Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

摘要

Swin Transformer 提出了一种层次化 Transformer，其表示是用移动窗口计算的。移位窗口方案通过将自注意力计算限制在不重叠的局部窗口的同时，还允许跨窗口连接来提高效率。这种分层架构具有在各种尺度上建模的灵活性，并且相对于图像大小具有线性计算复杂度。Swin Transformer 的这些特性使其与广泛的视觉任务兼容，包括图像分类（ImageNet-1K 的 87.3 top-1 Acc）和密集预测任务。证明了基于 Transformer 的模型作为视觉主干的潜力。本工作在原有的网络模型基础上使用了标签平滑损失函数，在 mammal 的数据集上 top-1 Acc 达到了 94.2 超过了 timm 库中的 SoftTargetCrossEntropy 的 92.4 以及传统交叉熵方法的 90.7。

关键词： Swin Transformer；图像分类；计算机视觉

1 引言

在 CV 领域，卷积神经网络（CNN）一直占据主导地位，从最早的 AlexNet [4] 到后来更为复杂的架构，如图像分类和目标检测中的 ResNet [3] 和 YOLO [2]，CNN 在各种视觉任务中展现出强大的性能。然而，在 NLP 领域，Transformer 架构崭露头角，特别是通过其在处理序列建模和转换任务中的卓越性能而引起关注。然而，CV 和 NLP 之间存在着一些显著的差异，这导致了在 Transformer 应用于 CV 时面临的一些挑战。其中一个关键差异是尺度的问题，因为视觉元素的尺度可以存在很大差异，这在目标检测等任务中变得尤为明显。另一个挑战是图像中的像素分辨率远高于文本段落中的文字，这在需要进行像素级别密集预测的任务中变得更为复杂。为了克服这些问题，本文介绍了一种新的 Transformer 架构，即 Swin Transformer。与现有基于 Transformer 的模型不同，Swin Transformer [5] 构建了层次化特征图，使其具有线性计算复杂度，从而更好地适应 CV 任务的特性。此外，Swin Transformer 引入了连续自注意力层之间的窗口分区的移位策略，显著增强了建模能力。本工作在学习了标签平滑技术技术之后，自己实现了损失函数，试图提高模型的性能。

2 相关工作

2.1 Transformer 架构

Transformer [6] 架构是一种在自然语言处理领域取得巨大成功的模型。最初由 Vaswani 等人提出，Transformer 模型通过引入自注意力机制实现了对序列数据的高效建模。这一框架在 NLP 任务中的广泛应用促使研究者将其引入计算机视觉领域，为后续的视觉任务模型提供了新的思路。

2.2 视觉 Transformer（ViT）

视觉 Transformer（ViT） [1] 是一种将 Transformer [6] 模型成功应用于图像分类任务的方法。Dosovitskiy 等人提出了一种将图像划分为小块，然后将这些块转换为序列输入的方法，通过这种方式，ViT 模型成功地将 Transformer 引入了视觉任务。ViT 在许多图像分类竞赛中取得了显著的成绩，证明了 Transformer 在处理视觉数据方面的潜力。

2.3 自注意力机制

在 Transformer 模型中，自注意力机制（Attention Mechanism）起到了关键作用。相关工作着重于对注意力机制的改进和优化，以适应不同任务和数据模式。这方面的研究有助于提高模型对图像或序列中不同部分的关注度，进一步改进了模型的表现。

3 本文方法

3.1 本文方法概述

Swin Transformer [5] 采用了层次化的架构，总共包含 4 个阶段（Stages），每个阶段都会逐渐缩小输入特征图的分辨率，类似于卷积神经网络（CNN）中逐层扩大感受野的过程。每个阶段由 Patch Merging 和多个块（Blocks）组成。Patch Merging 模块主要在每个阶段的开始处降低图像分辨率。而块主要包括 LayerNorm、MLP（多层感知器）、Window Attention 和 Shifted Window Attention。主要流程如图 1所示：

