Retrieval is Accurate Generation

太长不看版

本研究提出了一种创新的文本生成方法，该方法通过从支持文档中选择上下文感知的短语来生成文本，而非仅从固定词汇表中选择。

主要挑战：确定训练预言器→通过语言启发式方法初始化预言器，并采用迭代自我强化策略来引导和完善预言器。

实验结果显示，我们的模型在多个知识密集型任务上超越了标准语言模型，例如在OpenbookQA上的准确率显著提高，以及在开放式文本生成中的MAUVE得分大幅提升。即使是思路相近的CoG表现也不如我们的模型出色。模型在检索增强型基线中也展现了最佳性能和最低延迟。

而且在特定领域（医学）的表现也相当出色。

我们的贡献包括引入了一种新的语言建模方法，提出了一种新颖的训练预言器构建方法，并通过一系列实验验证了模型的有效性。

摘要

标准语言模型通过从固定的、有限的、独立的词汇表中选择标记来生成文本。我们介绍了一种新颖的方法，该方法从一系列支持文档（supporting documents）中选择上下文感知(context-aware)的短语。

这种范式转变的最大挑战之一是确定训练预测器(the training oracles)，因为一个文本字符串可以以多种方式分割，每个分割都可以从众多可能的文档中检索到。

为了解决这个问题，我们提出使用语言启发式方法(linguistic heuristics)来初始化训练预测器，更重要的是，通过迭代自我强化(iterative self-reinforcement)来引导预测器。

广泛的实验表明，我们的模型不仅在各种知识密集型任务上胜过标准语言模型，而且在开放式文本生成中展示了改进的生成质量。例如，与标准语言模型相比，我们的模型在OpenbookQA上的准确率从23.47%提高到36.27%，并在开放式文本生成中的MAUVE得分从42.61%提高到81.58%。

值得注意的是，我们的模型在几个检索增强型基线中也实现了最佳性能和最低延迟。总之，我们断言检索是更准确的生成，并希望我们的工作能鼓励对这个新范式转变的进一步研究。

引言

标准语言模型（LMs）将文本生成过程分解为连续的标记预测。每个标记是从固定的、有限的、独立的词汇表中选出的单词（或子词）。为了使生成更具可解释性并加速推理速度，Lan等人提出了一种名为CoG的方法，该方法从相似上下文中检索短语。

值得注意的是，与其他检索增强型生成框架一样，CoG仍然采用两阶段流水线，即文档检索后跟基于检索的短语提取。最终的性能受到第一阶段返回的质量与数量的限制。

在本文中，我们提出了一种新的范式，完全摆脱了对文档检索的依赖。据我们所知，我们的工作是首个通过直接短语检索进行文本生成的研究。

采用这种新方法的一个核心挑战是构建训练预测器。训练预测器是一个函数，它将文本字符串映射到创建训练示例的动作序列。对于给定的文本，存在许多不同的分割方式，每个潜在的短语都可以从大量的文档中检索到。为了更好地对齐生成过程和支持文档，我们引入了一种两步法：首先，我们利用语言驱动的启发式方法来初始化训练预测器。其次，我们通过迭代自我强化实施一个引导机制，逐步完善预测器。

与Lan等人仅评估开放式文本生成中的生成流畅性不同，我们对一系列知识密集型任务进行了全面严格的评估，例如开放域问答。我们提出的模型展现了卓越的零样本性能，胜过了基线方法。例如，在OpenbookQA数据集上，我们的模型显著提高了基础LM的性能，准确率从23.47%提高到36.27%（表1）。我们的模型在开放式文本生成中也展示了改进的质量，如MAUVE得分提高了38.97%（表4）。此外，当我们切换到扩大的（表2）或特定领域的（表3）短语表时，它甚至展示了更好的性能，而无需任何进一步的训练。此外，我们的模型在检索增强型基线中实现了最快的生成速度（表4）。我们相信，我们的研究可以激发未来的研究，构建更高效、更准确的LM，利用基于检索的方法的力量。

