arXiv:2304.04171v1 [cs.IR] 9 Apr 2023

学习用于生成式检索的分词

Weiwei Sun[1], **Lingyong Yan**[2], **ZhengChen**[1], **Shuaiqiang Wang**[2], **Haichao Zhu**[2]
Pengjie Ren[1], **Zhumin Chen**[1], **Dawei Yin**[2],**Maarten de Rijke**[3], **Zhaochun Ren**[1]山东大学 [2]百度公司 [3]阿姆斯特丹大学

{sunnweiwei,lingyongy,shqiang.wang}@gmail.com,{202000130223,renpengjie,chenzhumin,zhaochun.ren}@sdu.edu.cn
hczhu@ir.hit.edu.cn,yindawei@acm.org,m.derijke@uva.nl

摘要

传统的文档检索技术主要基于索引-检索范式。要以端到端的方式优化基于这一范式的流程颇具挑战。作为一种替代方案，生成式检索将文档表示为标识符（文档标识符），并通过生成文档标识符来检索文档，从而实现文档检索任务的端到端建模。然而，如何定义文档标识符仍是一个开放问题。目前，定义文档标识符的任务采用基于固定规则的文档标识符，例如文档标题或 BERT 嵌入的聚类结果，但这些方法往往无法捕获文档的完整语义信息。

我们提出了 GenRet，这是一种文档标记化学习方法，旨在解决生成式检索中文档标识符定义难题。GenRet 采用离散自编码方法，将文档分词为简短的离散表示（即文档标识符）。GenRet 包含三个组件：(i) 一个分词模型，用于为文档生成文档标识符；(ii) 一个重建模型，用于学习根据文档标识符重建文档；(iii) 一个序列到序列检索模型，用于针对指定查询直接生成相关文档标识符。通过采用自编码框架，GenRet 以完全端到端的方式学习语义文档标识符，并可将生成的文档标识符重建为原始文档，以确保其语义得以保留。我们还设计了一种渐进式训练方案，以捕捉文档标识符的自回归特性并稳定训练过程。

我们在 NQ320K、MSMARCO 和 BEIR 数据集上进行了实验，以评估 GenRet 的有效性。GenRet 在 NQ320K 数据集上取得了新的最先进水平。尤其是，与生成式检索基线相比，GenRet 在未见文档上取得了显著提升（例如，在 R@1 方面相对提升至少 +14%）。此外，GenRet 可以
与先前基于规则的分词方法相比，GenRet 能更好地表示和检索训练阶段未见过的文档。GenRet 在 MS MARCO 和 BEIR 上也优于具有可比性的基线方法，展现了该方法的泛化能力。[1]

1 引言

文档检索在网页搜索应用以及各种下游知识密集型任务中发挥着至关重要的作用，例如问答和对话系统，因为其目标是找出相关文档，以满足用户的查询。大多数传统的文档检索方法采用稀疏检索方法，依赖于构建倒排索引，并使用词项匹配指标，例如 TF-IDF [38], 查询似然 [22], 或 BM25 [39]。然而，词项匹配指标往往会受到词汇不匹配的影响 [24]。

近期，预训练语言模型（LM）的进展推动稠密检索（DR）模型取得了重大进展 [13,19, 31, 47]。如图1 （a）所示，DR 方法通过双编码器学习查询和文档的稠密表示，随后使用最大内积搜索（MIPS）检索相关文档 [18, 19]。DR 方法能够解决词汇不匹配问题，并在各种检索任务上取得最先进的性能 [25, 29]。

尽管取得了成功，DR 方法面临两大局限 [5,28]: （i）DR模型采用带有固定搜索过程（MIPS）的索引检索流程，因此难以以端到端的方式联合优化所有模块；以及（ii）学习策略（例如，对比学习 [19]）通常与预训练目标不一致，例如下一词元预测 [3]，这使得难以利用预训练语言模型中的知识 [1]。

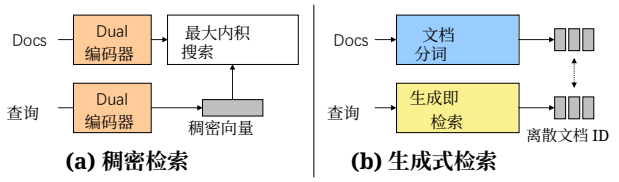


图 1：两种类型的文档检索模型：(a)稠密检索 **将查询和文档编码为稠密向量，并通过MIPS 检索文档；(b)**生成式检索**将文档标记化为文档标识符，并以自回归方式生成文档标识符作为检索结果。**

