对文献(ColPali: Efficient Document Retrieval with Vision Language Models)的理解

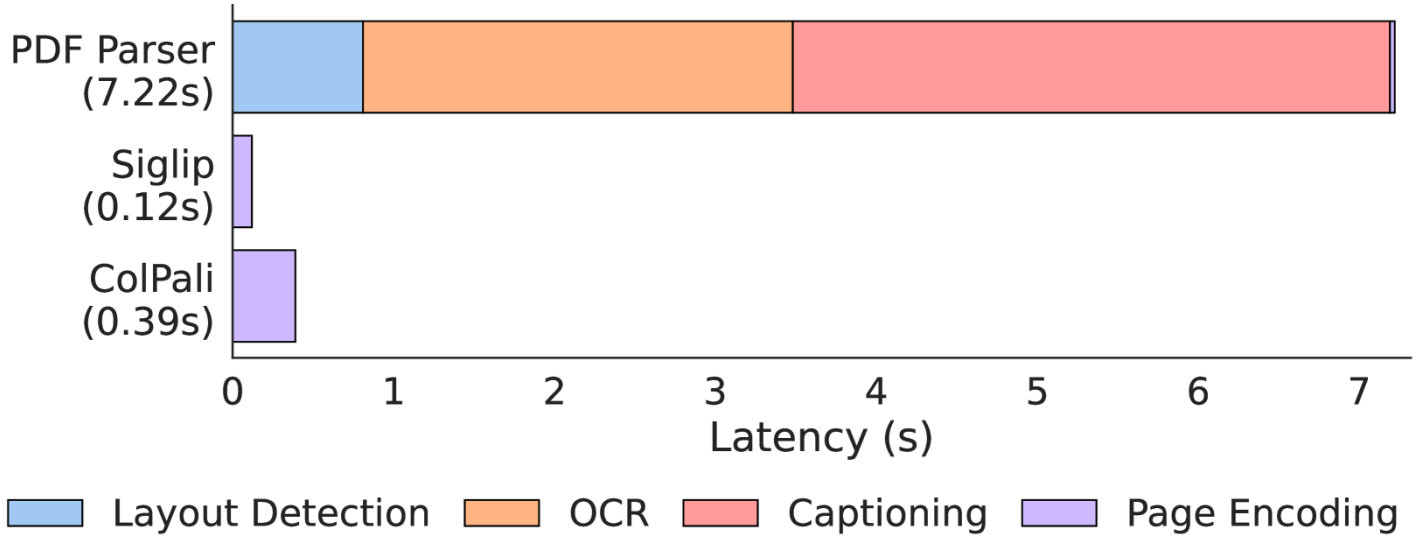
ColPali： 利用视觉语言模型进行高效文档检索

文献地址：ColPali: Efficient Document Retrieval with Vision Language Models

huggingface地址：https://huggingface.co/google/paligemma-3b-pt-224

背景

在现在的文档中有不同的文档格式 (PDF, DOCX, XLSX, JPG等等)，而文档中又包含不同的类型，图表，页眉，页脚…… 而RAG系统要想提取就需要依赖 OCR 工具，尽管 OCR 技术有了不小的改进，但还是会产生错误，这会导致不相关或错误信息的文本块被索引，对LLM回答问题的质量产生影响。故我们就引入了 ColPali ,从复杂的 PDF 文档中推断信息。



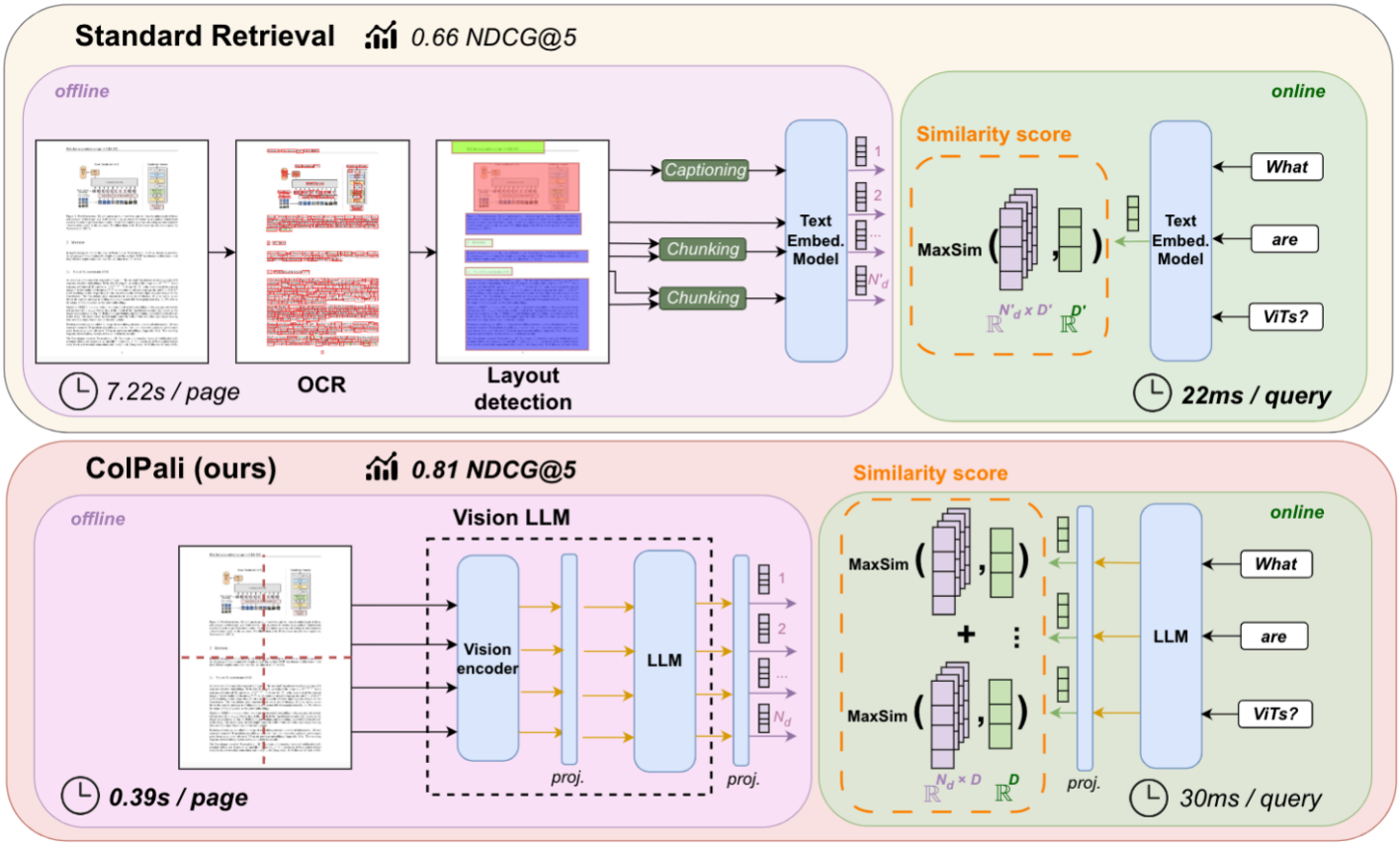
从图中我们可知，使用 ColPali 进行离线索引要简单得多，速度也快得多。这里 SigLIP(2023) 模型架构，是多模态学习模型，使用大量的图像-文本对，旨在学习更好地捕捉图像与文本之间的关系。文章中运行了SigLIP作为模型的图像编码器。