总之，本文的贡献可以总结如下：

我们引入了一种新的语言建模方法，专注于直接从一组支持文档中选择上下文感知的短语。

我们提出了一种新颖的方法，通过语言驱动的启发式方法和迭代自我强化引导，将文本生成分解为顺序的下一个短语检索。

我们在各种下游任务上验证了我们模型的有效性，包括开放域和特定领域的问题回答，以及开放式文本生成，突出了与标准LM和几个检索增强型基线相比的显著改进。

生成和检索的统一视图

标准语言模型将序列x的生成概率分解成一系列的条件概率，因此这种生成通过在已经生成的序列基础上重复预测下一个token，下一个token的预测概率为：

其中，是前缀的的向量表示, 是token x的向量表示，V代表token词汇表。通过上面的符号，我们可以看到标准LMs可以看作是连接不同前缀和token的双编码器匹配网络。如图一左侧所示，当目标编码器仅仅是一个token嵌入层的时候，源编码器被多层神经网络应用（如Transformer）。由此可见，双编码器网络的设计存在严重的不平衡。源编码器部分比目标编码器部分复杂得多。

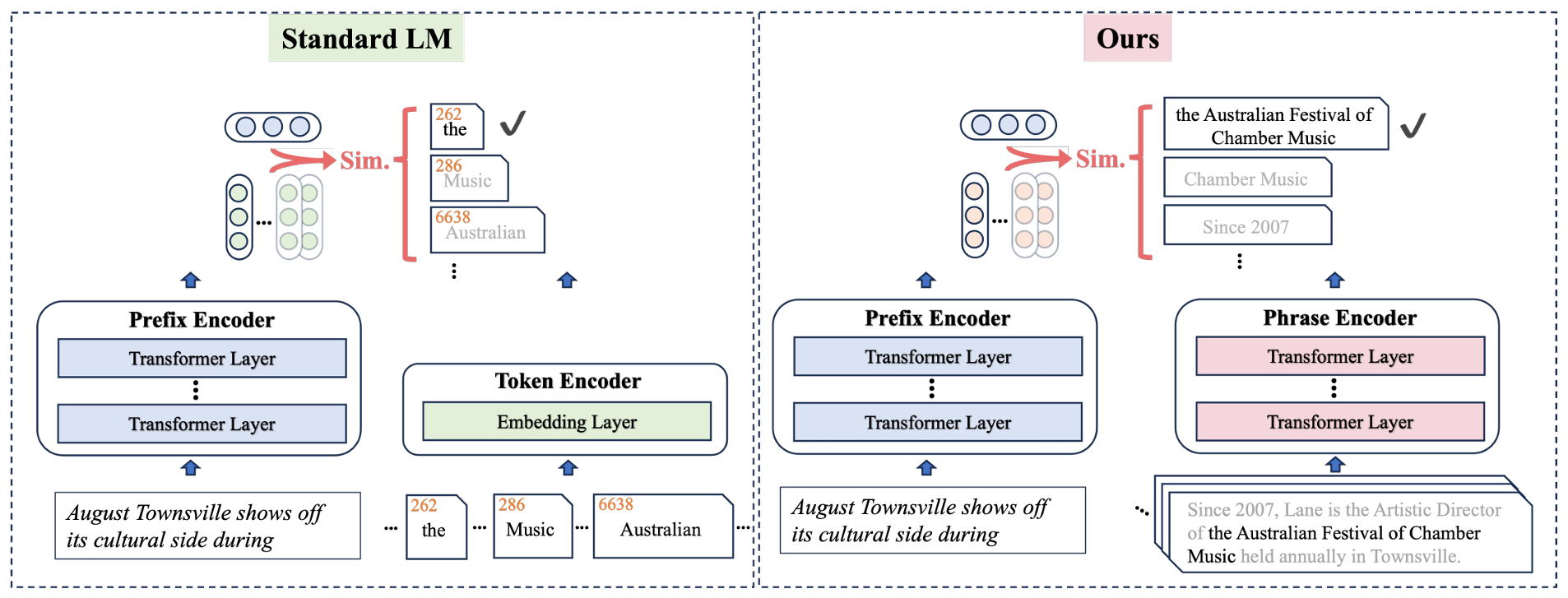


图 1: 我们的方法是与标准语言模型相比较的双编码器匹配网络，连接源前缀和目标续写。在目标侧，标准语言模型使用来自固定、有限、独立的词汇表的即时嵌入层为目标标记。相比之下，我们的方法使用来自可编辑、可扩展和上下文化的短语表的目标短语的表达短语编码器。