生成式检索。 近年来，生成式检索 已成为文档检索的一种新范式 [1, 5, 42, 46, 52, 53]。如图 1 （b）所示，生成式检索模型使用生成式语言模型，针对给定查询直接生成一个排序后的文档标识符（文档标识符）列表。具体而言，生成式检索模型包含两个主要步骤：(i) 文档分词，将语料库中的每篇文档分词为由离散字符组成的序列，即文档标识符；(ii)生成即检索，针对给定查询，通过自回归解码输出相关文档的文档标识符。与 DR 不同，生成式范式为文档检索任务提供了一种端到端的解决方案 [42]。它还为更好地利用近期大型语言模型的能力提供了一种颇具前景的方法 [1, 46]。

学习对文档进行分词。文档分词在生成式检索中发挥着至关重要的作用，因为它定义了文档在语义空间中的分布方式 [42]。如何定义文档标识符仍是一个悬而未决的问题。大多数先前的

[1]预印本。工作仍在进行中。

生成式方法倾向于采用基于规则的文档分词器，例如生成标题或 URL [5, 52], , 或对现成的文档嵌入进行聚类 [42, 46]。然而，这类基于规则的方法通常缺乏通用性，难以良好泛化。具体而言，分词结果可能有助于检索训练期间见过的文档，但对新文档或分布外文档的泛化能力较差 [23, 27]。

所提出的方法。 为解决上述问题，本文提出了 GenRet，这是一种文档标记化学习框架，采用离散自动编码方案，学习将文档分词为语义文档标识符。具体而言，GenRet 由一个基于序列到序列的共享文档分词模型、一个检索模型和一个文档重建模型组成。在所提出的自动编码学习方案中，分词模型学习将文档转换为离散文档标识符，随后由重建模型利用这些文档标识符重建原始文档。生成式检索模型经过训练，能够针对给定查询以自回归方式生成文档标识符。上述三个模型以端到端方式进行优化，从而实现无缝集成。

在使用自动编码优化生成式检索模型时，我们进一步发现了两个挑战：（i）具有自回归特性的文档标识符；（ii）文档标识符的多样性。为应对第一个挑战并稳定 GenRet 的训练过程，我们设计了一种渐进式训练方案。该训练方案通过固定已优化的前缀文档标识符 $z_{<t}$，实现模型的稳定训练。为优化每一步的文档标识符，我们采用了三种提出的损失函数：（i）用于根据生成的文档标识符预测文档的重构损失；（ii）用于约束文档标识符并避免遗忘的承诺损失；（iii）用于以端到端方式优化检索性能的检索损失。为应对第二个挑战，我们提出了一种参数初始化策略，并基于多样化聚类 技术重新分配文档标识符，以提高生成文档标识符的多样性。

实验。 我们在三个知名的文档检索基准数据集上开展实验：(i) NQ320K，包含维基百科的一个子集 [21, 42]；(ii) MS MARCO，包含与一组搜索查询相关的网页 [4, 52]；以及 (iii) BEIR，包含用于分布外评估的异构检索任务 [43]。我们的实验结果表明，与最先进的稠密检索或生成式检索模型相比，GenRet 取得了更优的检索性能。在 NQ320K 上的实验显示，GenRet 在该数据集上建立了新的最先进水平；与最佳基线方法相比，其在未见测试集上的相对提升达到 +14%。在 MS MARCO 和六个 BEIR 数据集上的实验还表明，GenRet 显著优于现有的生成式方法，并且与最佳稠密检索模型相比取得了具有竞争力的结果。对新文档进行检索的实验、分析实验和效率分析证实了所提出模型的有效性。

贡献。 本文作出以下贡献：(i) 我们提出了GenRet，这是一种将文档表示为语义离散文档 ID 的生成式检索模型。据我们所知，这是首个用于文档检索的分词学习方法。(ii) 我们提出了一种自编码方法，其中由我们的分词模型生成的文档标识符通过一个重建模型，以确保文档标识符捕获文档的语义信息。(iii) 我们设计了一种渐进式训练方案，用于建模文档标识符的自回归特性并稳定训练过程。(iv) 实验结果表明，与生成式检索基线相比，GenRet 取得了显著提升，尤其是在未见文档上。

2 预备知识

文档检索任务可以形式化为从文档集合D中为搜索查询 q检索相关文档 d。每个文档 $d \in$D 都是纯文本文档，由一系列词元组成，记作 $d = \{d_1, \ldots, d_{|d|}\}$，其中 $|d|$表示文档中的词元总数。