SigLIP文献地址：Sigmoid Loss for Language Image Pre-Training

huggingface地址：https://huggingface.co/google/siglip-so400m-patch14-384

ColPali介绍

1. **定义**
   ColPali （Collaborative Pali for Document Retrieval）是一种新颖的文档检索模型(多模态学习模型)，它利用视觉语言模型 (VLM) 的强大功能，仅根据文档的视觉特征高效地索引和检索文档中的信息。特别关注于提高多模态信息检索的效率和准确性。在RAG系统中，它被用作视觉检索器。Pali（Pathways Language and Image）在深度学习领域主要指的是一种多模态学习的框架或模型，旨在有效地结合视觉和语言信息，以解决复杂的多模态任务。
2. **特点**
   仅使用视觉特征的高效文档索引；低延迟；处理各种文档类型；使用 COLBERT 策略解锁文本和图像之间的高效交互且可端到端训练



从上面的图可知，一般的检索步骤是：首先，使用 PDF 解析器或光学字符识别 (OCR) 系统从页面中提取单词。然后可以运行文档布局检测模型来分段段落、标题和其他页面对象，例如表格、图形和标题。然后定义分块策略，以具有一定语义一致性的文本段落进行分组，现代检索设置甚至可以集成字幕步骤，以自然语言形式描述视觉丰富的元素，更适合嵌入模型。而 ColPali 只需一步就直接识别文档页面的图像。

补充：30ms/query: 每个查询平均处理时间为 30 毫秒**

指标：NDCG@5 是用于评估搜索系统在前 5 个结果(得分最高前5)中对相关文档进行排名的指标。

DCG(Discounted Cumulative Gain)：这衡量文档排名列表的相关性，同时考虑每个文档的相关性及其在列表中的位置。

$$\mathrm{DCG} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$$

- p: 计算DCG时考虑的文档数量
- rel_i: 第 i 个文档的相关性得分，通常是一个整数，表示该文档与查询的相关性
- 每个文档都有一个相关性分数，且是有范围的

$$\mathrm{NDCG} = \frac{\mathrm{DCG}}{\mathrm{IDCG}}$$

- 标准化：标准化方面来自将预测排名的 DCG 与理想排名的 DCG（即基于相关性的最佳文档顺序）进行比较。这通常使用 Ideal DCG （IDCG） 来完成。
- IDCG: 是按相关性最高的文档排序后的DCG，表示最佳排名的得分。

对于指标的具体解释[1]

目的：计算 NDCG@5 有助于确定实际排名与理想方案的接近程度，取值范围[0,1]，值越接近 1 表示性能越好。 Colpali+Late Inter在 ViDoRe上的 的 NDCG 平均值为0.813

3. **模型架构**
   核心：PaliGemma-3B 该模型结合了 SigLIP-So400m/14 vision encoder 与 Gemma-2B language model

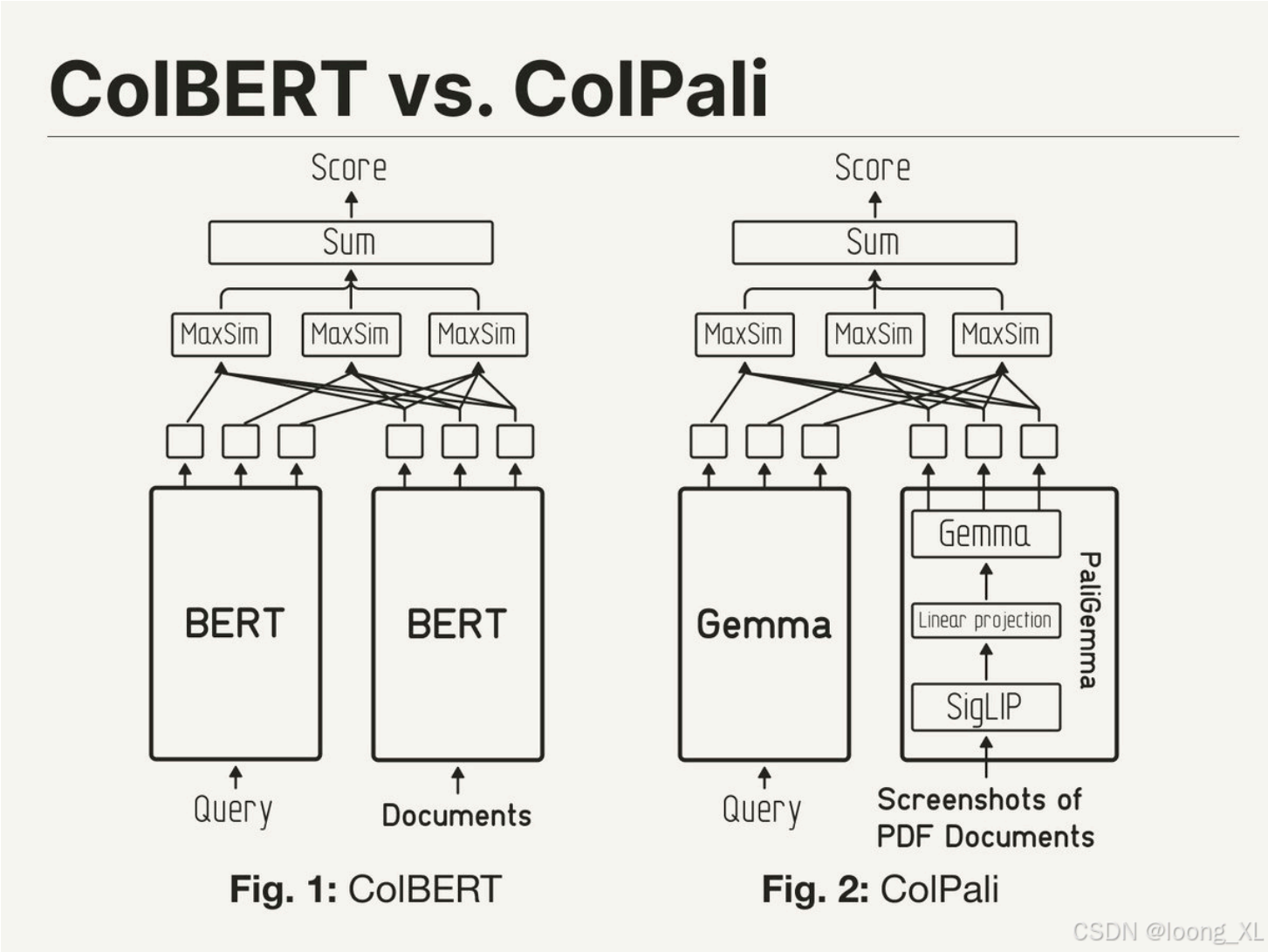
PaliGemma-3B文献：PaliGemma: A versatile 3B VLM for transfer