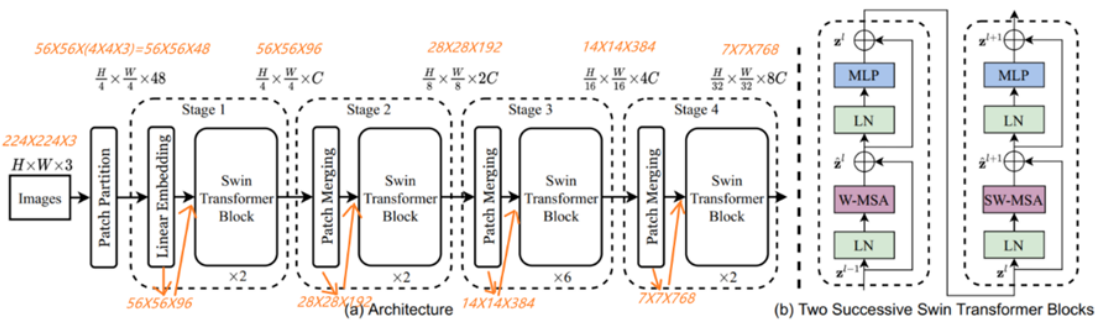

图 1. 方法示意图

3.2 特征提取模块

3.2.1 Patch Mergeing

Patch Mergeing 操作在行和列的方向上，选择每隔 2 个元素。然后将这些降采样后的张量沿新的维度拼接在一起，形成一个完整的张量，最后展平。在这个过程中，通道的维度会变成原来的 4 倍（因为 H 和 W 各减小了 2 倍）。接着，通过一个全连接层再次调整通道的维度，使其恢复到原来的两倍。如图 2所示：

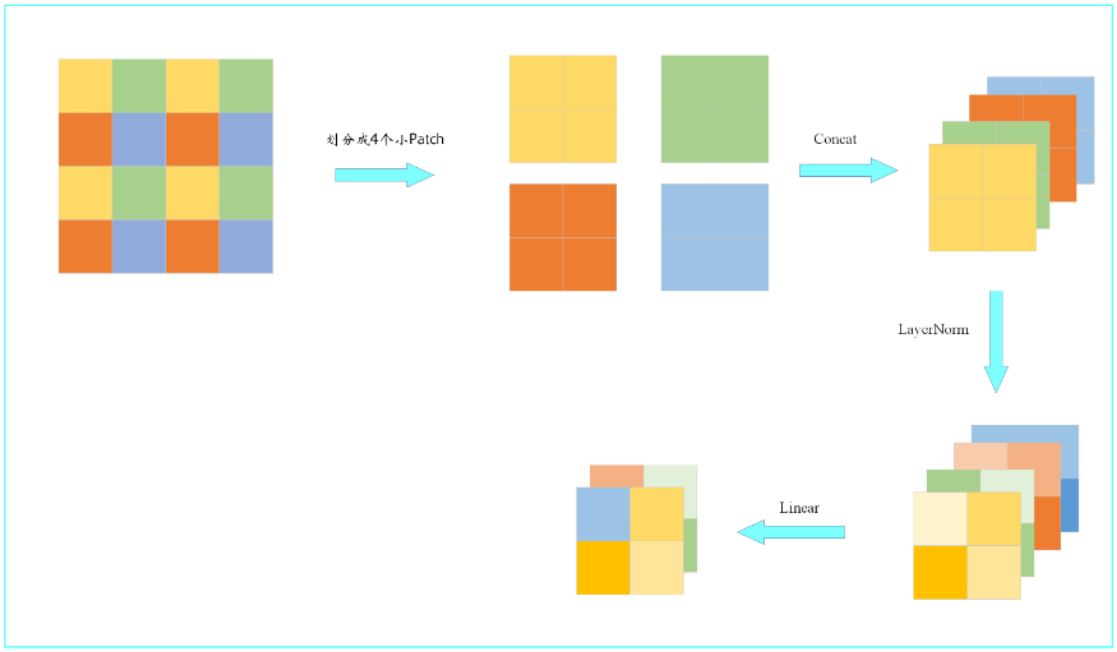

图 2. Patch Mergeing 示意图

3.2.2 SW–MSA 模块

在这种移位后，批窗口可由特征图中不相邻的子窗口组成，因此使用屏蔽机制将自注意计算限制在每个子窗口内。通过循环移位，批处理窗口的数仍与规则分区的窗口数相同。窗口移动效果如图 3所示：