与稠密检索方法不同，后者根据每个文档相对于给定查询 q的相关性得分返回最相关的文档，生成式检索 模型旨在使用生成式模型直接为给定查询 q 生成文档。

文档分词。 对于生成式检索 模型，直接生成通常长度较长的原始文档往往具有挑战性且计算效率低下。因此，大多数现有方法都依赖于名为文档分词的技术，该技术将文档 $d = \{d_1, \ldots\ldots, d_{|d|}\}$表示为较短的离散令牌序列（文档标识符） $z = \{z_1, \ldots\ldots, z_t, \ldots\ldots, z_M\}$，其中每个令牌 z_t $z_t \in$ 1 2 . . . K 是一个 K-路分类变量，其取值为， ， 。

作为原始文档的一种替代序列，分词后的文档标识符 z 应满足以下两个性质：(i) 不同文档具有简短但彼此不同的文档标识符；(ii) 文档标识符应尽可能充分地捕捉其关联文档的语义 [42]。由于 z 是固定长度的序列，且通常短于原始文档 d，因此可以简化模型的训练和推理，并提高效率。

如上所述，本文采用一种分词模型Q： $d \to z$ 将 d映射为文档标识符 z。关于 Q 的更多细节见第3.1节。

生成即检索。将每篇文档分词为文档标识符z后，生成式检索模型 P: $q \to z$ 通过自回归地生成查询 q 对应的文档标识符 z，学习检索相关文档:

$$z = \prod_{t=1}^{M} P(z_t \mid z_{<t}, q), \tag{1}$$

其中， $z_{<t}$ 表示 z 截至时间步 t的前缀。模型采用受约束解码技术，以确保生成的文档标识符 z 存在于语料库 D [5]中。具体而言，基于 D 中有效的文档标识符构建前缀树，并在解码过程中将无效文档标识符的生成概率截断为 0.0。模型使用束搜索检索多篇文档。

3 GENRET

传统上，文档分词通过固定的预处理步骤完成，例如使用文档标题，或使用从 BERT [5, 42] 获得的层次聚类结果。然而，人们发现，这类临时采用的文档分词方法往往无法捕捉文档的完整语义。

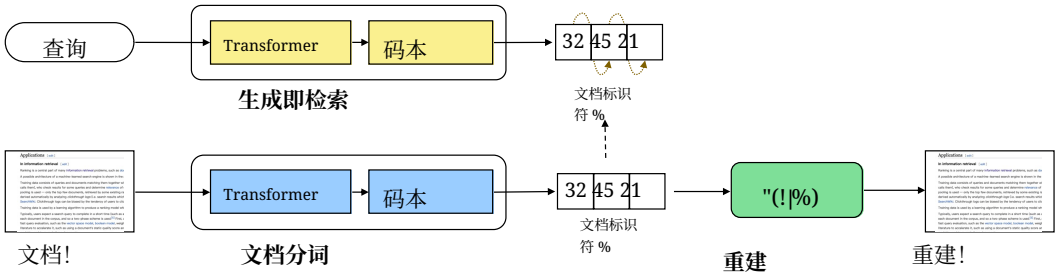


图 2：所提出方法的概述。该方法利用文档分词模型将给定文档转换为一系列离散词元，称为文档标识符。通过这一分词过程，重建模型能够重建原始文档。随后，采用自回归生成模型，通过生成相应的文档标识符来检索文档。

例如，网页标题往往不存在，或与网页内容的相关性较低；而基于聚类的文档标识符则会在离散空间中任意定义文档。

本文提出了GenRet，这是一种基于离散自编码的新型分词学习方法，能够以完全端到端的方式学习语义文档标识符。图2 概述了所提出的方法。所提出的GenRet 由三个主要部分组成：(i) 基于序列到序列的检索模型（$P(z \mid q)$）、(ii) 文档分词模型（$Q(z \mid d)$）以及 (iii) 重建模型（$R(d \mid z)$）。文档分词模型将文档 d 分词为唯一的离散变量 z，而检索模型经过训练后，可针对给定查询 z 生成潜在变量 q。此外，重建模型用于从 z 重新生成原始文档，以确保 z 尽可能充分地捕捉原始文档的语义。