huggingface地址：https://huggingface.co/google/paligemma-3b-pt-224

> SigLIP图像编码器：这是一个强大的视觉-语言双向编码器，已经经过WebLI英文数据集的大规模预训练。

> Gemma-2B 解码语言模型：这是一个解码器的语言模型，旨在平衡模型大小与性能。它接收来自 SigLIP 编码器的图像嵌入，并将其与文本嵌入融合，以生成最终的多模态表示。

具体执行步骤：通过将图像分割成一系列块来对图像进行编码，这些块被馈送到视觉转换器 (SigLIP-So400m)。这些块嵌入被线性投影(这里应该是为了嵌入更高维的向量空间，捕获视觉特征)并作为“软”标记(嵌入是连续的向量)输入到语言模型（Gemma-3B），以便在语言模型空间中获得高质量的上下文块嵌入，然后我们将其投影到较低维度（D = 128）高效存储。这就是每个页面图像的多向量文档表示的构造和存储方式。



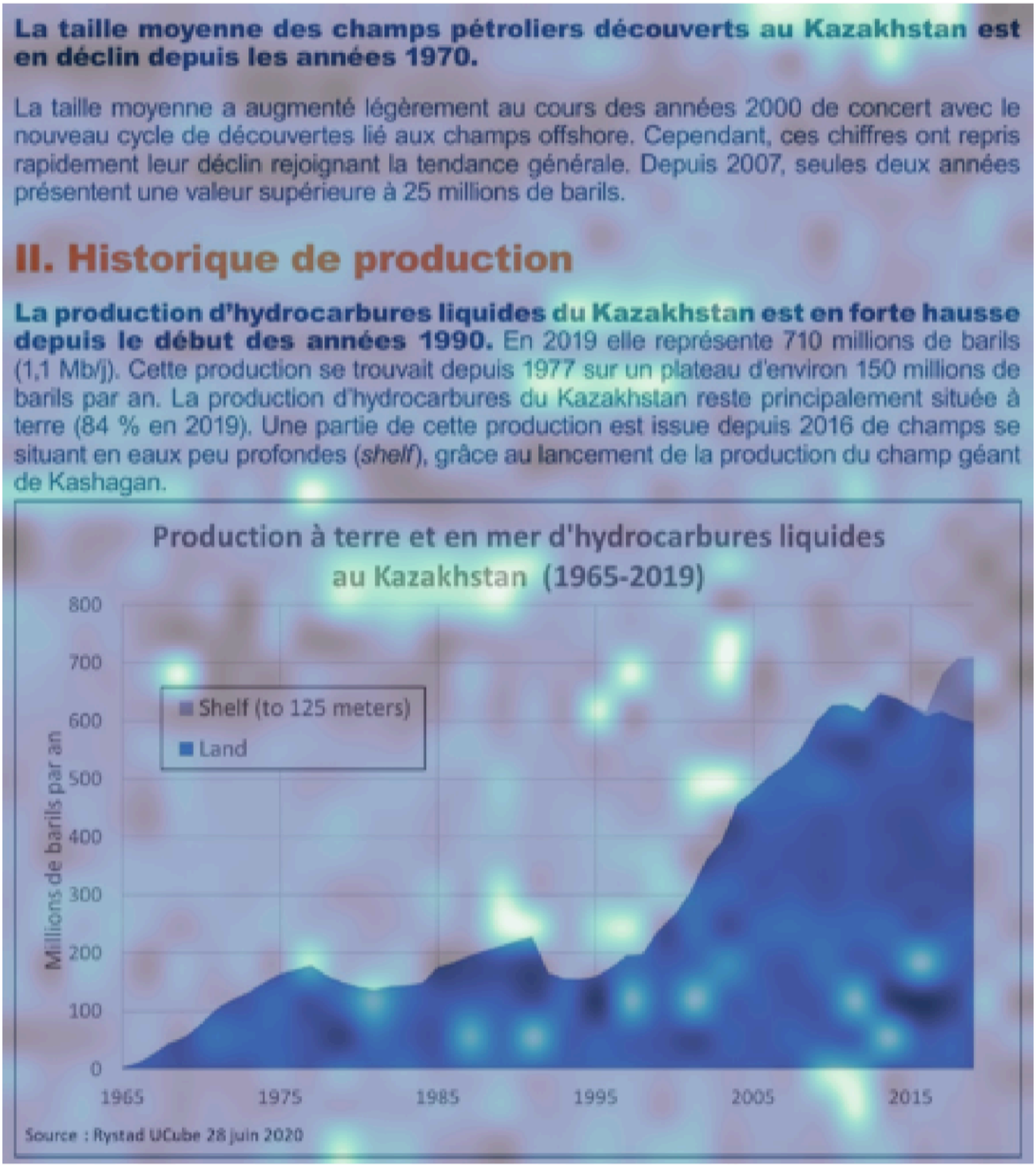
4. **工作流程**

主要分为离线索引阶段(Offline indexing phase)与在线查询阶段(Online querying phase)

> - 页面通过视觉编码器(SigLIP)，将页面全部转换成图像格式(32×32)在colpali中是**1024 patchs**，如果是独立的SigLIP则是**729 patchs**
> - 生成的图像块嵌入解码器语言模型(Gemma-2B)处理
> - 投影层将输出映射到低维空间(D=128)(每页的内存占用控制在约 256KB)
> - 生成的嵌入存储为文档页面的多向量表示

> - 查询时语言模型进行编码
> - 后期交互机制计算查询标记和文档补丁之间的相似度分数
> - 系统根据这些分数返回最相关的文档

(这里结果可以将后期交互热图叠加在原始图像之上，可视化的看到每个术语相关的最突出的图像块，如下图所示)



Query: "Quelle partie de la production pétrolière du Kazakhstan provient de champs en mer ?"