然后，作者提到了最近提出的检索增强型语言模型CoG，它除了选择标记外，还允许从支持文档集合中检索短语（即可变长度的n-grams）。作者认为，CoG增强了传统LMs的目标侧，通过扩大候选池来包含可变长度的短语，并使目标编码器不仅考虑候选本身，还考虑它们的上下文。

然而，从大规模语料库中搜索短语是资源密集型的。因此，CoG采用了两阶段搜索策略：首先检索相关文档以减少短语选择的搜索空间。为了构建训练预测器，CoG使用前向最大匹配算法从检索到的文档中找到最长匹配的短语。尽管CoG取得了有希望的结果，但它不能保证为短语检索提供全局最优解，并且高度依赖于外部文档检索工具。与此相反，作者提出了一种新的范式，通过直接短语检索来生成文本。

提出的方法

# 3.1 概览

研究的目标是通过短语检索来增强LMs的可解释性和事实性，从而提高文本生成的质量。

1.作者的方法通过将短语的语义与其上下文相结合，提高了推理时的区分性表示。

2.每个检索到的短语都可以追溯到原始文档，增强了输出的可追溯性。

为了将给定的前缀与一组可变长度的短语相连接，我们的模型遵循第2节中描述的双编码器结构，但强调与标准语言模型相比的平衡设计，后者严重偏向源侧（见图1）。具体来说，源编码器()是一个多层神经网络（例如，Transformer），与通常情况一样。目标编码器()也是一个多层神经网络，用于学习支持文档中短语的上下文感知表示。

与标准语言模型类似，我们使用点积作为匹配度量。在推理过程中，我们可以使用高效的最大内积搜索（MIPS）算法从大量候选短语中检索。整个框架在图1中描绘。接下来的问题是如何训练我们的模型。

# 3.2 训练预测器

我们将文本生成分解为一系列下一个短语的检索任务。形式上，每一步都取当前前缀作为其状态，一个预测器策略将状态映射到一个动作 (p) → (f, s)，其中f是后续短语，s是支持文档中短语f的一个副本。

如图2所示，从原始语料库创建这样的三元组(p, f, s)面临两个挑战。

首先，由于一个续写可以以多种方式分割，短语f的边界不明确。

其次，每个短语s的来源不明确，因为一个短语可能在大量文档中多次出现。

另一方面，给定文本的生成路径的多样性也表明，训练预测器对于我们模型的最优和快速收敛至关重要。

为了解决上述问题，我们首先提出了一套基于语言学启发式的启发式方法来初始化训练预测器（第3.2.1节），然后描述了我们如何允许模型以自我强化的方式完善其生成路径（第3.2.2节）。

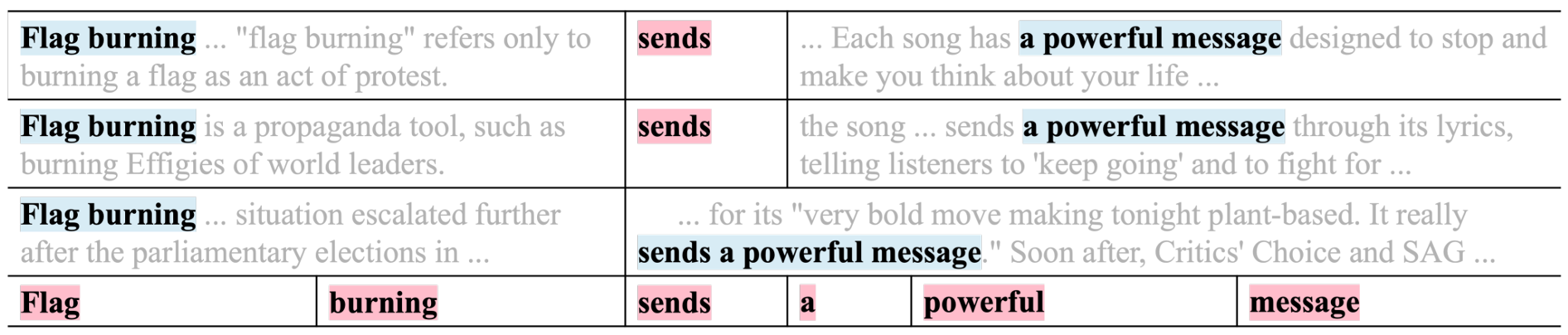


图 2: 句子 “Flag burning sends a powerful message” 的四种可能生成路径。用蓝色（红色）突出显示的内容是从支持文档（来自标记词汇表）检索到的短语。标准LM可以被视为只考虑底部的生成路径。