我们将在第3.1节详细介绍文档分词和文档检索模型的架构，在第3.2节介绍所设计的重建模型，并在第3.3节介绍模型优化方法。

3.1 文 档分词与检索 模型

由于文档分词和生成式检索的目标都是将输入文本映射为离散文档 ID，因此我们为文档分词和生成式检索模型采用共享的 T5 Transformer 架构。具体来说，给定输入文本 d[2]，基于 T5 的分词模型会对 d 和文档标识符 $z_{<t}$ 的前缀进行编码，并在时间步 t连续生成 d 的潜在表示 $\mathbf{d}_t$：

$$\mathbf{d}_t = \text{Decoder}(\text{Encoder}(d), z_{<t}) \in \mathbb{R}^D, \tag{2}$$

其中，D 表示模型的隐藏层维度，Encoder(d)表示 Encoder 的输出。

然后，分词模型根据$\mathbf{d}_t$为每个文档生成一个标记。在每个时间步 t，我们定义一个名为 码本 $\mathbf{E}_t \in \mathbb{R}^{K \times D}$ 的外部嵌入矩阵，其中 K 是离散潜在空间的大小。其中有 K 个嵌入向量$\mathbf{e}_{t,j} \in \mathbb{R}^D, j \in [K]$，每个向量 $\mathbf{e}_{t,j}$ 都可以视为一个分段的质心。

基于码本 $\mathbf{E}_t$，通过使用码本进行点积查找，计算时间步 t 处的离散潜变量 z_t

[2]此处我们用文档 d 表示其指称，并指出当输入为 q 时，计算过程相同。

$\mathbf{E}_t$：

$$Q(z_t = j \mid z_{<t}, d) = \text{Softmax}_j(\mathbf{d}_t \cdot \mathbf{E}_t^\top), \tag{3}$$

其中，$Q(z_t = j \mid z_{<t}, d)$ 表示在时间步 t将 d分词为特定值 $j \in [K]$ 的概率，Softmax$_j$ 是用于输出轴 j概率的 softmax 函数。

最后，分词模型会选出概率最大的文档标识符，以此定义文档标识符 z_t：

$$z_t = \arg\max_j Q(z_t = j \mid z_{<t}, d). \tag{4}$$

其中，模型选择 ID j，该 ID 对应的嵌入向量 $\mathbf{e}_{t,j}$ 与 $\mathbf{d}_t$ 的内积最大，作为时间步 t的文档标识符。

生成式检索模型 $P(z \mid q)$ 与 $Q(z \mid d)$具有相同的架构，同时使用输入查询 z生成 q，如公式 1所示。

3.2 文档重建模型

分词模型 Q 生成的文档标识符需要捕获文档的语义信息。为此，我们提出了一种自动编码训练方案，其中设计了一个重建模型 R：$z \to d$，它使用 z 预测 d，以此迫使分词模型 Q：$d \to z$ 生成一个文档标识符 z，使其能够重建回原始文档。

重建模型的输入是文档标识符 z，输出是其关联的文档 d。我们首先使用分词模型的码本，将 z嵌入表示矩阵 $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_M\} \in \mathbb{R}^{M \times D}$：

$$\mathbf{z} = \{\mathbf{e}_{1,z_1}, \mathbf{e}_{2,z_2}, \ldots, \mathbf{e}_{M,z_M}\} \in \mathbb{R}^{M \times D}, \tag{5}$$

其中每个 $t \in [M]$, $\mathbf{z}_t = \mathbf{e}_{t,z_t} \in \mathbb{R}^D$ 是 z_t 在 t步码本$\mathbf{E}_t$中的嵌入向量。

随后，我们设计了一个基于检索的重建模型，根据输入 $\mathbf{z}$，通过从文档集合D 中检索目标文档 d 来预测该文档。输入文档标识符 z与目标文档 d之间的相关性得分定义如下：

$$R(d \mid \mathbf{z}) = \prod_{t=1}^{M} \frac{\exp(\mathbf{z}_t \cdot \text{sg}(\mathbf{d}_t^\top))}{\sum_{d^* \in S(z_{<t})} \exp(\mathbf{z}_t \cdot \text{sg}(\mathbf{d}^{*\top}_t))}, \tag{6}$$

其中，$S(z_{<t})$ 是D的一个子集合，由文档标识符前缀与 $z_{<t}$相同的文档组成。$d^* \in S(z_{<t})$ 表示子集合 $S(z_{<t})$中的一个文档。$\mathbf{d}_t$ 和 $\mathbf{d}^*_t$ 是连续的

