一开始提视觉表征

CNN

早期多模态融合：

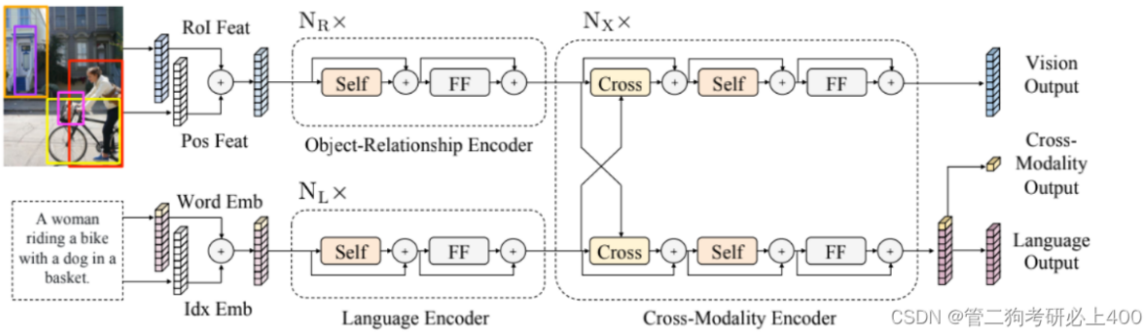
1.双塔对比学习（距离度量）

2.用transformer进行modality interaction

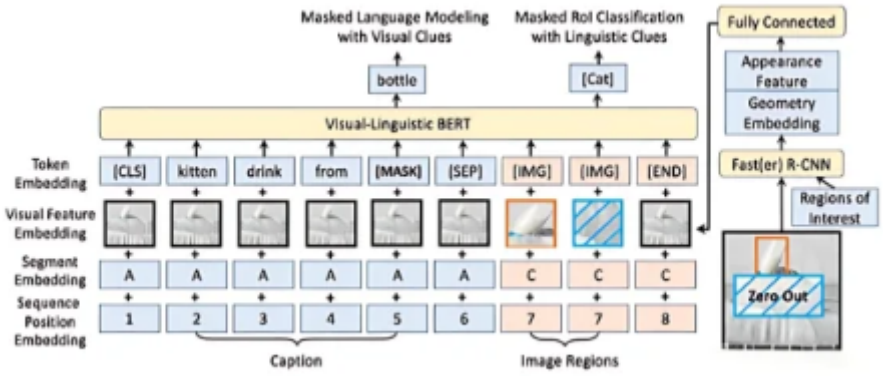
由此引申出如何对视觉特征编码来与文本tokenembedding对齐

输出很受限

LXMERT,两路输入



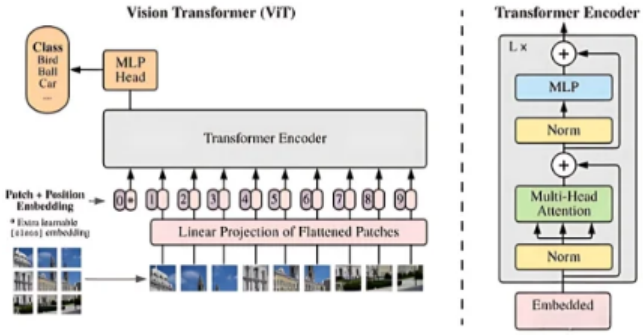
VL-BERT,单路输入



视觉表征提取发展（CNN -> VIT，进而影响模态融合）

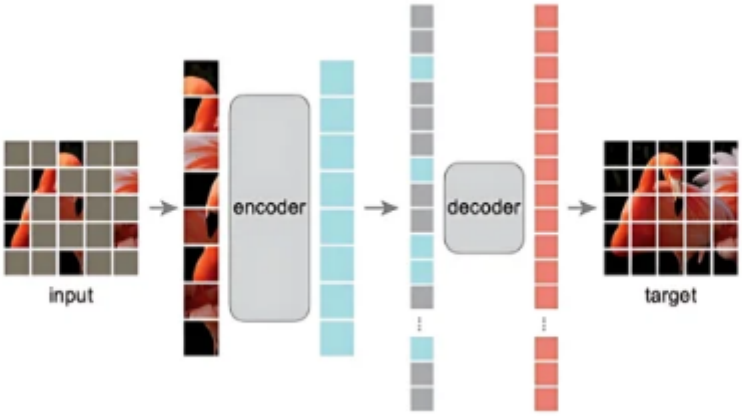
如图7，VIT将输入图片平铺成2D的Patch序列（16x16），并通过线性投影层将Patch转化成固定长度的特征向量序列，对应自然语言处理中的词向量输入。

> **同时，每个Patch可以有自己的位置序号，**通过Embedding层对应到位置向量。最终Patch向量序列和视觉位置向量相加作为Transfomer Encoder的模型输入，这点与BERT模型类似。



这个时期大量借鉴BERT，并且已经开始预训练+微调

MAE



BEIT

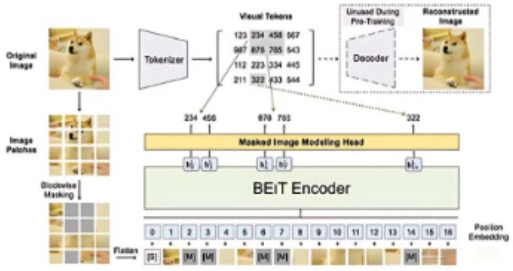


图9

具体的如图9，在预训练之前，BEIT先通过一个离散自回归编码器（discrete Variational AutoEncoder，dVAE）学习了一个“图像分词”器，最终可以将图像编码成离散的视觉Token集合。