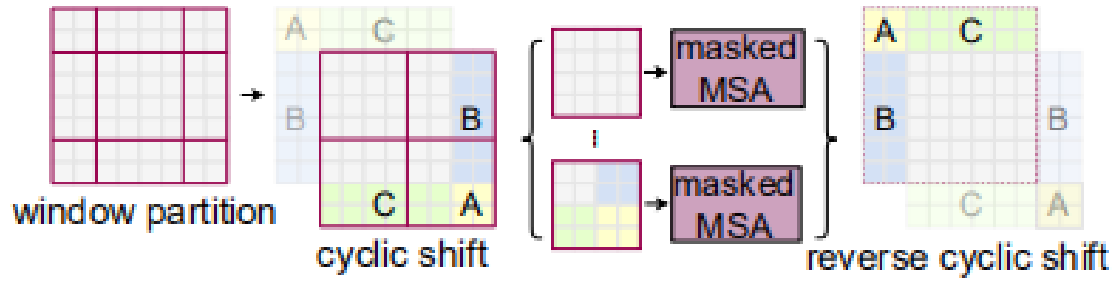

Figure 4. Illustration of an efficient batch computation approach for self-attention in shifted window partitioning.

图 3. 滑动窗口示意图

掩码方式如图 4所示：

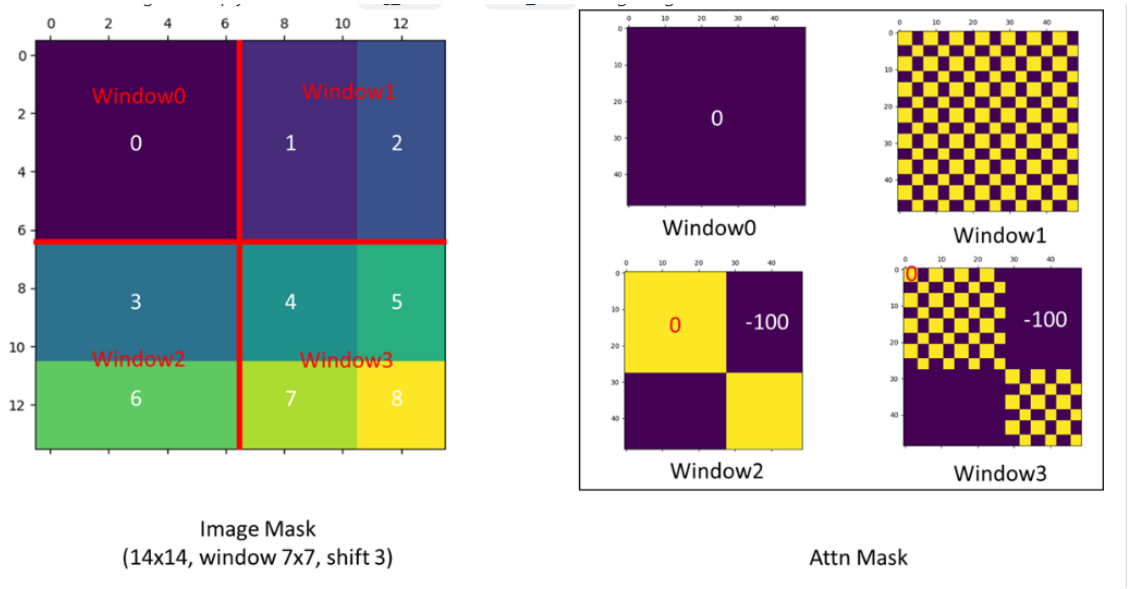

图 4. 掩码计算示意图

3.3 损失函数定义

本实验自己实现了一种标签平滑的损失函数，是一种防止过拟合的正则化方法。如果数据集本身的标签存在问题，这对模型的伤害是非常大的，因为在训练过程中会强制学习非本类的样本，并且让其概率非常高，这会影响后验概率的估计。并且有时候类与类之间并不是完全无关，如果鼓励输出的概率间相差过大，可能会导致过拟合。标签平滑的思想是不再使用独热编码作为目标，而是采用以下方法表示：

