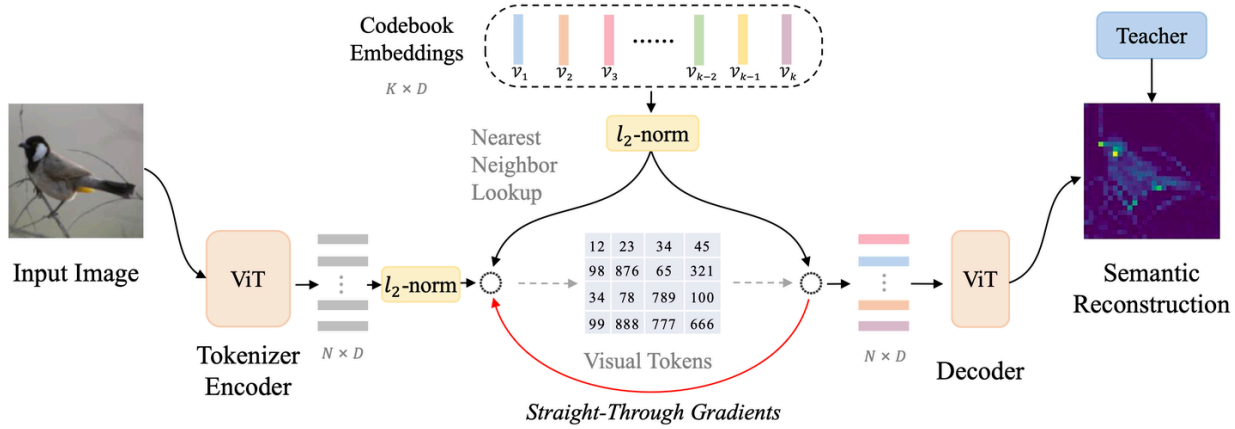
图像预训练： BEiT v2

目录

1. 先导知识
    1. 前言
    2. 1. 背景介绍
    3. 1.1 视觉码本
    4. 2. 算法详解
    5. 1.1 BEiT v2概述
    6. 1.2 VQ-KD
    7. 1.2.1 Tokenizer
    8. 1.2.2 Decoder
    9. 1.2.3 损失函数
    10. 1.2.4 视觉码本训练
    11. 1.3 BEiT v2预训练
    12. 1.3.1 掩码图像模型
    13. 1.3.2 [CLS]预训练
    14. 2. 总结
    15. Reference



先导知识

- BEiT
- BERT
- CLIP

前言

BEiT v1[2]提出的掩码图像模型（Masked Image Model，MIM）展示了它在图像预训练上的优秀效果。在BEiT v1的部分，我们讲到BEiT v1是一个两阶段的算法，它首先通过一个dVAE将图像映射成离散的视觉视觉标志（Visual Token），然后再通过视觉Transformer学习带掩码的图像Patch到视觉标志的映射。BEiT v1这么做的目的是将图像映射到一个离散的语义空间，然后模型通过学习每个掩码Patch到这个离散空间的映射来完成预训练。但是BEiT v1并未对dVAE学到的这个语义空间进行深入的探讨和优化，这也大大限制了BEiT v1的可解释性和使用空间。

BEiT v2[1]的提出正是为了解决这个问题的，它的核心思想是通过一个训练好的模型，例如CLIP[3]或是DINO[4]作为Teacher来指导视觉标志的学习，这个方法在BEiT v2中被叫做**矢量量化-知识蒸馏**（Vector-quantized Knowledge Distillation，VQ-KD）。BEiT v2的另一个创新点是引入了`[CLS]`标志符来学习整个图像的特征，从而使得BEiT v2在线性探测（Linear Probe）方式也能拥有非常高的准确率。

1. 背景介绍

1.1 视觉码本

视觉码本（Visual Codebook）又被叫做视觉字典（Visual Dictionary），它是一个传统计算机视觉概念，它一般被用于查找图像的视觉特征。例如在传统的目标识别中，我们一般会通过SIFT，Blob，图像梯度等算法来提取图像的高阶特征，这些特征一般被叫做视觉单词（Visual Words）。因此每个图像都可以表示为由视觉单词组成的集合。