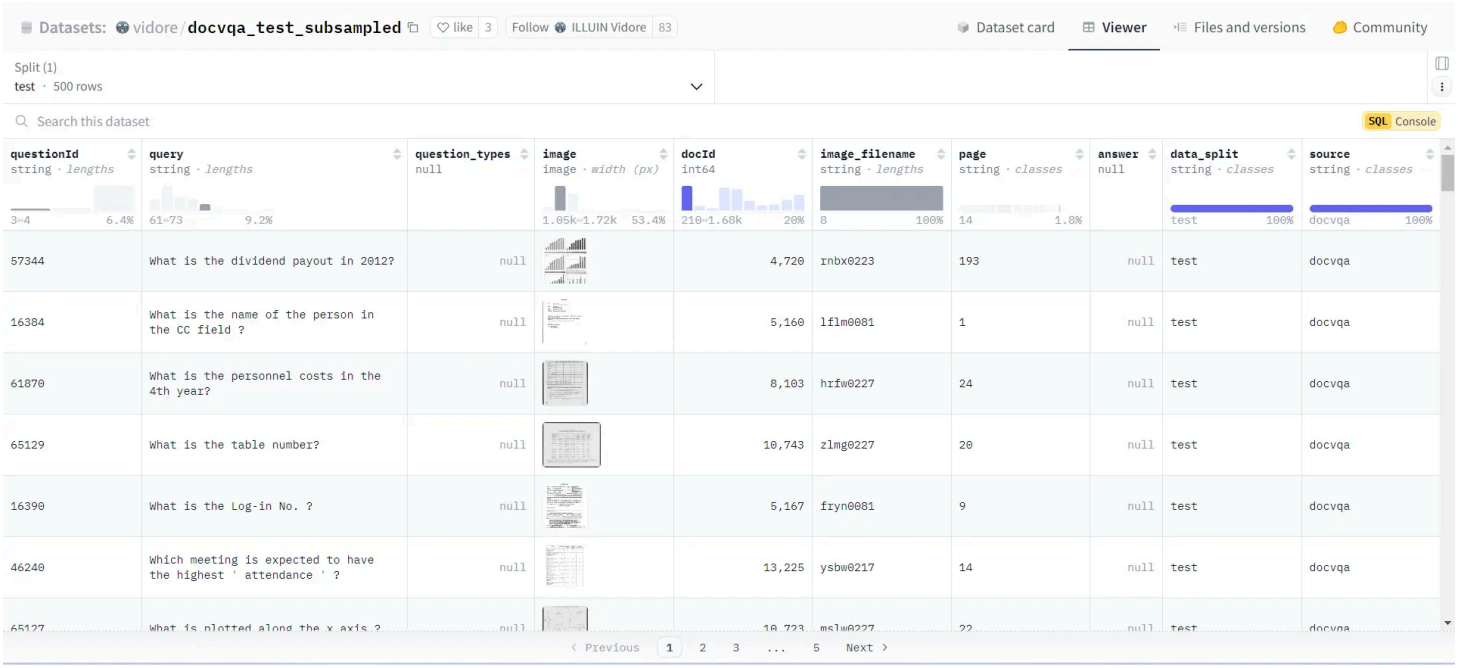
5. **性能基准**

   这里为评估 ColPali 性能，官方引入一个名为 ViDoRe(视觉文档检索)的基准。

认为文档检索系统不应仅根据文本嵌入模型的能力来评估，但也应考虑要检索的文件的上下文和视觉元素。

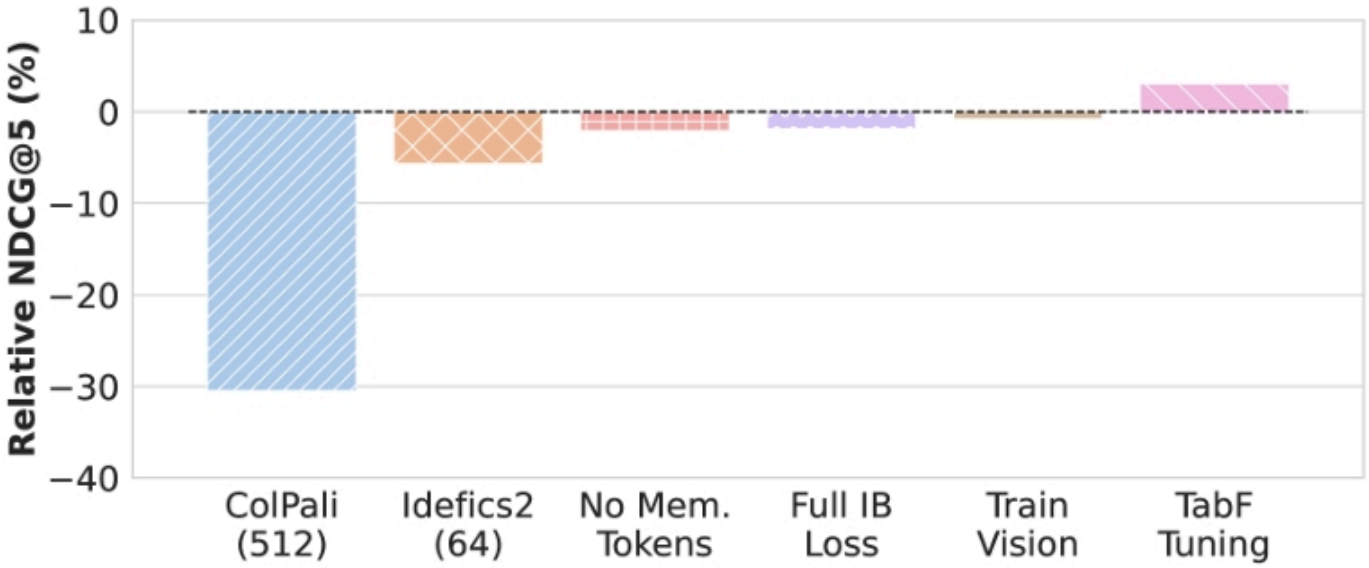
ViDoRe 旨在全面评估检索系统在页面级别将查询与相关文档匹配的能力。该基准测试包含多个任务，侧重于各种模态 - 文本、图形、信息图表、表格; 主题领域 - 医学、商业、科学、行政; 或语言 - 英语 、法语。

文献的数据集都在 fuggingface 中。数据集地址



上图为对数据集页面的预览

6. **消融研究**



PaliGemma 模型的图像块从 1024 减半至 512。尽管块数减少导致性能略有下降，但内存的使用大大减少。
Idefics2-8B 虽然规模大，但是采用64 patch