$$q_i = \begin{cases} 1-\varepsilon & \text{if } i = y, \\ \frac{\varepsilon}{K-1} & \text{else}, \end{cases}$$

其中 ε 是一个比较小的数，K 为预测的分类总数。即在真实类别的位置上留有较大的概率，但不是 1，其他位置上也有非零的小概率。这在一定程度上避免了过拟合，也缓解了错误标签带来的影响。

3.4 优化器

本实验采用的是 AdamW 优化器，它是一种常用的优化算法，可以自动调整学习率。AdamW 论文中的伪代码如图 5所示：

Algorithm 2 Adam with L_2 regularization and Adam with decoupled weight decay (AdamW)

given $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$
initialize time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
repeat
  $t \leftarrow t + 1$
  $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
  $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) + \lambda \boldsymbol{\theta}_{t-1}$
  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\boldsymbol{g}_t$ ▷ here and below all operations are element-wise
  $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\boldsymbol{g}_t^2$
  $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t / (1 - \beta_1^t)$ ▷ β_1 is taken to the power of t
  $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t / (1 - \beta_2^t)$ ▷ β_2 is taken to the power of t
  $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ can be fixed, decay, or also be used for warm restarts
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left(\alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) + \lambda \boldsymbol{\theta}_{t-1} \right)$
until *stopping criterion is met*
return optimized parameters $\boldsymbol{\theta}_t$

图 5. AdamW 伪代码

4 复现细节

4.1 与已有开源代码对比

本代码已开源。本工作采用了三种不同的的 loss 训练数据集，常规的交叉熵函数，自己改写的 timm 库中的 SoftTargetCrossEntropy 以及自己实现的平滑标签损失函数。最终平滑标签损失函数取得了最好的效果。train.py 利用 utils.py 生成数据集与验证集，然后通过 model.py 进行训练。源代码中模型不支持多尺度训练，现在的 model.py 在发现图片尺寸不是窗口大小整数倍的情况下会进行填充。utils.py 会随机取 80% 的数据作为训练集，剩余 20% 作为验证集。my_dataset.py 文件定义了一个名为 MyDataSet 的自定义数据集类，该类继承自 PyTorch 的 Dataset 类，用于处理和加载图像数据，用于将多个样本组合成一个批次。collect_wrong.py 的主要作用是找出在验证集中预测错误的图片，并将这些信息记录在 record.txt 文件中。predict_file.py 脚本的主要作用是对指定的数据集进行预测，并计算模型的准确率，并可以选择生成混淆矩阵显示模型对各个类别的预测情况。显示模型对各个类别的预测情况。

4.2 实验环境搭建

torch==1.13.0+cu116，tqdm==4.66.1，opencv-python==4.4.0.46，Pillow==9.5.0，显卡采用 A800。

4.3 创新点

参考相关资料，自己实现了一个损失函数，并和软标签交叉熵与传统交叉熵函数进行了对比。

5 实验结果分析

本实验对两个数据集进行了实验，分别是 flower_data 与 mammals_data。其中 flower_data 有 5 个类别，mammals_data 有 45 个类别。其中 flower_data 的混淆矩阵如图 6所示：