由VIT引申出的多模态对齐

CLIP

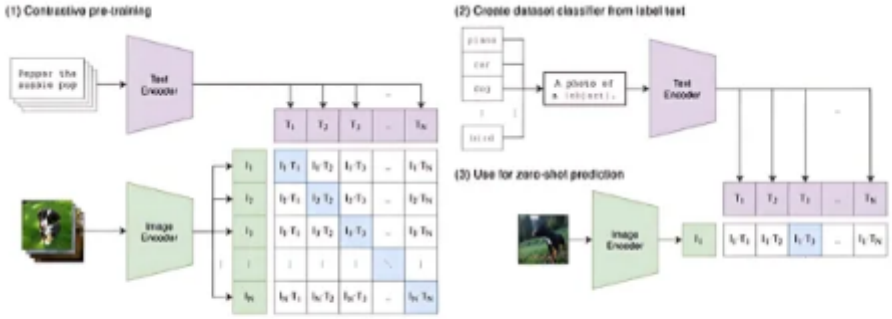


图11

VILT

直接拼接

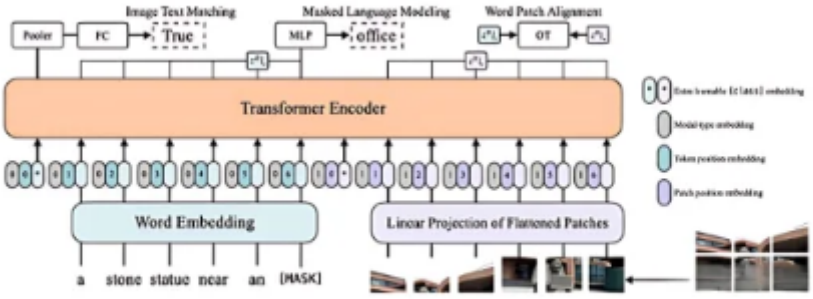


图12

ALBEF和BLIP

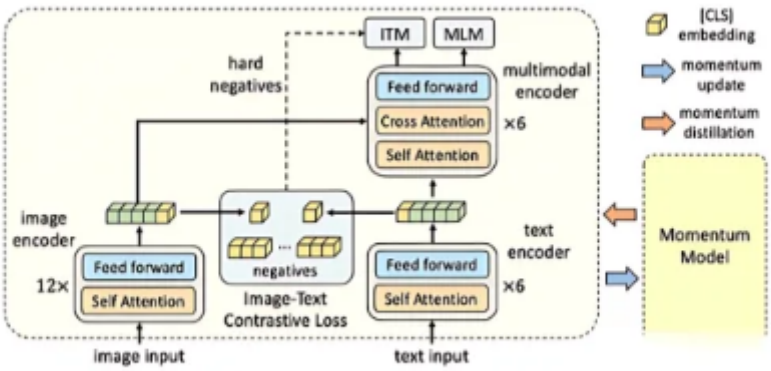


图13

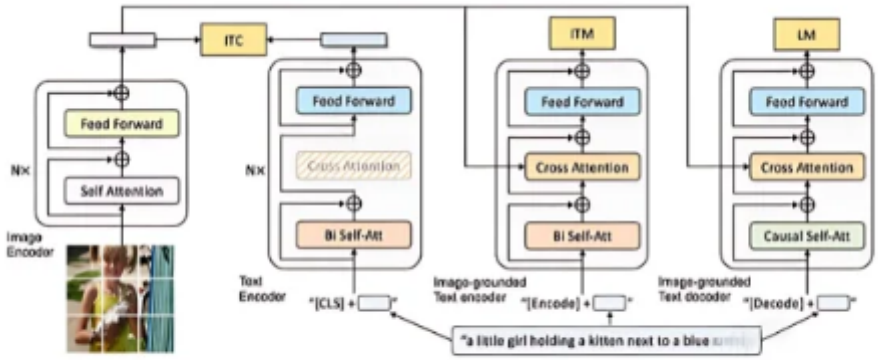


图14

VL-BEIT, VLMO, BEIT-3

2.2.4 VL-BEIT、VLMO与BEIT-3

- **ALBEF和BLIP之类的工作虽然能够同时兼顾对比和深度融合两种训练模式，**但视觉和自然语言仍然需要单独的Encoder分别编码，这显然还不是我们所期望的真正的多模态统一模型框架。

可以从Microsoft Research的VL-BEIT、VLMO与BEIT-3这一系列工作一窥这个方向的探索过程。

