TransFG 论文复现

1 简介

我们基于计图深度学习框架 [1]，复现了论文 TransFG[2] 中的模型。

TransFG 在 VIT[3] 的基础上，引入了部分选择模块以及对比损失函数来提高模型在细粒度图像识别上的表现。我们使用计图实现的模型与使用 PyTorch 实现的模型在测试中达到了与 torch 相近的准确率和训练速度。此外，我们还在多种细粒度图像识别数据集上测试了计图实现的 TransFG 模型，并做了充分的消融实验以证实各个模块的必要性。我们还制作了一个网页 Demo，可以识别用户上传的图像，并用红色方框显示出 Part Select Module 挑选出的图像块，图形化地展现了论文在可解释性方面的贡献。

我们工作的创新性包含以下几个方面：

- 基于计图框架和 torch 框架复现了 TransFG。
- 在多个细粒度图像识别测试集上做了测试。
- 对论文所有提出的方法做了消融实验。
- 制作了网页版 Demo，图形化展现模块输出。
- 实现了适配 jittor 的余弦学习率衰减。

项目链接：https://github.com/whycreeper/TransFG_reproduction。

2 复现论文介绍

TransFG 在 VIT[3] 的基础上，引入了部分选择模块 (Part Select Module) 以及对比损失函数 (Contrastive Loss) 来提高模型在细粒度图像识别上的表现。

Part Select Module (PSM) 将 Transformer 的所有原始注意力权重整合为一个注意力图，用于引导网络有效且准确地选择判别性图像块并计算它们之间的关系。

$$a_{final} = \prod_{l=0}^{L-1} a_l$$

然后，选择 a_{final} 中 K 个不同注意力头对应的最大值 $A_1, A_2, \ldots, A_K$ 的索引位置。这些位置被用作索引，供我们的模型从 z_{L-1} 中提取相应的 token。最后，我们用对应于信息区域的 token 替换原始的整个输入序列，并将分类 token 连接作为最后一层 Transformer 的输入。

PSM 模块使得模型可以更加关注到图像中的某些细节，模型对不同鸟类的鸟喙、毛色、眼睛有了更加深入的分析识别。这使得模型在细粒度图像识别任务上的表现有所提升。

Contrastive Loss 与交叉熵损失函数的结合进一步提高了模型的学习能力。

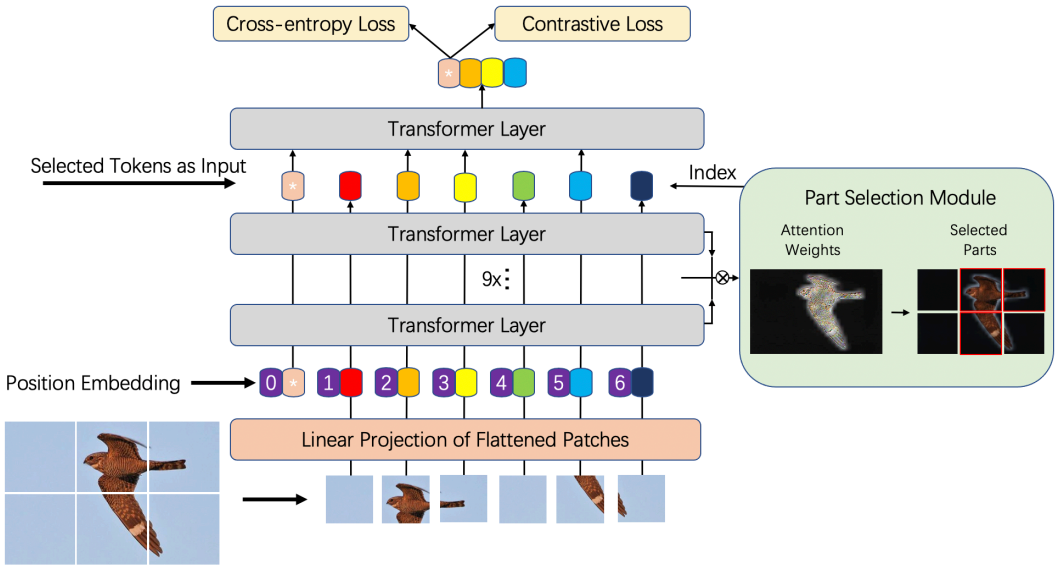

Figure 1: TransFG 的框架。以 VIT 的架构为基础，在最后一个 Transformer Layer 之前，增加部件选择模块（PSM），挑选出对图像识别最为重要的图像块输入到最后一层。