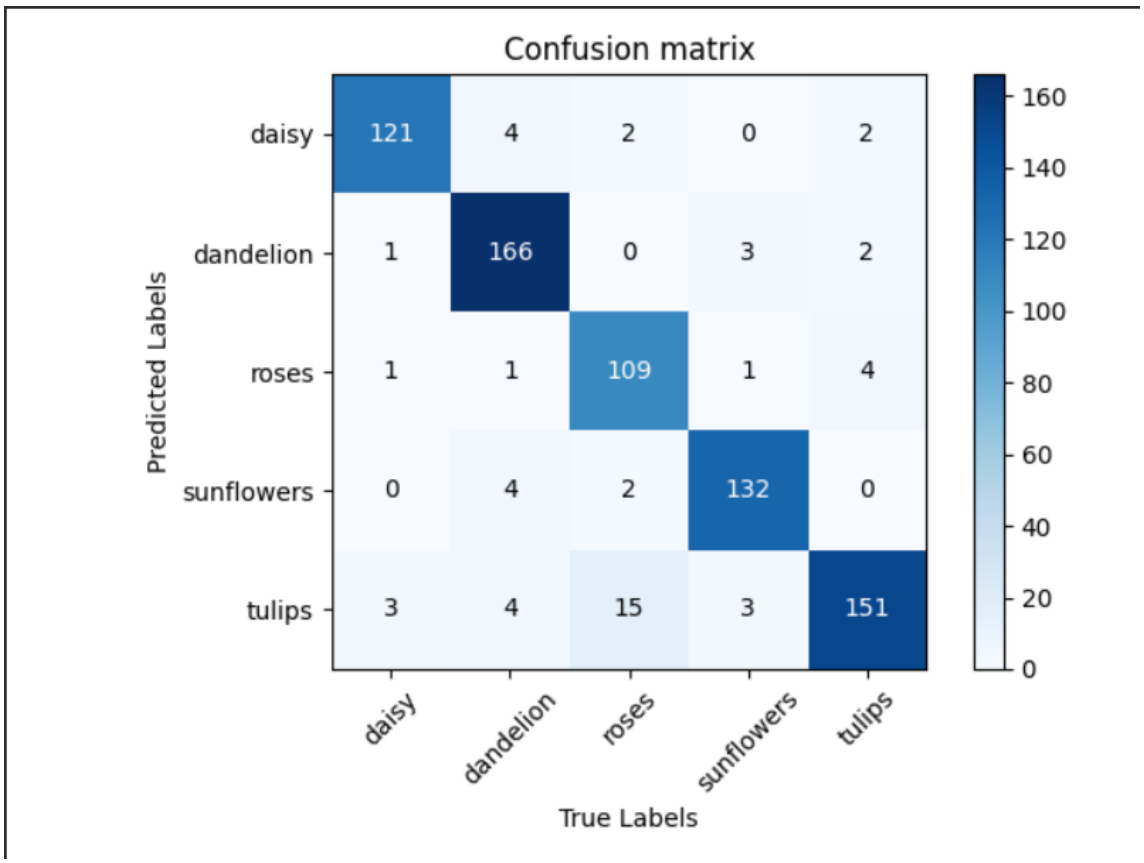

图 6. flower_data 的混淆矩阵

使用 cross entropy 的 mammals_data 的预测准确率如图 7所示：

```
Top-1 Accuracy:  90.73599414134017
Top-5 Accuracy:  98.90150128158184
```

图 7. mammals_data 的准确率 1

使用 timm 库中的 SoftTargetCrossEntropy 的 mammals_data 的预测准确率如图 8所示：

```
Top-1 Accuracy:  92.42035884291468
Top-5 Accuracy:  99.12120102526546
```

图 8. mammals_data 的准确率 2

使用自己实现的标签平滑损失函数的 mammals_data 的预测准确率如图 9所示：

```
Top-1 Accuracy:  94.25119004027829
Top-5 Accuracy:  99.37751739289638
```