3.2.1 语言学启发式方法

我们通过以下原则开始设计训练预测器。

**句法结构(Syntactic Structure)** 受到语言句法结构及其对语言生成的影响的启发，我们将短语限制为对应于句法解析树中成分单元的连续单词序列。这种方法确保每个短语具有相对完整和明确的意义，同时避免可能导致语义歧义或无意义组合的任意单词组合。

**分布稀疏性(Distributional Sparsity)** 包含高频短语显著增加了候选池的大小。这是因为我们将不同上下文中词汇相同的短语视为池中的不同条目。因此，一个单一的高频短语可能引入数万甚至数百万的条目。在我们的维基百科分析中，我们发现仅消除前1%的高频短语就可以减少总条目数的50%。然而，这些高频短语，如“as well as”，通常缺乏具体意义。它们的包含可能导致训练不平衡，从而可能对模型的整体性能产生不利影响。关于极低频的短语，我们认为它们是罕见用法，实际用途有限。包含它们将显著增加训练的复杂性。因此，我们也选择排除它们。

**语义相似性(Semantic Similarity)** 尽管一个短语的词汇相同的副本可以在不同的地方找到，但考虑到多义性是至关重要的，因为词汇相同的短语可以根据其上下文表现出不同的含义。此外，即使词汇相同的短语共享相似的含义，不同的上下文也可能出现微妙的细微差别，这要求在选择最合适的匹配时进行彻底的语义相似性评估。

具体来说，我们首先运行Stanford Parser来从训练数据中提取成分。然后，我们根据以下标准过滤这些成分：（1）移除带有标签如WHADJP、WHADVP的琐碎成分；（2）排除太短（<2个单词）或太长（>10个单词）的成分；（3）丢弃具有过高或过低的逆文档频率（IDF）值的成分。值得注意的是，我们对较长的成分应用了更宽松的IDF阈值。接下来，我们将词汇相同的短语分组，并使用BM25和现成的短语编码器计算成对的语义相似性。因此，我们可以根据分数为每个前缀识别出最合适的下一个短语。。

3.2.2 迭代自我强化

上述启发式方法确定的生成路径是模型无关的，可能是嘈杂(noisy)和次优(sub-optimal)的。为了进一步提高性能，我们允许模型根据其已获得的能力调整自己的生成路径。也就是说，从模仿预测器过渡到加强自己的偏好。具体来说，我们提出了一个自举算法来迭代调整目标短语。对于每个前缀p，我们首先让模型使用其当前策略从整个候选池中检索k个最佳短语。然后，我们从这些k个短语中选择具有最高语义匹配得分的有效短语作为新目标。如果没有找到这样的短语，即k个最佳短语都与真实续写不匹配，我们保留前一个目标。

3.3 训练目标

我们使用InfoNCE损失优化我们的模型，为每个三元组引入了一个负短语集。

负短语集的构建详见下文。为了保留在标记级别生成的能力，我们还使用标准的下一个标记预测损失Lt（Lan et al., 2023）训练我们的模型。训练目标定义为。

**负面采样(Negative Sampling)。**我们采用了两种类型的负面样本来提高模型区分短语的能力：

1. 批内负面样本：我们将同一训练批次中的所有其他候选短语视为这种类型的负面样本。这些负面样本帮助模型在不产生显著成本的情况下，大规模学习更具辨别性的表示。

2. 困难负面样本：回想在第β.2.2节中，我们定期通过检索每个前缀的前k个候选短语来更新生成目标。在这些k个短语中，尽管有一个可能被选为新的生成目标，但其余的短语可以作为强大的负面样本，因为它们很可能会让模型混淆。