$$\mathcal{L}_{con} = \frac{1}{B^2}\sum_{i}^{B}[\sum_{j:y_i=y_j}^{B}(1 - Sim(z_i, z_j) + \sum_{j:y_i\neq y_j}^{B}\max((Sim(z_i, z_j) - \alpha), 0)]$$

$$\mathcal{L} = \mathcal{L}_{cross}(y, y') + \mathcal{L}_{con}(z)$$

Overlap Overlap 是论文提出的一个分割图片的方法。原始的分割方法是将图像切割成不重叠的块，这样会破坏局部的邻接结构，特别是当区分性区域被切割时。为了解决这个问题，论文提出了使用滑动窗口生成重叠的图像块。具体来说，假设输入图像的分辨率为 H * W，图像块的大小为 P，滑动窗口的步长为 S。因此，输入图像将被分割成 N 个图像块。其中：

$$N = N_H * N_W = \lfloor\frac{H - P + S}{S}\rfloor * \lfloor\frac{W - P + S}{S}\rfloor$$

在此之上，论文使用消融实验证明了上述三个模块的有效性，并确定了计算 Contrastive loss 时最佳的 alpha 值为 0.4。

原深度学习框架是 torch，用 jittor 复现的难点在于原框架使用的诸多库只和 torch 相适配，而我们需要将这些功能重新实现或者选择用等效的方法代替；另一难点在于 jittor 框架无法使用原本的 batchsize，否则会爆显存，因此我们一定程度上做了一些超参数设置的摸索。

3 实验

3.1 实验设置

3.1.1 数据集

我们所使用的数据集均为图像分类的数据集。CUB-200-2011、Stanford Cars、Stanford Dogs 和 NABirds 是几个常用的视觉识别数据集，特别用于物体检测、图像分类和物种识别等任务。它们包含多种不同种类的鸟类、汽车、狗和其他动物的图像，适用于训练和评估计算机视觉模型。链接如下：

- CUB-200-2011[1]
- Stanford Cars[2]另一部分[3]
- Stanford Dogs[4]
- NABirds[5]

3.1.2 超参数

由于 torch 的显存利用更加高效，使用 jittor 框架时，我们需要调整批量大小防止 GPU 显存不足。需要对于图像切割采用 overlap 的实验，我们设置 batchsize = 4；对于 Non-Overlap 的实验，设置 batchsize = 8。如下是经过测试后，对于 jittor 框架的最佳超参数设置：

Accuracy	**steps**	**lr**	**overlap**
CUB-200-2011	10k	0.03	✓
Stanford Cars	40k	0.03	✓
Stanford Dogs	20k	0.003	
NABirds	90.01	0.03	✓

其他设置与原论文相同。

3.2 实验结果

3.2.1 对实验结果的说明

在实验过程中，我们发现原论文的实验结果数据是不可复现的。原作者对于 cub 数据集给出了他的实验设置，而我们即使使用 torch 框架（也就是作者的一字不改的代码），完全按照作者的参数实验，在 cub 上依旧无法达到论文中的准确率。而对于其他的数据集，作者没有明确给出训练参数；论文中的 Experiments Setup 一节并没有完全写清楚实验设置，对于各个数据集的不同设置仅有一句话“The learning rate is initialized as 0.03 except 0.003 for Stanford Dogs dataset and 0.01 for iNat2017 dataset”。这显然是远远不够的，因为我们在改变学习率而不改变其他设置的情况下，用 torch 框架复现了其他数据集，仍旧达不到论文的指标。我们怀疑论文的数据是否真实。