图 9. mammals_data 的准确率 3

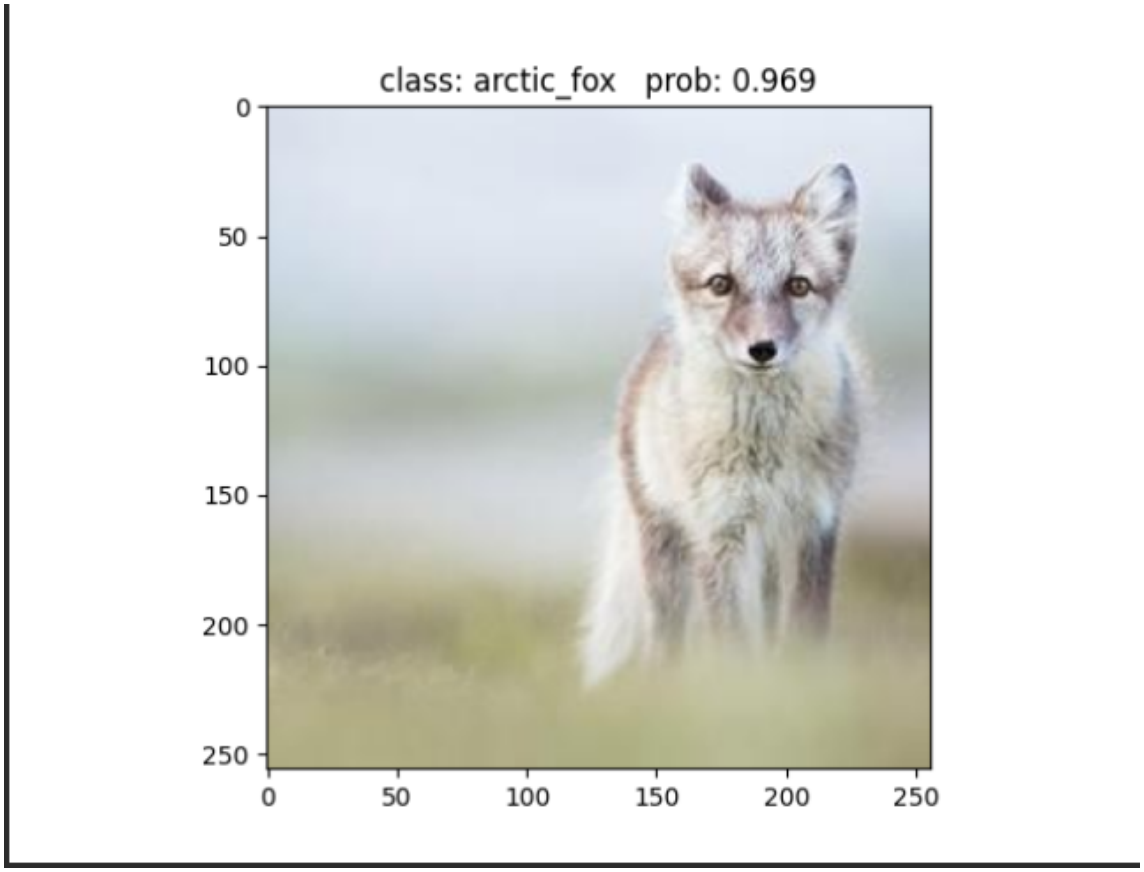

图 10. mammals_data 的 predict

6 总结与展望

通过研读相关论文和文档，Swin Transformer 的架构有了深入的理解，包括滑动窗口和多头注意力机制的实现原理。了解了数据加载、数据增强和训练过程的基本步骤，为将来在更广泛任务中应用模型奠定了基础。目前对于对于深度学习的理论知识和实践经验仍然较为薄弱，需要进一步学习深度学习的基本概念和常用技术。由于资源和时间有限，在小规模数据集上进行了训练和验证，对于模型在大规模任务中的性能表现尚未全面评估。总体而言，这次学习为我打开了深度学习的大门，但也揭示了在知识体系和实践经验上的不足。通过不断学习和实践，将能够更好地应对深度学习领域的挑战，提升自己的技能水平。

参考文献

[1] 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, 2021.

[2] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021, 2021.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[5] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.