- 总结：模型在性能、延迟、内存消耗之间是权衡的。
- 微调视觉编码器和投影层并没有显著提高模型的效率。
- 在ColERT中，(Memory Tokens）)特定增强策略(即在不影响查询内容的情况下，对模型提供额外的上下文提示，使更好的理解意图)可能对不同语言有不同的影响，对英语没有显著变化，对法语有显著的提升。
- 在使用批量内负对比损失（即在同一批次中考虑多个负样本进行对比学习）进行训练时，模型的聚合基准性能可能会略微下降(2.4%)。这与只考虑最难负样本的成对交叉熵（CE）损失的训练方式相比，后者可能在性能上表现更好。
- 模型可以适应新的任务，端到端的训练，结果表明NDCG@5 有显著的改进。

7. **不足**

- 关注范围有限：虽然评估了多种文档模式（如图形、文本、表格、信息图表），但主要聚焦于PDF文档。这可能限制了模型在其他格式或类型文档上的适用性和性能。
- 语言资源不均：模型主要针对高资源语言（如英语和法语），尽管展示了对微调集之外语言的推广能力，但在模型的语言主干中未包含的语言上的表现尚不明确。这可能导致在某些低资源语言中的效果较差。
- 实际应用中的局限性：文中提到的设置假设存在相关文件，而在实际应用中，信息检索系统可能面临更复杂的场景，例如信息缺失或不确定性较高的环境。因此，模型在面对这些实际情况时的表现仍需进一步研究。
- 置信度估计的缺乏：虽然置信度估计在信息检索中可能是重要的，但该模型的设置没有探讨如何在置信度不确定的情况下进行有效的检索。这可能影响模型在真实场景中的可靠性和准确性。

结论

ColPali 通过 VLMs 模型以及后期交互机制，其结果的显著性远超现有的文档检索管道，并且发布一个全面的基准数据集 ViDoRe,用于评估在页面级别文档检索任务上的系统性能。

ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT

ColBERT：基于 BERT 的上下文化后期(延迟)交互的高效且有效的段落搜索

文献地址：ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT

huggingface地址：https://huggingface.co/colbert-ir

背景

在当时(2020)利用微调深度语言模型(LM)进行文档排名虽然非常有效，但计算成本是增加了几个数量级。每次的查询-文档需要计算相关性分数。所以为了解决这一问题，我们引入 ColBERT。

ColBERT介绍

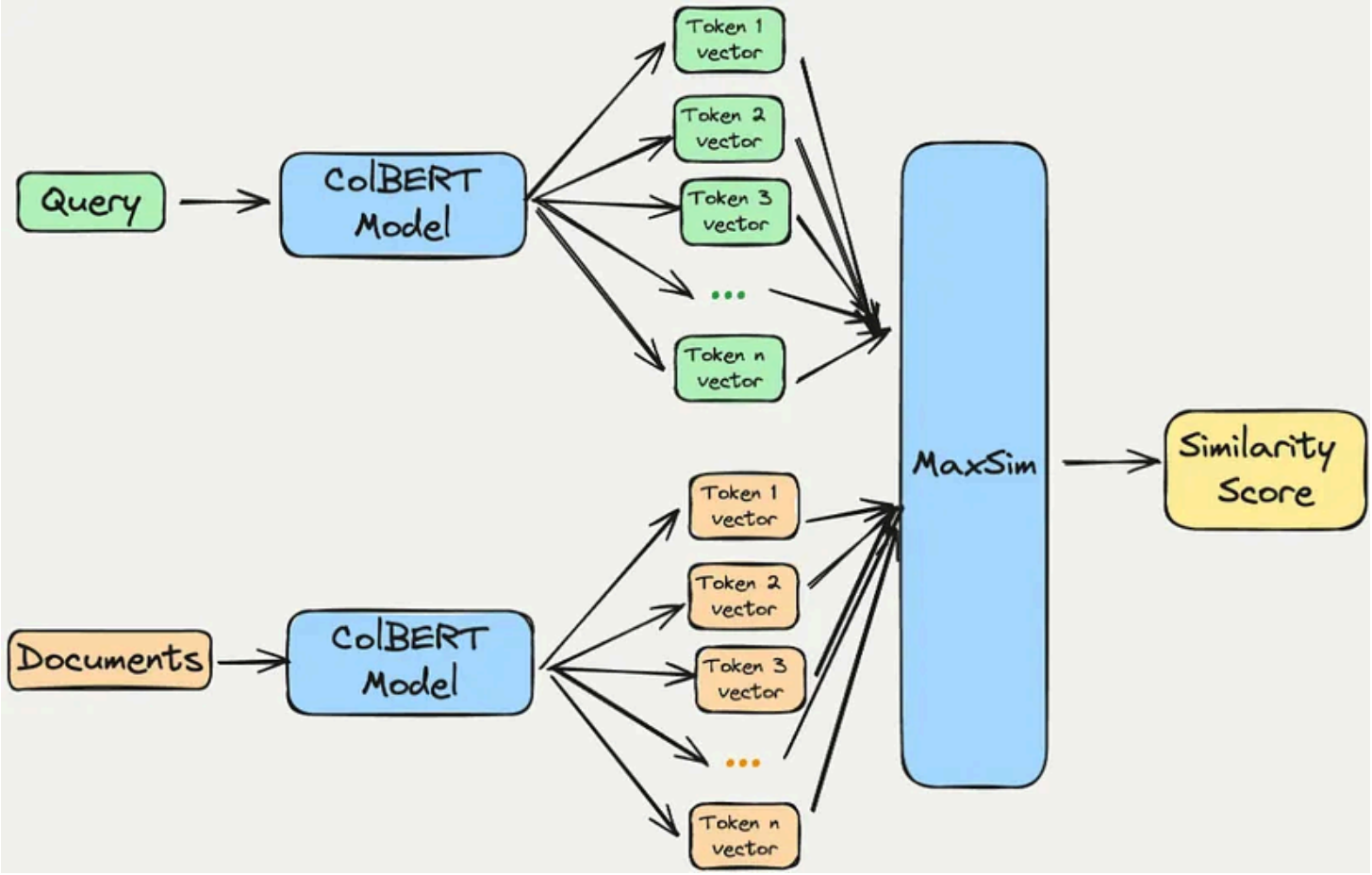
1. **定义**
   COLBERT(Contextualized Late Interaction over BERT) 是一种专为高效信息检索而设计的创新模型。
2. **特点**
   ColBERT 是基于BERT上的改进，加快了查询处理速度，降低了文档进行重新排序的成本。
   独立编码、后期交互(可以通过压缩和聚类机制大幅改善朴素 ColBERT 索引策略的内存占用)和高效的稀疏计算