不仅如此，我们在 transfg 的 git 仓库的 issue 中发现了许多研究者遇到了同样的问题，我在此一一列举：

- https://github.com/TACJu/TransFG/issues/30：这位研究者复现 cub 数据集得到 91% 准确率
- https://github.com/TACJu/TransFG/issues/24：其中一位研究者复现 dog 数据集得到 90.5% 准确率
- https://github.com/TACJu/TransFG/issues/14：这位研究者复现 cub 数据集得到 91.1% 准确率
- https://github.com/TACJu/TransFG/issues/13：其中两位研究者复现 car 数据得到 90% 和 90.86% 准确率

[1]https://data.caltech.edu/records/65de6-vp158

[2]https://www.kaggle.com/datasets/jessicali9530/stanford-cars-dataset/discussion?sort=hotness

[3]https://www.kaggle.com/datasets/abdelrahmant11/standford-cars-dataset-meta?resource=download

[4]https://www.kaggle.com/datasets/jessicali9530/stanford-dogs-dataset

[5]https://www.dropbox.com/scl/fi/yas70u9uzkeyzrmrfwcru/nabirds.tar.gz?___hsfp=3208242174___hssc=75100365.2.17340
f2bb-4009-8a18-6916b361a650

以上 issue 中所有研究者得到的数据均低于论文中的数据，而他们的数据与我们的数据相类似，这进一步加强了我们对原文的质疑。因此，我们在后续的实验结果讨论中，更倾向于将 jittor 框架的结果与我们自己的 torch 框架的结果相比较，但是我们仍然会列举出原论文的数据，并将 VIT，也就是原论文的 baseline 数据列举出来。

3.2.2 基本实验

本节对应实验要求的基本部分与 option1 部分。

我们分别使用 jittor 和 torch 框架，在 CUB-200-2011, Stanford Cars, Stanford Dogs, NABirds 上分别训练模型，成功使 jittor 框架与 torch 框架达到同一标准线。下图的表格展现了训练结果，“jittor”一列为我们的复现模型的测试结果，“torch”一列为运行官方仓库得到的复现结果，最后一列为论文中给出的结果。

Accuracy	**Jittor**	**Torch**	**TransFG(in paper)**	**VIT(in paper)**
CUB-200-2011	91.15	91.18	91.7	90.3
Stanford Cars	90.24	90.21	94.8	93.7
Stanford Dogs	90.93	90.63	92.3	91.7
NABirds	90.01	90.74	90.8	89.9

3.2.3 消融实验

我们分别对 PSM、Contrastive Loss 两个论文对 VIT 的改进方法进行了消融实验，并对图像切分方式和 α 的选择进行了对比实验。后两个表格中的实验使用了 Non-Overlap 的分割方式以节省训练时间，每组实验都采取 steps $= 10000$ 的设置进行。

Method	**Patch Split**	**Accuracy (%)**	**Training Time (h)**
ViT	Non-Overlap	90.54	0.77
ViT	Overlap	90.28	1.38
TransFG	Non-Overlap	90.89	0.82
TransFG	Overlap	**91.15**	2.17

Method	**Contrastive Loss**	**Acc (%)**
ViT		90.49
ViT	✓	90.54
TransFG		90.78
TransFG	✓	**90.89**

Method	**Value of α**	**Accuracy (%)**
TransFG	0	90.64
TransFG	0.2	90.58
TransFG	0.4	**90.88**
TransFG	0.6	90.59

消融实验结果显示：

- PSM 和对比损失函数的引入提高了模型在细粒度图像识别任务上的能力。
- Overlap 的图像切分方式表现更好，但训练时间更长。
- $\alpha = 0.4$ 的选择使模型表现最佳。

以上结论均与原论文相符。

3.3 额外工作

3.3.1 demo

如图2所示，我们实现了一个网页版 Demo。这一 Demo 可以将模型 Part Select Module 中选择的细粒度区分关键信息用方框标注，这是这篇论文的核心思想。