视觉码本可以通过K-means等方法来构建，对于一个图像数据集得到的视觉标志，我们可以使用K-means将它们聚类成几个主要类别。这个类别数便是视觉码本的大小，每个类别的中心便是字典中视觉单词的具体内容。

在给定一张输入图像之后，我们可以使用传统方法将它编码成视觉特征的集合，然后我们通过查找特征在视觉码本的最近单词，便可以根据视觉码本得到这个图像的视觉表示。也就是说，通过这种方式，我们可以将任意一个张图像映射到视觉码本对应的表示空间中。

2. 算法详解

1.1 BEiT v2概述

BEiT v2和BEiT v1一样，它也是一个两阶段的模型：它的第一阶段是VQ-KD的训练，第二阶段是预训练模型的训练。针对我们上面分析的问题，为了提升BEiT v1的效果，BEiT v2做了如下改进。

- 提出了VQ-KD方法来对图像进行编码，它将原始图像作为输入，使用另外一个模型作为教师系统（Teacher）来引导视觉标志模型的训练。VQ-KD在这里重建的是教师系统编码的特征而非原始像素。
- 在通过VQ-KD得到图像的视觉标志之后，我们使用这个视觉标志作为预训练模型的训练目标。不同的是BEiT v2加入了`[CLS]`符号来建模图像的全局信息。

BEiT v2采取了和ViT[5]提出的图像Patch的表示方式，为了方便介绍后续其它算法，我们这里先给出图像表示的定义。给定一张彩色图像 $\boldsymbol{x} \in \mathbb{R}^{H \times W \times C}$，我们首先将它reshape成一个由图像patch组成的序列 $\{\boldsymbol{x}_i^p\}_{i=1}^N$，其中 $\boldsymbol{x}^p \in \mathbb{R}^{N \times (P^2 C)}$以及序列长度 $N = NW/P^2$，(P, P)是图像的尺寸。这些图像patch会被展开成一个序列，然后输入到视觉Transformer中并编码成 N个特征

向量： $\{\boldsymbol{h}_i\}_{i=1}^N$。在这一部分，BEiT v2和BEiT v1的参数保持一致，即输入图像的大小是 224×224，每个patch的大小是 16×16。

1.2 VQ-KD

正如我们在上面介绍的，VQ-KD的作用是将输入图像转化为视觉标志，即将输入图像 $\boldsymbol{x}$转化为视觉标志 $\boldsymbol{z}=[z_1, \cdots, z_N] \in \mathcal{V}^{(H/P)\times(W/P)}$， 其中 $\mathcal{V}$指的是视觉字典，或被叫做视觉码本。VQ-KD由两部分组成，分别是**Tokenizer** 以及 **Decoder** 。它的计算流程如图1所示。