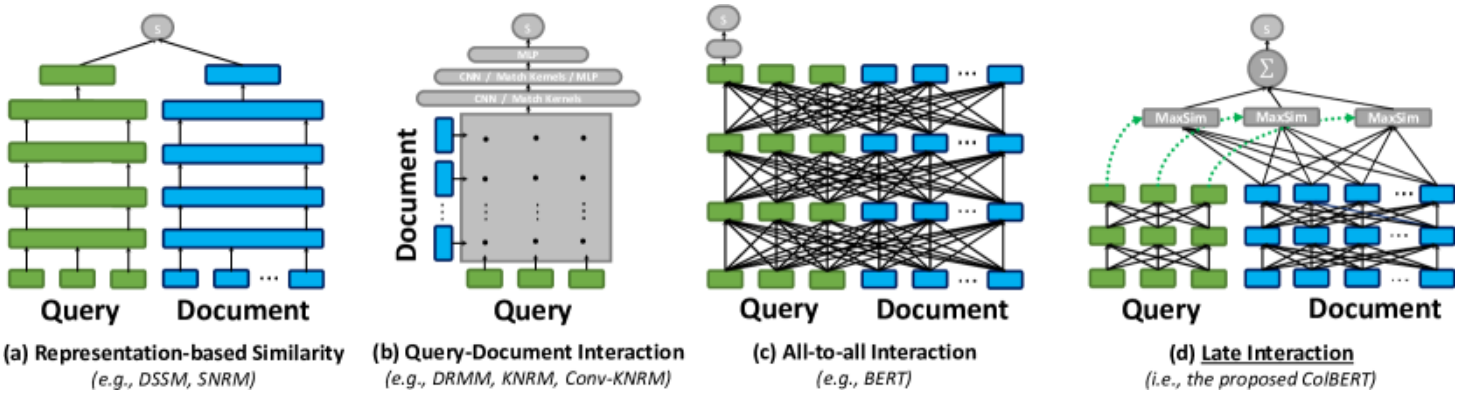
BERT 在信息检索中属于前期交互，模型在用户发出请求时，把用户的查询问题和文档内容一同编码，这样每次请求都要重新计算两者的匹配。这种方法的语义理解效果好，但效率低；而 COLBERT 在用户查询之前，文档就已经单独编码好了，存储成易于检索的表示。查询到来时，只需要将查询问题编码，再在最后一步和文档编码进行轻量级的匹配，这样一来就提升了效率，适合快速响应大规模数据检索。二者各有优缺，视情况选择。

BERT 与 ColBERT 具体方面的不同：

- 处理方式：BERT 将查询和文档作为一个整体输入进行编码，而 ColBERT 则分别编码查询和文档，允许更灵活的相似性计算。

- 相似性评分：BERT 采用的是整体相似性计算，而 ColBERT 利用后期交互机制，允许对文档进行稀疏检索和快速评分。
- 目标应用：BERT 更加通用，适用于多种下游任务；ColBERT 则专注于信息检索，特别是在高效处理大规模文档集时的性能优化。



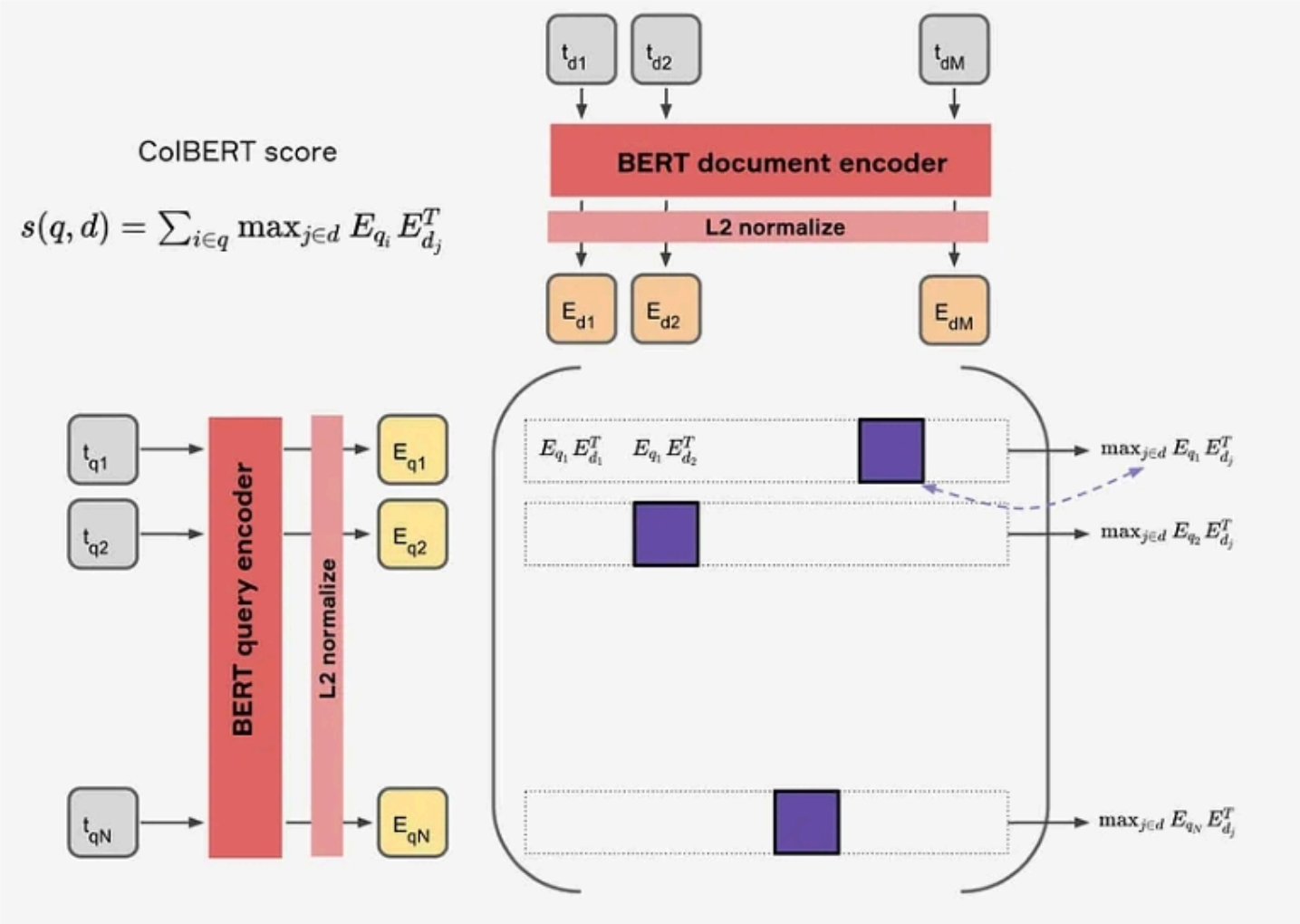


从图中我们发现后期交互与现有的神经匹配范式，(a)它独立计算一个嵌入 q向量和另一个嵌入 d向量，并将相关性估计为两个向量之间的单个相似性分数
通过计算查询和文档之间的相似度; (b)是通过深度神经网络来建模 q 和 d 之间的单词和短语级关系并进行匹配；(c)同时建模了 q 和 d 内部以及跨之间的单词交互; (d)通过隔离文档和查询的编码过程，可以离线预先计算文档编码，从而显著减少每个查询的计算负载

