分类任务，具体的微调方法可以通过冻结参数并仅使用最后几层encoder的自注意层[1]以及最后的分类头进行训练。这样的冻结训练方式在对准确率影响低的同时，也降低了对GPU的内存占用。

当输入图片分辨率发生变化，输入序列的长度也发生变化，虽然ViT可以处理任意长度的序列，但是预训练好的位置编码无法再使用（例如原来是3x3，一种9个patch，每个patch的位置编码都是有明确意义的，如果patch数量变多，位置信息就会发生变化），一种做法是使用插值算法，扩大位置编码表。但是如果序列长度变化过大，插值操作会损失模型性能，这是ViT在微调时的一种局限性。

### Vision Transformer (ViT) Model

The Vision Transformer (ViT) model addresses fine-grained image classification by dividing an image into fixed-size patches and treating these patches as a sequence of inputs. By leveraging the self-attention mechanism of the Transformer model, initially developed for natural language processing (NLP), ViT captures intricate image features. The process involves flattening the patches and mapping them to a high-dimensional feature space, similar to how words are processed in NLP. Positional encoding is added to retain spatial information of the image patches, enabling the self-attention model to handle image data effectively. This method overcomes the limitations of traditional convolutional neural networks (CNNs) and recurrent neural networks (RNNs) in capturing long-range dependencies, allowing ViT to process both local and global information effectively, making it well-suited for fine-grained image classification tasks.

#### Model Principles

The ViT model transforms an image into a sequence and processes it using the Transformer model through the following steps:

- \*\*Image Partitioning\*\*: The input image \( x \in \mathbb{R}^{H \times W \times C} \) is divided into fixed-size patches \((P \times P)\). This results in \( N = \frac{HW}{P^2} \) patches, each of dimension \((P^2, C)\) after flattening.

- \*\*Patch Embedding\*\*: Each flattened patch is linearly transformed into a high-dimensional space:

\[ \text{z}\_0^i = \text{Linear}(\text{Flatten}(x^i)) \]

where \( x^i \) is the \(i\)-th patch, and \( \text{z}\_0^i \) is the corresponding embedding vector, thus converting the three-dimensional image into a two-dimensional input.

- \*\*Positional Encoding\*\*: Since the Transformer model is agnostic to the positional information of the sequence elements, positional encoding \( \text{E}\_{\text{pos}} \) is added to the patch embeddings:

\[ \text{z}\_0^i = \text{z}\_0^i + \text{E}\_{\text{pos}}^i \]

- \*\*Classification Token\*\*: A special classification token \( \text{z}\_0^0 = \text{class token} \) with an independent embedding vector \( \text{E}\_{\text{cls}} \) and positional encoding \( \text{E}\_{\text{pos}}^{\text{cls}} \) is added at the beginning of the input sequence to capture global image features, which is used for the classification task after being processed by the Transformer encoder.

- \*\*Transformer Encoding\*\*: The sequence of patch embeddings and positional encodings is passed through a standard Transformer encoder:

\[ \text{Z}\_l = \text{TransformerLayer}(\text{Z}\_{l-1}) \]

where \( \text{Z}\_l \) is the output of the \( l \)-th Transformer layer.

- \*\*Classification Head\*\*: The final classification result is obtained through a linear layer followed by a softmax layer:

\[ \text{y} = \text{softmax}(\text{Linear}(\text{z}\_L^0)) \]

[Insert Diagram of ViT Model Architecture]

#### Model Architecture

1. \*\*Image Preprocessing\*\*: The image is divided into equally sized patches, and each patch is flattened and embedded through a linear transformation.

2. \*\*Linear Projection of Flattened Patches\*\*: Each patch is linearly projected into a high-dimensional vector space, forming the input sequence.

3. \*\*Position Embedding\*\*: Positional information of the patches is added to the input sequence through positional encoding, enabling the Transformer to utilize the spatial structure of the image.

4. \*\*Extra Learnable [class] Embedding\*\*: A classification token with an independent embedding vector and positional encoding is added at the beginning of the sequence for the classification task.

5. \*\*Transformer Encoder\*\*: The input sequence passes through multiple encoder blocks. Each encoder block consists of two main components: a normalization layer with multi-head self-attention and a normalization layer with a multi-layer perceptron (MLP). Both components are connected with feed-forward neural networks.

6. \*\*MLP Head\*\*: A linear layer maps the output of the classification token to class labels, followed by a softmax layer to obtain the classification result.

#### Function and Advantages of the Model

The ViT model effectively captures both local and global features in images through its self-attention mechanism, making it particularly effective for tasks requiring fine-grained classification. Treating the image as a sequence simplifies the processing of positional information. Additionally, the Transformer architecture is highly scalable. By increasing the number of Transformer encoder layers, expanding the hidden units per layer, and adding more attention heads, larger models can be constructed. The self-attention mechanism also supports parallel computing, allowing each layer to process all input patches simultaneously, significantly enhancing training and inference speed. This enables ViT to efficiently handle large-scale data and models.

#### Comparison with ResNets

When the training dataset is not sufficiently large, the performance of ViT tends to be inferior to that of ResNets. This is due to the lack of inductive bias in Transformers compared to Convolutional Neural Networks (CNNs). Inductive bias refers to the prior assumptions built into the learning algorithm. CNNs extract image features through convolutions, benefiting from two primary inductive biases: locality (two-dimensional neighborhood structure), where adjacent regions in an image have similar features, and translation equivariance, represented as \( f(g(x)) = g(f(x)) \), where \( g \) denotes the convolution operation and \( f \) represents the translation operation. These biases enable CNNs to learn effective models with relatively small datasets. In contrast, the self-attention mechanism in ViT requires large amounts of training data to achieve acceptable visual representations. Starting from scratch with small datasets yields suboptimal results. Moreover, due to memory constraints, it is often impractical to train the entire ViT model's parameters from scratch. Therefore, in experiments, we utilized pre-trained ViT models and performed fine-tuning.

#### Fine-Tuning

Fine-tuning the ViT model allows it to adapt to higher resolutions or different downstream classification tasks. The specific fine-tuning method involves freezing most parameters and only training the last few self-attention layers of the encoder and the final classification head . This approach minimizes the impact on accuracy while reducing GPU memory consumption.

When the resolution of the input image changes, the length of the input sequence also changes. Although ViT can handle sequences of arbitrary lengths, the pre-trained positional encodings may no longer be applicable. For instance, if the original patch configuration was \(3 \times 3\) (yielding nine patches) with specific positional encodings, increasing the number of patches alters the positional information. One approach to address this is to use interpolation to resize the positional encoding table. However, significant changes in sequence length can degrade model performance due to interpolation inaccuracies, presenting a limitation for ViT during fine-tuning.