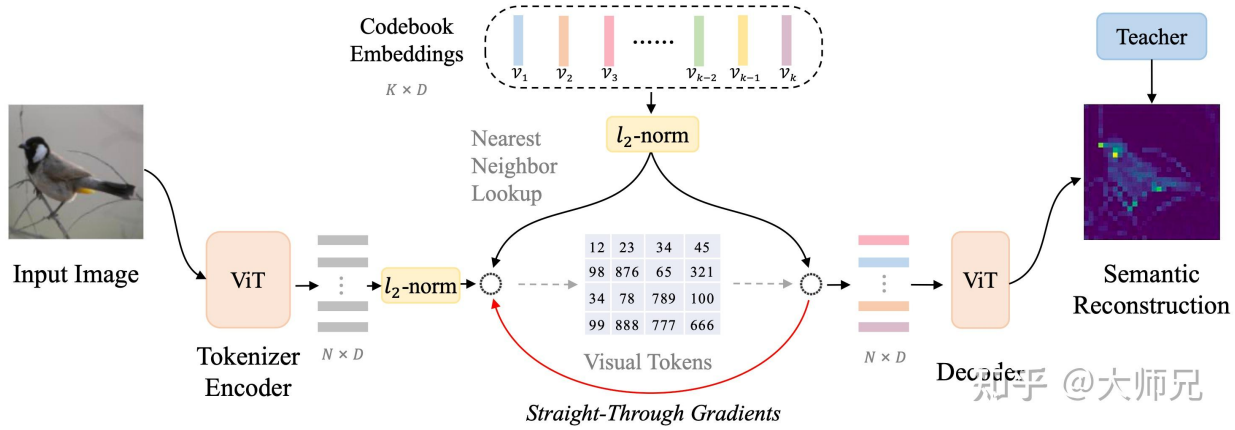


图1： VQ-KD的计算流程

1.2.1 Tokenizer

Tokenizer的计算分成两步：它首先使用ViT将输入图像编码成特征向量，然后使用从码本中查找最近的邻居。具体的讲，假设图像序列 $\{\boldsymbol{x}_i^p\}_{i=1}^N$编码成的序列表示为 $\{h_i^p\}_{i=1}^N$，码本的嵌入表示为 $\{\boldsymbol{e}_1, \cdots \boldsymbol{e}_{|\mathcal{V}|}\}$。那么第 i个图像Patch的特征 $\boldsymbol{h}_i$对应的视觉标志可以通过它和视觉码本中所有视觉单词的最小余弦距离来确定，如式(1)。

$$z_i = \arg\min_j \|\ell_2(\boldsymbol{h}_i) - \ell_2(\boldsymbol{e}_j)\|_2.$$

其中 ℓ_2是特征的L2正则化。

1.2.2 Decoder

对于一个由 N个Patch组成的序列，通过Tokenizer，我们可以得到由 N个视觉单词组成的序列，表示为 $\{\boldsymbol{z}_i\}_{i=1}^N$。通过将这些视觉单词进行正则化（L2正则），我们便可以将它输入到解码器（ViT）中，并到的 N个输出 $\{\boldsymbol{o}_i\}_{i=1}^N$。

1.2.3 损失函数

为了对Tokenizer和Decoder进行训练，VQ-KD的策略是使用模型蒸馏中提出的特征学习的策略。VQ-KD采用了CLIP[3]或是DINO[4]作为教师系统，然后以教师系统生成的特征作为输出的优化目标来进行模型的训练。具体的讲，我们用 $\boldsymbol{t}_i$表示教师系统在第 i个Patch上生成的特征，我们的目标表示最大化 $\boldsymbol{o}_i$和 $\boldsymbol{t}_i$的相似度（余弦距离）。

很多同学可能已经注意到了，式(1)的 $\arg\min$操作是不可导的，因此无法用传统的BP策略进行优化。为了将梯度传回到编码器中，VQ-KD采用了VQ-VAE [6]中提出的直接将梯度从接码器的输入复制到编码器的输出中（图1的红色箭头）。这里可以这么理解，因为我们通过比较和视觉码本中视觉单词的相似度的方式得到了输入图像的编码，因此对视觉单词的优化也可以近似看做对编码器编码的特征的优化。至少，它们的优化方向是一致的，因此可以用这种直接复制梯度的方式。

最终，VQ-KD的损失函数可以表示最大化模型输出以及教师系统生成的特征相似度并最小化生成特征和视觉单词的距离。因为存在不可导操作，所以损失函数的内容如式(2)。

$$\max \sum_{x\in\mathcal{D}} \sum_{i=1}^{N} \cos\left(\boldsymbol{o}_i, \boldsymbol{t}_i\right) - \left\|\mathrm{sg}\left[\ell_2\left(\boldsymbol{h}_i\right)\right] - \ell_2\left(\boldsymbol{e}_{z_i}\right)\right\|_2^2 - \left\|\ell_2\left(\boldsymbol{h}_i\right) - \mathrm{sg}\left[\ell_2\left(\boldsymbol{e}_{z_i}\right)\right]\right\|_2^2$$

其中 $\mathcal{D}$表示训练集， $\mathrm{sg}[\cdot]$表示停止梯度计算（stop-gradient）操作。

1.2.4 视觉码本训练

矢量量化训练的一个常见问题叫做码本衰减（Codebook Collapse），指的是码本中只有一少部分视觉单词被频繁使用，这样便大大降低了码本的使用效率。为了提高码本的使用率，VQ-KD在这里使用了下面几个策略：