- VL-BEIT可以看作是前文提到的BEIT在多模态对齐预训练工作的延续，同时借鉴了ViLT的网络结构。

如图15，与ViLT的区别在于，VL-BEIT期望将单模态和多模态统一到一个模型中，在预训练任务设计上，同时考虑了纯文本、纯视觉以及图文多模态任务。

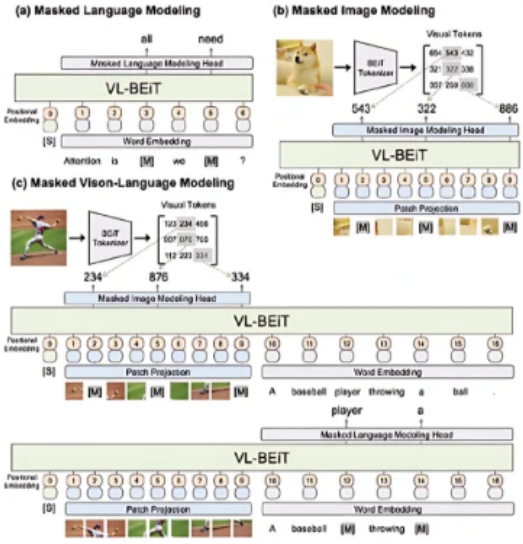


图15

VLMO是VL-BEIT的同期工作，如图16。

VLMO相较于VL-BEIT的不同之处在于：

- 1、舍弃了视觉侧的Visual Token ID预测，简化了整体的网络结构；
  2、增加了类似CLIP的图文对比学习任务，以及交互型的图文匹配任务。

- **虽然VLMO相对于VL-BEIT在效果上并不出彩，但为后续BEIT-3的工作奠定了基础。**网络结构上，VLMO是VL-BEIT都使用MoME Transformer结构，对不同的模态使用不同的Expert头，以区分不同模态的表征。