需要注意的是，上述负面样本可能包含假的负面（样本），即那些没有被选为目标但仍然是有效后续的短语。为了最小化风险，我们移除了所有构成真实续写前缀的短语。

# **3.4 模型**

**前缀编码器(Prefix Encoder)** 我们将前缀视为一系列标记，其中先前预测的短语被分割成标记。这个标记序列使用带有因果注意的标准Transformer架构进行编码。通过序列中最后一个标记的最后一层表示的线性投影获得前缀表示。

**短语编码器(Phrase Encoder)** 我们使用深度双向Transformer为支持文档生成上下文化的标记表示。短语的表示是通过连接其第一个和最后一个标记的表示，然后将连接的表示投影到与前缀表示相同的维度获得的。为了保留使用单个标记组成输出的能力，我们还将标记词汇表添加到我们的短语表中。这些独立的标记可以被视为特殊短语，它们的表现通过LM的标准嵌入层获得。

实验设置

# 4.1 实现细节

我们在 MiniPile2的训练集上训练我们的模型，并使用2023年3月1日的英文维基百科转储作为支持文档。具体来说，我们将每篇维基百科文章分割成多个最多128个单词的不相交文本块作为文档，共得到29,488,431个文档。我们的短语索引大小为137,101,097。我们使用GPT-2和DensePhrases4来初始化前缀编码器和短语编码器。

为了效率，我们仅对前缀编码器进行微调。这样可以避免重新计算与更新短语编码器相关的短语嵌入所带来的计算负担。在通过自我强化修订训练预测器时，我们为每个前缀检索前k=128个短语。

# 4.2 推理细节

在推理过程中，我们使用FAISS，一个用于向量相似性搜索和聚类的库，来进行高效的检索。

**续写生成(Continuation Generation)** 对于文本生成，我们直接从整个短语表（包括上下文感知短语和独立标记）中检索前k个候选项。然后，我们对这些候选项的匹配分数应用softmax函数，创建下一个短语的概率分布，并使用top-p采样来选择下一个短语。在所有实验中，我们将k设置为128（有关k的分析请参见附录G中的表7），p设置为0.95。为了控制短语检索的比例，我们过滤掉概率低于阈值的短语。除非另有说明，否则阈值设置为ϕ=0.4。

**可能性估计(Likelihood Estimation)** 为了计算给定文本的可能性，我们通过求和所有可能的生成路径来近似可能性。例如，给定句子“The Moon rises”，可能存在以下生成路径：

1. The→moon→rises; (2) The moon→rises; (3) The moon rises。

每条路径的概率是沿着该路径的所有短语（标记）的概率的乘积。例如，路径(2)的概率是通过 p(rises|The moon) · p(The moon) 来计算的。每一步的概率是按照我们为续写生成构建下一个短语概率分布的相同方式获得的。请注意，所有可能路径的总和可以使用动态规划高效计算，其时间复杂度为O(n²)，其中n代表文本中的标记数量。

# 4.3 基线

我们在zero-shot setting下将所提出的方法与标准语言模型（LM）进行比较，并绘制以下最新的检索增强方法作为基线：

Base LM 是使用Transformer架构的标准标记级语言模型。我们对预训练的GPT-2进行微调。

kNN-LM是一种检索增强的LM，它将基础LM的下一个标记分布与k-最近邻（kNN）模型插值。

RETRO是一种检索增强的LM，它结合了预训练的文档检索器、文档编码器和交叉注意力机制。

CoG是另一种检索增强的LM，它采用两阶段搜索流水线。它首先检索语义相关的文档，然后将其中的所有必要n-gram作为候选短语考虑。

5 实验

# 5.1 知识密集型任务

## 5.1.1 数据集

我们采用了五个知识密集型数据集，包括三个开放领域问答数据集：OpenbookQA (Mihaylov et al., 2018)

ARC-Challenge (Clark et al., 2018)

TruthfulQA (Lin et al., 2022)

以及两个特定领域的（医学）数据集：

MedMCQA (Pal et al., 2022)

MedUSMILE (Jin et al., 2021)

这些数据集的详细信息可以在附录C中找到。

为了量化模型性能，我们采用了分类选择的方法来衡量模型的表现。这种方法涉及向模型展示一系列选项，并计算每个选项是正确回答的可能性。选择概率最高的选项作为模型的预测。然后，我们报告模型预测的准确性。