- 使用了L2归一化进行码本查找，如式(1)；
- 将查找空间降低到了 32维，在特征被输入到解码器之前再映射回高维空间；
- 使用滑动平均来进行码本更新。

1.3 BEiT v2预训练

在通过VQ-KD得到图像的视觉标志之后，我们便可以以它为目标进行视觉Transformer的预训练了。为了学习图像的全局信息，BEiT v2在输入编码中拼接了`[CLS]`标志，然后通过对`[CLS]`标志的预训练来得到图像的全局信息。因此BEiT v2的预训练分成两个部分：分别是掩码图像模型的训练和`[CLS]`标志的训练，如图2所示。

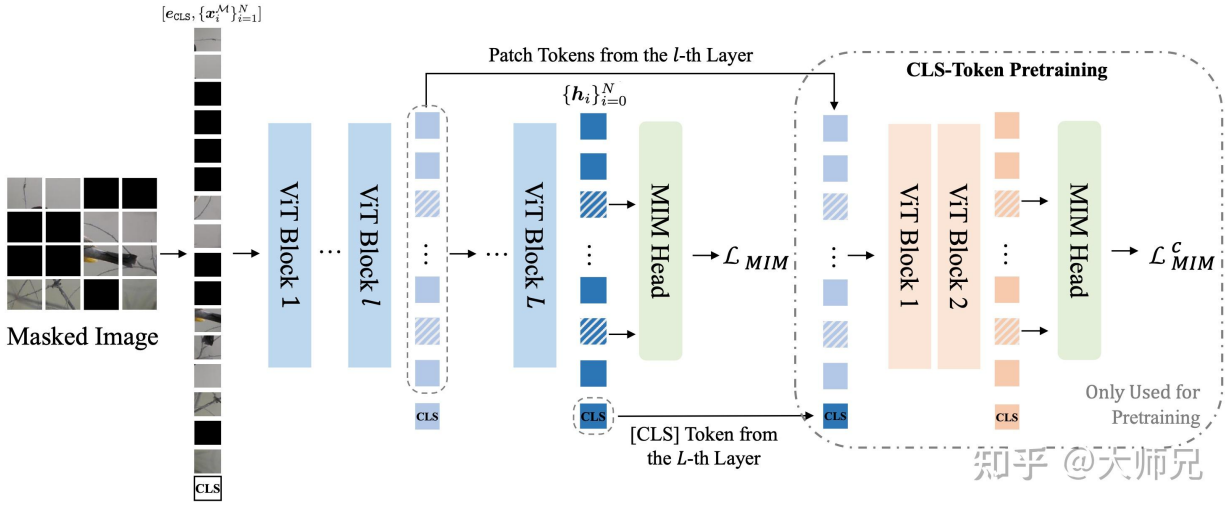


图2：掩码图像模型的计算流程

1.3.1 掩码图像模型

BEiT v2的预训练遵循了和BEiT v1类似的方式，不同的是它在输入数据中拼接了`[CLS]`标志，我们在这里会简单梳理一下MIM的基本流程和BEiT v2的调整，关于MIM的详细内容参考我的BEiT v1一文。

对于一张输入图像，我们首先使用块级别的掩码策略对输入图像的40%的图像Patch进行掩码，其中掩码的位置表示为 $\mathcal{M}$。接下来，我们使用共享的图像块嵌入 $\boldsymbol{e}_{[\mathrm{M}]}$ 替换掉被掩码的patch，得到 $\boldsymbol{x}_i^{\mathcal{M}}$，它的计算方式可以表示为式(3)，其中 δ 是指示函数。

$$\boldsymbol{x}_i^{\mathcal{M}} = \delta(i \in \mathcal{M}) \odot \boldsymbol{e}_{[\mathrm{M}]} + (1 - \delta(i \in \mathcal{M})) \odot \boldsymbol{x}_i^p.$$

在BEiT v2中，我们会向 $\boldsymbol{x}_i^{\mathcal{M}}$中加入一个`[cls]`符号共同输入到视觉Transformer中，因此BEiT v2的MIM的输入可以表示为 $[\boldsymbol{e}_{\mathrm{CLS}}, \{\boldsymbol{x}_i^{\mathcal{M}}\}_{i=1}^N]$。通过视觉Transformer的计算，我们可以得到模型的输出 $\{\boldsymbol{h}_i\}_{i=0}^N$，其中 $\boldsymbol{h}_0$是`[CLS]`标志对应的特征。