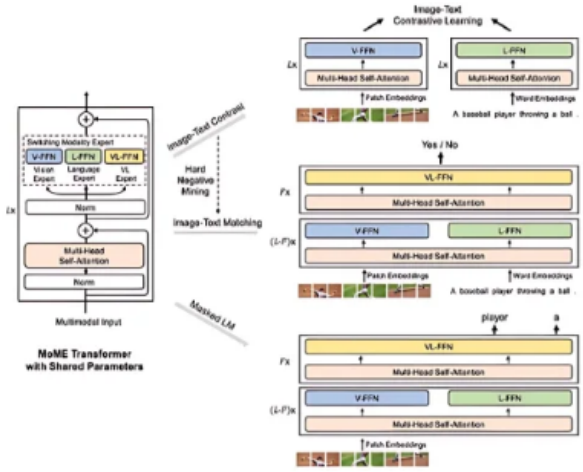


图16

在预训练任务上，如图17，BEIT-3相比之下也更加全面，不仅包括常用的图文对比学习、MLM和图像文本描述生成任务，还引进了文本和图像的单模态任务。

- **这样的训练方式，使得BEIT-3真正统一了多模态的不同输入类型，**同时更加全面和灵活的支持不同模态的下游任务。为了能够实现这样的能力，BEIT-3使用了更多的预训练数据，模型容量相对于之前的工作也有了显著的提高（达到1.9B），相应取得了在当时更好的效果。

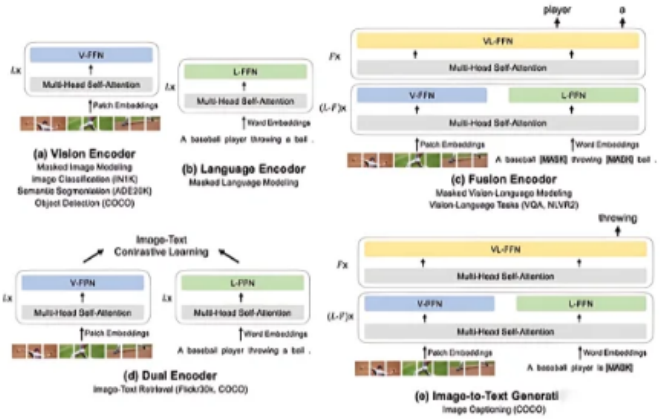


图17

多模态大模型

Flamingo

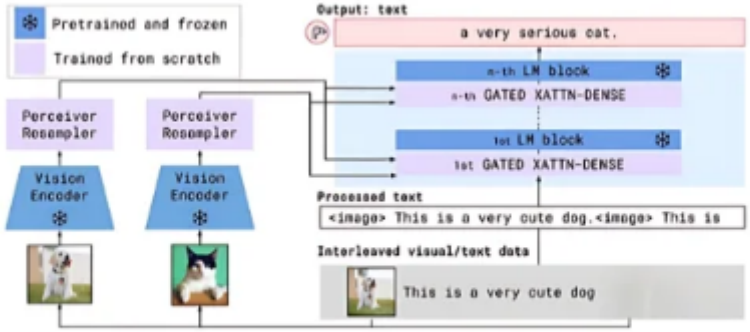
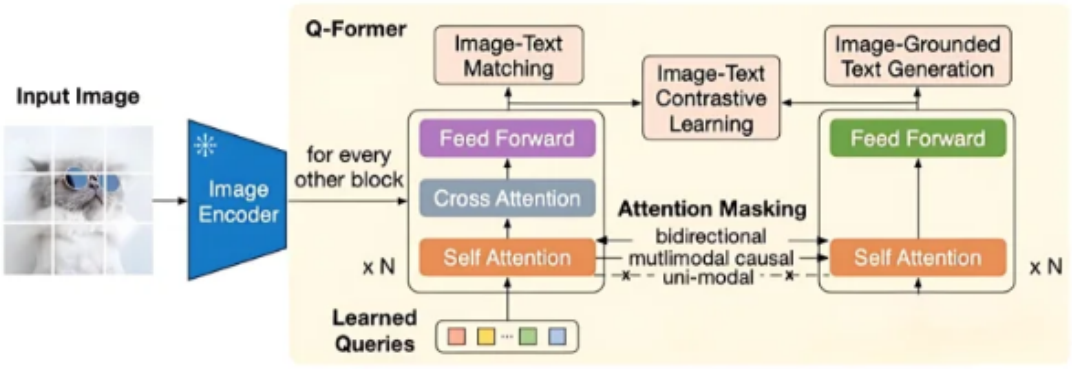