依赖于细粒度的上下文后期交互,Query 和 Document 文本使用两种不同的 BERT 模型分别编码（分词化）到上下文嵌入中

我的理解：通过不同的编码器，独立分别的将D 与 Q 进行编码，然后将多组嵌入向量分别进行交互，此时是细颗粒度的相似性匹配，这样使搜索更加精确，然后进行剪枝来减少计算量，使用最大池化操作进行相似性评分，进行排名(Top-k)，得到输出，这样既精确又节省了计算资源

注意：单个的maxSim只是每个查询词与文档中最相似词的相似度，求和是一篇文档的总相似性，然后返回得分前几篇的文档，进行整合一起生成一个连贯的回答。
如果查询的令牌 Nq 少于预定义的数量，则我们用 BERT 的特殊 [掩码] 令牌填充它，最长为 Nq（否则，我们将其截断为前 Nq 令牌）。在截断的情况下，ColBERT 返回溢出的令牌以及输出。（即输入长度被模型是固定的，会采取不同的措施使它达到那个长度，目的是确保了模型的输入的一致性）
(虽然文档的块数量可能不够（例如，最后一个块可能只包含少量词），但是只要这些块的向量的维度与查询向量的维度一致，便可以进行运算。这是因为向量的维度是由模型的架构决定的，而不是由输入的长度直接影响。)



3. **架构**
   ColBERT由三部分组成

- 查询编码器 fQ
  (batch=32)每个维度使用4个字节(32位浮点数)； 每个查询的表示由 32 个 128 维的嵌入向量组成;
- 文档编码器 fD
- 后期交互机制
  给定查询 q 和文档 d，fQ 将 q 编码到一个固定大小的嵌入向量 Eq 中，而 fD 将 d 编码到另一个 Ed 中，使用Eq 和Ed 通过后期交互计算 q 和 d 之间的相关性分数(这里选择的是MaxSim算子的求和)
  无交互模型（也称为基于表示的模型）将查询和文档编码为单向量表示，并依靠简单的相似性度量（如余弦相似度）来确定相关性。 但是他们无法捕获查询和文档术语之间的复杂细微差别和关系。这时就需要有交互的，并且时延迟交互(即查询和文档表示之间的交互发生在检索过程的后期，在两者都经过独立编码之后的过程。)

4. **工作流程**

> - 输入处理。将查询与文档进行分割(为了与BERT兼容)
> - 独立编码。分别将D 与 Q 进行编码，生成一系列的词向量嵌入序列(细颗粒度)，其中D 的编码在查询之前就已经完成。
> - 初步检索。使用 L2 距离和FAISS 索引从大量的文档集合中筛选出可能相关的候选文档。
> - Top-k检索。基于计算出的相关性分数(MaxSim),进行剪枝，排序，选择得分最高的前k 个文档作为检索结果。

5. **模型的训练**
   训练ColBERT-弱自我监督训练
   ColBERT 使用成对的正负文档进行训练，通过 softmax 和交叉熵损失函数优化模型，使其能够更准确地区分正样本和负样本（相关与不相关的文档）。
6. **评估指标**:

- MMR@10：平均倒数排名，主要用于多次查询的结果中，以计算模型在所有查询上找到正确答案的平均排名。这里的@10 是返回的前10个结果中进行评估。MRR 越高表明检索结果的质量更高。

$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}$$

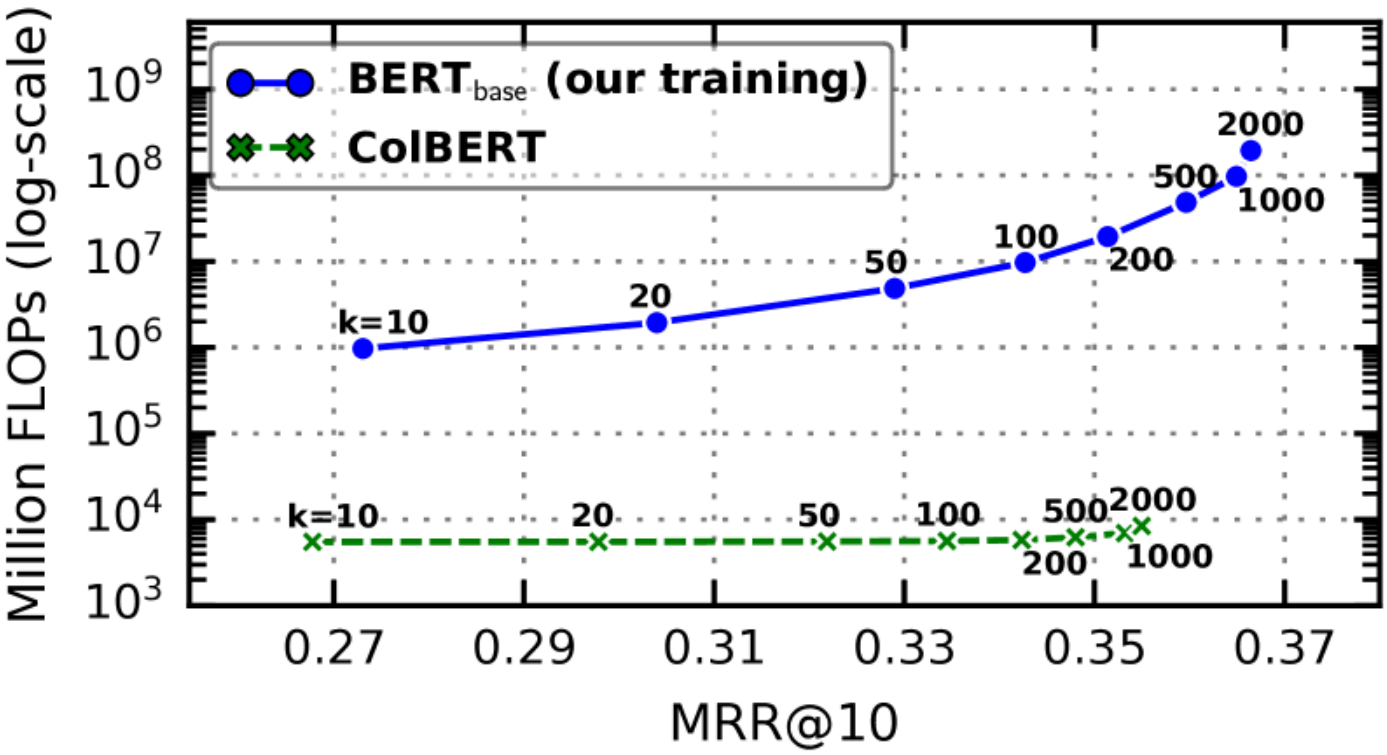
其中:

- $|Q|$ 表示查询的总数量。
- rank_i 表示第 i 个查询中第一个正确答案的排名。
- FLOPS：是 FLOP 的每秒执行次数，用来表示计算设备的性能。FLOPS越高说明模型的计算复杂度越大，用户等待回复的时间就越长。(FLOP是指单次浮点运算，执行一次任务所需的浮点数运算的总次数)

$$\text{FLOP} = h \times w \times k \times k \times c$$

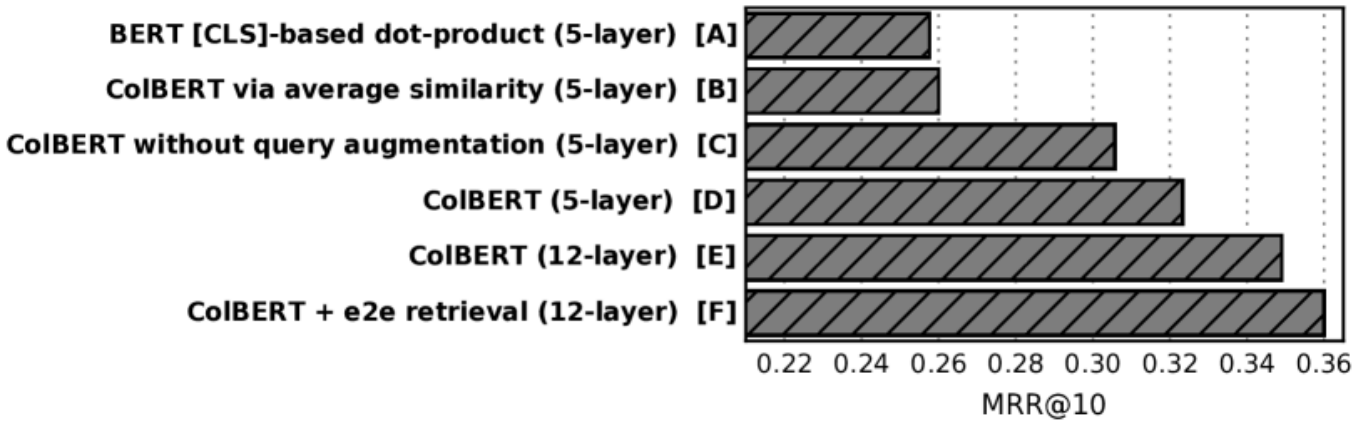
其中：

- $k \times k$ 是卷积核大小。
- h, w 为输入特征图的长宽，c 为通道。



在这里，k 是排名。

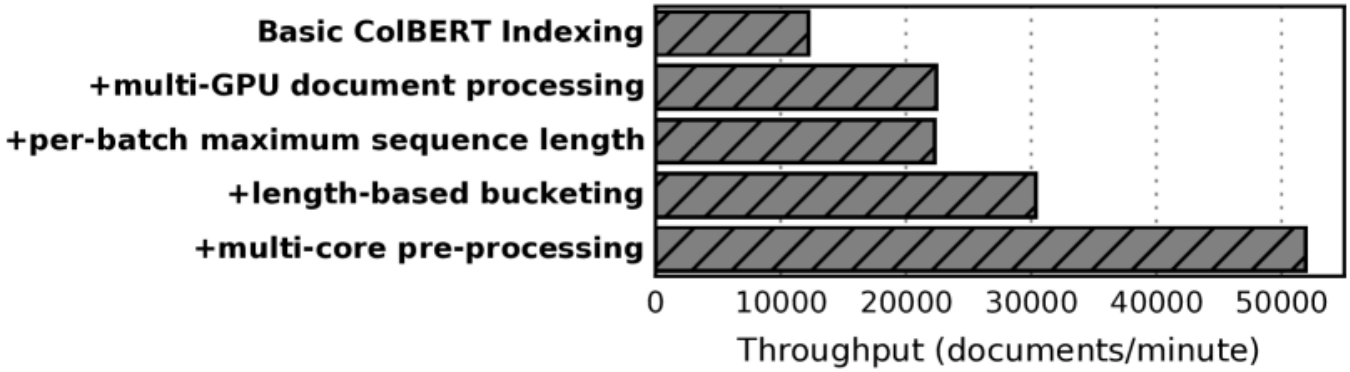
7. **消融研究**



- [A] 后期交互的细颗粒度交互使得模型性能增强。
- [B] 没有使用基于MaxSim的后期交互，而是使用平均相似度使得模型性能降低。

- [C] 同理，C是没有使用查询增强。
- [D] 保留12层中的5层 BERT
- [E] 正常情况模型
- [F] 添加了端到端设置(跳过了粗排阶段，直接生成最终结果，虽然效果较好，但计算开销大)

8. **索引吞吐量**



每分钟处理的文档数量（吞吐量）
从上面的图可知，

- ColBERT 索引的基础设置，即不进行任何额外的优化
- 增加了多 GPU 进行文档处理的支持
- 针对每个批次动态调整最大序列长度
- 基于文档长度进行分桶（bucketing）。将长度相似的文档放在同一个批次中，这样可以进一步减少填充操作
- 使用多核(多个CPU的核心)进行预处理。

注："+" 表示是在前一个优化策略的基础上叠加。

结论

本研究提出了 ColBERT，一种基于 BERT 的高效有效的文本检索模型，通过延迟文档和查询之间的交互，实现了文档的离线编码和快速的在线检索。它能够在保持深度语言模型表现力的同时，显著降低计算成本，使得模型能够在不牺牲效果的前提下，大幅度提高检索速度。同时通过实验表明, ColBERT 的表现是优于非 BERT 基线模型。

1. 对于DCG，模型返回5个文档，对每个文档进行相关性打分，比如说，一组检索结果前5个的相关性评分为 [3, 2, 5, 4, 1]，但是理想的是[5, 4, 3, 2, 1]。在理想情况下，系统会尝试将高相关性的文档排在前面，确保用户先看到最相关的内容，但是实际排序结果可能与理想排序存在差异(系统评分误

差,排序算法的局限性),这个指标在于不仅看分数，还看排序位置，如果高分数的文档位置靠后，会导致NDCG 下降，说明模型的效果不是很好。 ↩