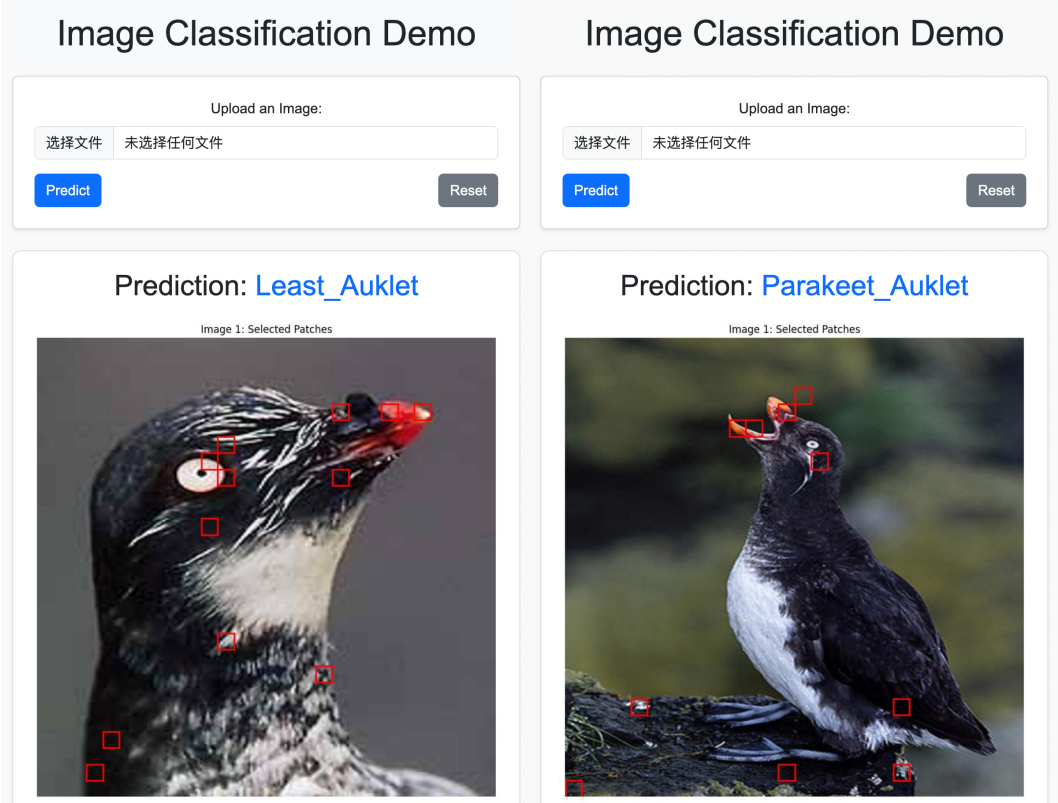

Figure 2: 网页版 Demo。展现图像识别结果，红色方框为 PSM 挑选出的送入最后一层分类头的图像块，展现了本工作在图像识别可解释性上的贡献。

3.3.2 余弦学习率衰减

我们实现了 jittor 没有的模块，余弦学习率衰减。具体代码在 utils/scheduler.py 中。这一实现的主要优势在于使用极为简单，只需在训练之前初始化，并在训练的过程中，在每次梯度下降后调用 step 函数即可。这一函数不依赖于 torch 库，可以完美适配 jittor 框架。

4 总结

本研究通过使用 Jittor 框架成功复现了 TransFG 模型，并在多个数据集上进行了全面评测。我们主要的贡献包括：

消融实验：通过消融实验评估了 TransFG 模型中各个模块的效果，深入分析了不同组件对模型性能的贡献。网页版 demo：实现了一个互动式的网页演示，便于实时体验模型的功能和性能。余弦学习率衰减的适配：在 Jittor 框架中实现了余弦学习率衰减，优化了训练过程，提高了模型的稳定性和收敛速度。

通过本研究，我们发现模型中某些模块在不同数据集上的表现差异，尤其是在细粒度分类任务中，一些局部特征的提取和模块的协同工作尤为重要。未来的研究可以聚焦于进一步优化模型结构，探索更多的数据增强技术，以及在更大规模数据集上的测试。

参考文献

References

[1] Shih-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Science China Information Sciences*, 63, 2020.

[2] Ju He, Jieneng Chen, Shuai Liu, Adam Kortylewski, Cheng Yang, Yutong Bai, Changhu Wang, and Alan Loddon Yuille. Transfg: A transformer architecture for fine-grained recognition. In *AAAI Conference on Artificial Intelligence*, 2021.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020.