## 5.1.2 结果

我们在多个设置中将我们的方法与基线模型在知识密集型任务上进行了比较。

**主要结果** 如表1所示，我们的模型在所有数据集上始终优于各种基线模型。与基础语言模型（Base LM）相比，我们的模型将TruthfulQA和OpenBookQA数据集的准确率从29.73%提高到34.27%和从23.47%提高到36.27%。当我们从我们的模型中移除短语检索功能，仅使用独立标记（Ours w/o phrase）时，性能显著下降，这证明了在我们的方法中结合短语检索的有效性。

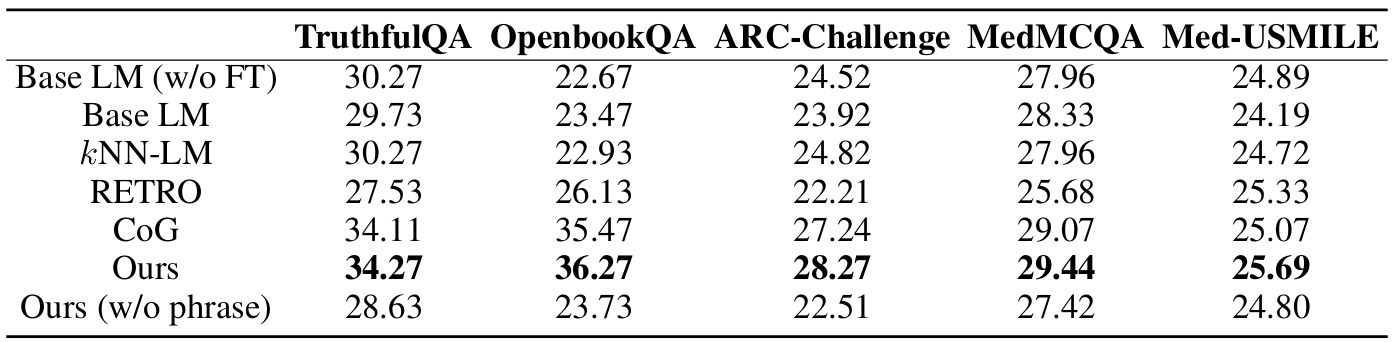


表1：在知识密集型任务上的实验结果。Ours (w/o phrase): 限制模型只使用独立标记而不检索上下文感知短语的我们模型的变体。

请注意，表中展示的模型是从预训练的语言模型（LMs）初始化的。为了分析预训练模型在我们框架中的作用，我们从头开始，使用随机初始化训练所有模型。结果显示在附录G的表中，我们的模型在所有数据集上都优于基线模型。例如，我们的模型在OpenbookQA上相对于基础语言模型实现了12.8%的绝对提升，这表明我们的训练框架并不严重依赖于预训练模型。为了阐明短语检索在知识密集型任务中的作用，我们深入研究了附录D中描述的一个案例研究。

**扩大短语索引** 回顾我们在第3.2.1节中排除了过高或过低IDF值的短语这一策略。这种策略不仅稳定了训练过程，还提高了训练效率。然而，最初被过滤掉的短语可以被重新利用，以无需训练的方式扩展我们的短语索引。这种扩展的短语索引，现在是原始大小的三倍，强调了我们方法的可扩展性。

如表2所示，这种扩展提高了我们模型的性能，例如在TruthfulQA上的准确率提高了5.32%。这不仅突出了我们模型潜在的泛化到未见短语和文档的能力，还强调了其即插即用的特性，能够在无需重新训练的情况下适应更大的短语表。