图19

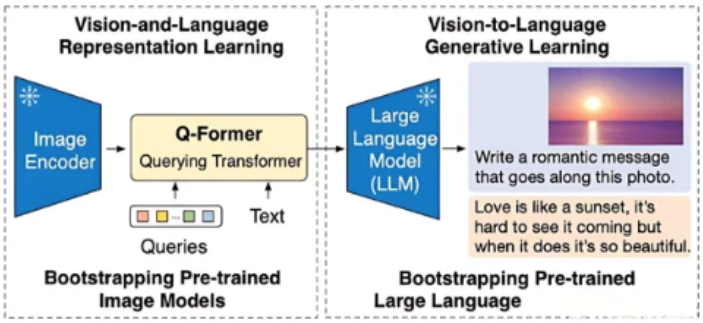
BLIP2

https://blog.csdn.net/jiaoyangwm/article/details/130048612



BLIP-2的全称是Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models，**核心思路是通过利用预训练好的视觉模型**和语言模型来提升多模态效果和降低训练成本。

BLIP-2的网络结构如图20所示，从架构上来说，和Flamingo十分类似。包括视觉编码层、视觉与文本的Adapter（Q-Former）以及大语言模型层。



QWEN-VL

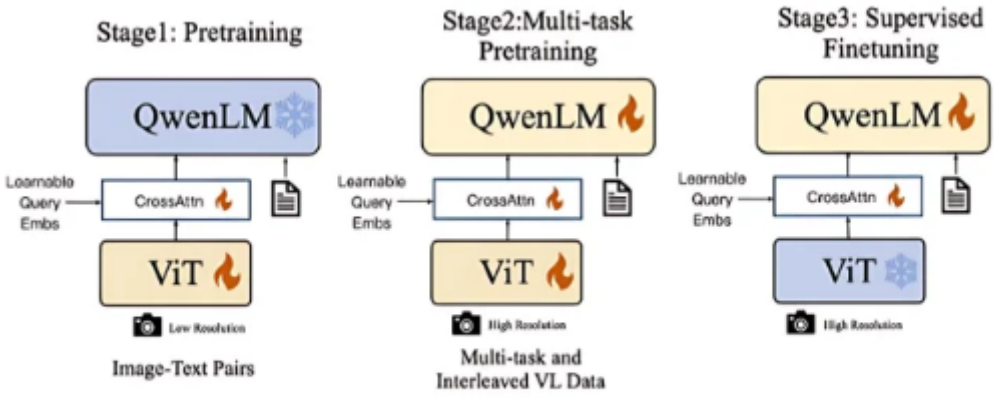


图24

LLAVA

..