最后，我们会在视觉Transformer之后添加一个MIM的输出头，用于预测图像patch对应的视觉标志。也就是对于每个 $\{h_i : i \in \mathcal{M}\}_{i=1}^N$，使用softmax损失函数预测每个patch的输出概率，表示为式(4)。

$$p\left(z' \mid x^{\mathcal{M}}\right) = \mathrm{softmax}_{z'}\left(\boldsymbol{W}_c \boldsymbol{h}_i + \boldsymbol{b}_c\right).$$

其中 z'是模型预测的视觉标志。 $\boldsymbol{W}$和 $\boldsymbol{b}$分别是权值矩阵和偏置向量。所以，最终MIM的损失函数可以表示为式(5)。其中 z_i是我们在1.2节介绍的通过VQ-KD得到的视觉标志。

$$\mathcal{L}_{\mathrm{MIM}} = -\sum_{x \in \mathcal{D}} \sum_{i \in \mathcal{M}} \log p\left(z_i \mid x^{\mathcal{M}}\right)$$

1.3.2 [CLS]预训练

为了捕获图像的全局信息，我们需要对`[CLS]`标志进行训练。`[CLS]`的预训练如图2的右半部分的虚线框所示，它的输入是由第 l层视觉Transformer的特征向量和第 L层的`[CLS]`的特征向量拼接而成，表示为 $\boldsymbol{S} = [h_{\mathrm{CLS}}^L, h_1^l, \cdots, h_N^l]$。接下来我们将特征 $\boldsymbol{S}$输入到一个两层的Transformer中来预测掩码图像的视觉标志。并且`[CLS]`的预训练的输出头是和原始MIM的输出头的参数是共享的。因此`[CLS]`的任务有两个，分别是原始的第L层接的MIM任务和由浅层Transformer计算的MIM任务。

为什么说通过对`[CLS]`的预训练，我们得到的特征 h_{CLS}^L具有图像的全局信息呢?因为在训练`[CLS]`时，我们舍弃了图2中最左侧的第 $l+1$层到第 L层的图像Patch的信息，而只将第 L层的 h_{CLS}^L的值传递到了`[CLS]`的预训练的过程中。而我们却要求两个不同的MIM任务共享同一个MIM输出头，这就迫使 h_{CLS}^L学习更多的全局信息，以弥补被舍弃的图像所有标志的特征的信息。

最后我们说明一下，网络的两个浅层Transformer只会用在`[CLS]`预训练中，当预训练完成之后，这一部分便会被舍弃。

2. 总结

BEiT v2最核心的贡献是使用了VQ-KD作为视觉标志的生成结构，对比BEiT v1的dVAE，BEiT v2使用教师系统来引导视觉标志的生成，因为作为教师系统的CLIP或是DINO本身就是非常出色的预训练模型，因此它们携带的信息要比原始像素携带的信息量更加具体和具有代表性。在训练MIM时，BEiT v2借鉴了Condenser[8]的思想引入了`[CLS]`标志来学习图像的全局信息，`[CLS]`预训练的方式还是非常有创意和特点的。

BEiT v2是该系列最后一篇纯图像预训练模型，随着多模态的火热，BEiT v3也将开启它的多模态之路，我们下一篇也将带来BEiT v3的详细介绍。

Reference

[1] Peng, Zhiliang, et al. "BEiT v2: Masked Image Modeling with Vector-Quantized Visual Tokenizers." arXiv preprint arXiv:2208.06366 (2022).

[2] Bao, Hangbo, Li Dong, and Furu Wei. "Beit: Bert pre-training of image transformers." arXiv preprint arXiv:2106.08254 (2021).

[3] Radford, Alec, et al. "Learning transferable visual models from natural language supervision." International Conference on Machine Learning. PMLR, 2021.

[4] Caron, Mathilde, et al. "Emerging properties in self-supervised vision transformers." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

[5] Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).

[6] Van Den Oord, Aaron, and Oriol Vinyals. "Neural discrete representation learning." Advances in neural information processing systems 30 (2017).

[7] Gao, Luyu, and Jamie Callan. "Condenser: a pre-training architecture for dense retrieval."*arXiv preprint arXiv:2104.08253*(2021).