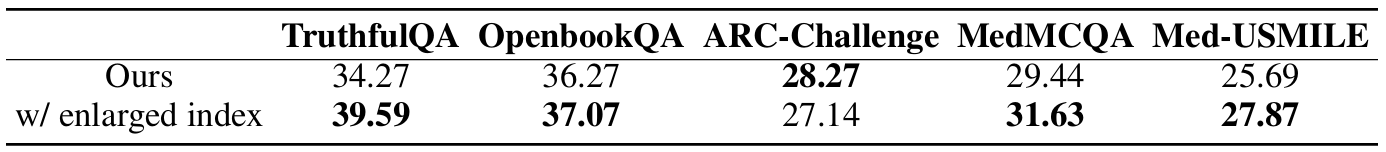


表2：使用扩大的短语索引的我们模型的结果。

**领域适应** 短语索引的即插即用(plug-and-play)属性进一步激励我们使用特定领域的索引来处理医学领域的问答任务，而无需任何特定领域的训练。

为此，我们通过从一个小型医学领域文本集合中提取短语，构建了一个包含300万个短语的索引。为了公平比较，我们也在它上面对基础语言模型进行了微调。如表3所示，尽管与原始的维基百科索引相比，索引大小大幅减少（300万对1370万），但我们的模型在两个医学QA数据集上表现出更好的性能。

这一结果强调了我们的模型通过利用特定领域、精心策划的短语索引，以无需训练的方式提高其在特定领域的性能的能力。

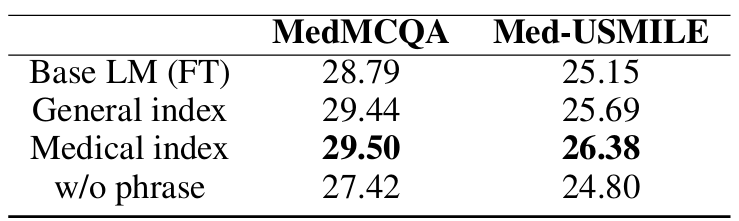


表3：在医学数据集上的结果。

# 5.2 开放式文本生成

我们在MiniPile测试集上进行开放式文本生成实验。对于测试集中的每个文档，我们采用前128个标记作为前缀。基线模型和我们的模型都需要基于相同的前缀生成长度为128个标记的文本续写。

## 5.2.1 评估指标

遵循之前的工作，我们使用三个自动评估指标来衡量生成文本的质量：

1. MAUVE通过估计内容的平均效用，来捕捉生成文本的整体有用性；
2. 连贯性衡量生成文本的逻辑一致性和流畅性，确保输出结构良好且易于理解；
3. 多样性评估生成内容的多样性，促进生成独特和创造性的文本。

我们将MAUVE和多样性报告为百分比（%）。这些指标的详细信息可以在附录E中找到。我们还测量了一个模型在给定128个标记的前缀的情况下解码一个由128个标记组成的续写的平均时间成本，称为延迟。

## 5.2.2 结果

如表4所示，我们的模型在所有模型中获得了最高的MAUVE分数，展示了生成文本的高质量。其他检索增强方法在MAUVE分数上表现不如基础语言模型，这是由于文本退化，与之前工作的发现一致。我们的模型在连贯性和多样性之间也表现出了强烈的平衡。我们的模型的连贯性得分为3.25，除了CoG之外，超过了大多数基线模型。

然而，我们发现CoG经常生成词汇上相似、无意义的句子，这反映在其低多样性得分55.04%中。与此同时，我们模型的多样性得分为76.26%，虽然略低于一些基线模型，但这些模型经常生成不连贯的句子，如它们较低的连贯性得分所反映的那样。