VILA

不同的是VILA通过实验，总结了多模态预训练的一些经验，其中有些经验在相关工作中也有所体现，主要为以下三点：

- **LLM参与训练更好：**在预训练阶段冻结LLM参数，能做到不错的zero-shot的能力，但会损失in-context学习的能力，而LLM参数参与训练的话可以有效缓解；

- **预训练数据使用图文交替数据更好：**图文Pair对并不是最优的选择，图文交错的数据效果更好；

- **SFT时纯文本数据图文数据混合更好：**在图文指令微调训练数据中混入纯文本的指令数据，不仅可以缓解纯文本能力的遗忘，还能提升VL任务的能力。

具体的，如图28，VILA的训练分为3个阶段，视觉编码模块ViT参数均是冻结状态。

Step 0 使用图文Pair数据对初始化Projector（图文Adapter）参数，LLM模块参数冻结；Step 1使用图文交替数据全参数预训练；Step 2使用指令微调数据进行全参数微调，其中微调数据混合了图文指令和纯文本指令；

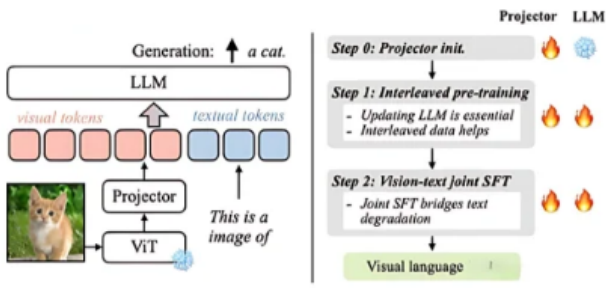


图28

补充：

https://blog.deephour.com/index.php/2024/04/02/%E5%A4%9A%E6%A8%A1%E6%80%81%E5%A4%A7%E6%A8%A1%E5%9E%8B-clip-blip-blip2-llava-minigpt4-instructblip-%E7%B3%BB%E5%88%97%E8%A7%A3%E8%AF%BB%E8%BD%AC%E8%BD%BD/

MiniGPT-4

基本思想：

GPT-4 具有先进的多模态生成能力的主要原因在于利用了更先进的大型语言模型（LLM），因此提出仅用一个投影层将一个冻结的视觉编码器和一个冻结的 LLM（Vicuna）对齐。

模型结构：

类似BLIP2，包括一个冻结的视觉编码器（ViT-G/14 + Q-Former），一个冻结的 LLM（Vicuna），一个投影层。

两阶段训练：

第一阶段在大量对齐的图像文本对上对模型进行预训练，以获取基础的视觉语言知识。 在第二阶段，使用规模较小但更高质量的图文对数据集和精心设计的对话模板对预训练模型进行微调，以增强模型的生成可靠性和可用性。关于loss设计论文没有讲太细，看代码里面应该主要是计算language modeling loss：
https://github.com/Vision-CAIR/MiniGPT-4/blob/main/minigpt4/models/mini_gpt4.py#L320

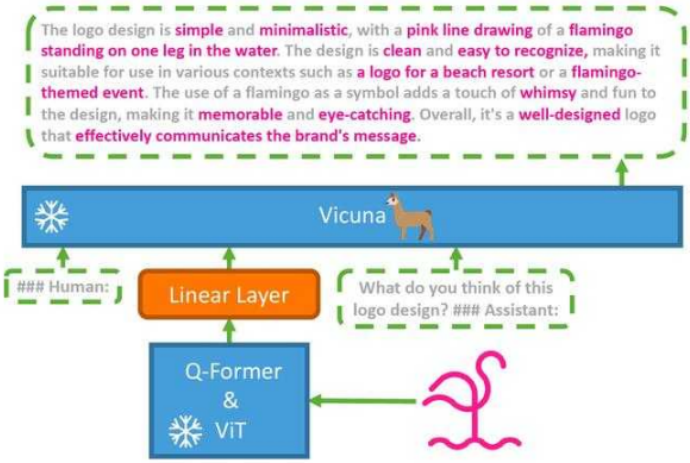


Figure 1: **The architecture of MiniGPT-4.** It consists of a vision encoder with a pretrained ViT and Q-Former, a single linear projection layer, and an advanced Vicuna large language model. MiniGPT-4 only requires training the linear projection layer to align the visual features with the Vicuna.