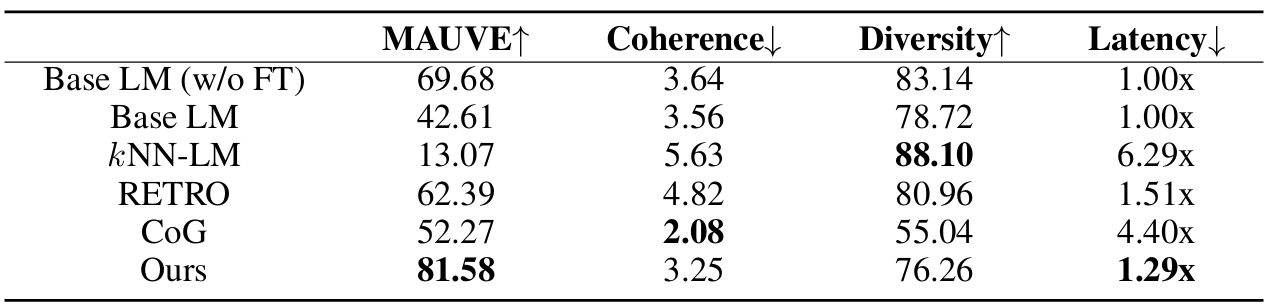


表4：开放式文本生成的结果。

**人类评估** 为了获得进一步的见解，我们随机抽取了100个案例，并从四个角度评估了基础语言模型、未经过微调的基础语言模型（w/o FT）和我们的模型的结果：流畅性、连贯性、信息量和语法。每个方面都在1到4的Likert量表上进行评分（1代表“差”，2代表“一般”，3被认为是“好”，4表示“非常好”）。我们在表5中报告了平均分数。正如我们所看到的，我们的方法在所有四个类别中都优于基础语言模型，特别是在连贯性和信息量方面。这表明我们的基于短语检索的模型更擅长遵循前面的上下文，并提供更有信息量的内容。至于基础语言模型与未经过微调的基础语言模型（w/o FT）相比得分较低的原因，我们发现主要是由于格式问题。进一步的分析可以在附录F中找到。

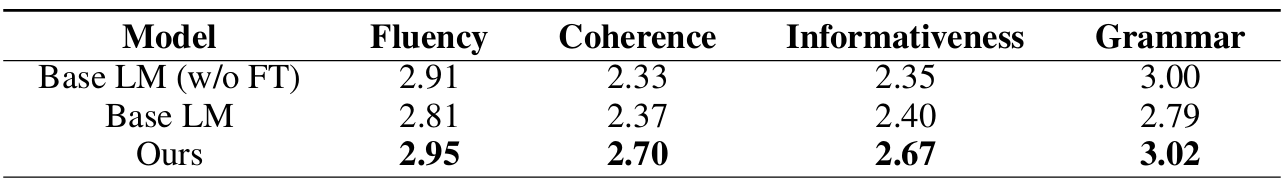


表5：人类评估的结果。

**生成速度** 现在我们来讨论不同模型的生成延迟。在表4中，我们报告了相对延迟，以基础语言模型为基准。kNN-LM由于需要将基础语言模型的标记分布与使用其数据存储计算的另一个分布进行插值，因此产生了最高的延迟(cost)。CoG模型也表现出显著的延迟(overhead)，因为它涉及到从检索到的文档中提取所有n-gram，对标记和所有n-gram应用softmax，并从结果概率分布中进行采样。RETRO模型虽然比前两个模型更快，但仍然需要时间来应用检索到的文本块的表示进行注意力计算。我们的方法在生成速度上脱颖而出，因为它直接检索和利用短语。

**自我强化的效果** 对自我强化（SR）机制的影响进行的消融研究揭示了我们模型性能的重要见解。在知识密集型任务的情况下，我们没有观察到SR对我们模型性能的显著影响。

然而，对于开放式文本生成的情况则有所不同。表6显示，经过SR训练的模型在多轮中在MAUVE得分上表现出了显著的改进，这表明SR在提高文本生成质量方面的重要性。在第二轮之后，我们没有观察到额外的SR迭代轮次带来明显的改进，这表明模型已经收敛到其最优状态。

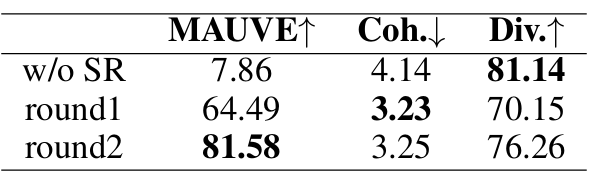


表6：在自我强化效果上的消融研究

7 结论

我们提出了一种新颖的基于检索的文本生成方法，使用上下文感知的短语检索。我们的方法通过基于启发式的初始化和迭代自我强化来解决构建训练预测器的主要挑战。

在知识密集型任务和开放式文本生成任务上的实验表明，我们提出的方法优于标准语言模型和最新的检索增强方法。

此外，我们的模型在使用扩大或更小的特定领域索引时展现出卓越的性能，并且与其它检索增强基线相比，实现了最低的生成延迟。

这项工作为NLP研究社区贡献了一种通过检索实现更准确生成的范式转变。随着我们继续探索和完善这一范式，我们邀请读者考虑我们当前工作的局限性，以充分理解未来研究的范围。