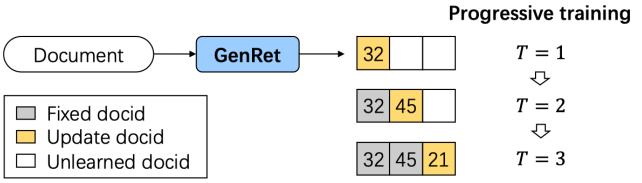


图 3：渐进式训练方案。在第 t步训练中优化 z_t （时间步t处的文档标识符），而 $z_{<t}$ （时间步 t之前的文档标识符）保持固定。

文档的表示 d和 d^*，分别如公式 2所定义。算子 sg(·) 是停止梯度算子，其定义如下：

$$\mathrm{sg}(x) = \begin{cases} x, & \text{forward pass} \\ 0, & \text{backward pass.} \end{cases} \tag{7}$$

直观而言， R（$d \mid \mathbf{z}$）旨在于每个时间步 t从文档集合 S（$z_{<t}$）中检索特定文档 d。集合 S（$z_{<t}$）仅包含与目标文档 d具有相同文档标识符前缀 $z_{<t}$的文档。通过使用这一损失函数，模型在每一步 t都能学习到先前文档标识符 $z_{<t}$未能捕捉的文档残余语义。

3.3 模型优化

对于文档分词模型 Q（$z \mid d$）、生成式检索模型 P（$z \mid q$）和重建模型 R（$d \mid z$），使用自编码联合优化这三个模型颇具挑战，原因有以下两点：

• **以自回归方式学习文档标识符**。具体来说：(i) 在时间步 z_t 的预测 t 依赖于先前预测的文档标识符 $z_{<t}$，而这些标识符在训练初期往往优化不足且变化迅速，因此难以收敛。(ii) 同时优化 z 使得难以保证文档标识符分配的唯一性。为稳定GenRet的训练，我们设计了一种渐进式训练方案（见第3.3.1节）。

• **生成具有多样性的文档标识符**。使用自动编码进行模型优化通常会导致文档标识符分配不均：少数主要文档标识符会被分配给大量文档，而其他大多数文档标识符很少被分配。这种次优的文档标识符分布会影响模型的可区分性，进而触发文档标识符长度增加，以区分相互冲突的文档。我们引入了两种多样化聚类技术，以确保文档标识符的多样性（见第3.3.2节）。

3.3.1 渐进式训练方案。为了以自回归方式优化上述三个模型中的每一个，我们提出了一种渐进式自编码学习方案，如图 3 所示。整个学习方案包含 M 个学习步骤，这些步骤对应于 M 个分词中的最终文档标识符。步骤 z_T 处的文档标识符 $T \in [M]$ 在相应的学习步骤中进行学习和优化。此外，在每一步 $T \in [M]$，都会更新文档标识符 z_T 以及与 z_T生成相关的模型参数，而先前生成的文档标识符 $z_{<T}$和其他参数则保持固定。通过逐步执行上述过程，我们最终可以优化并学习模型。

在每个优化步骤中，例如第 T 步，我们为文档分词设计学习目标，该目标由以下详述的三个损失函数组成。

重构损失。我们利用重构模型 $R(d \mid z)$作为辅助模型来学习优化文档标识符生成，其主要目标是尽可能捕获文档标识符中的语义。因此，我们将步骤T 的重构损失函数定义如下：

$$\begin{aligned} \mathcal{L}_{\text{Rec}} &= -\log R(d \mid \hat{\mathbf{z}}_{\le T}) \\ \hat{\mathbf{z}}_{\le T} &= \{\mathrm{sg}(\mathbf{z}_1), \ldots, \mathrm{sg}(\mathbf{z}_{T-1}), \mathbf{z}_T\} \in \mathbb{R}^{T \times D} \\ \forall t \in [T]: \ \mathbf{z}_t &= \mathbf{e}_{t,j^*} \in \mathbb{R}^D, \ j^* = \arg\max_j Q(z_t = j \mid z_{<t}, d), \end{aligned} \tag{8}$$

其中，$\hat{\mathbf{z}}_{\le T}$是 z的第一个 T表示，并且只有变量$\mathbf{z}_T$在步骤 T中进行优化。 $Q(z_t = j \mid z_{<t}, \ d)$定义见公式3。因此，最小化 LRec 时可以优化文档分词模型 Q。

值得注意的是，由于计算涉及不可微运算——argmax(·)，我们采用直通梯度估计，从重构损失 [44, 49] 反向传播梯度。具体而言，文档表示 $\mathbf{d}_T$ 的梯度定义为 $\frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{d}_T} := \frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{z}_T}$。码本 *embed- ding* $\mathbf{e}_{T,j}$ 的梯度定义为 $\frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{e}_{T,j}} := 1_{z_T = j} \frac{\partial \mathcal{L}_{\text{Rec}}}{\partial \mathbf{z}_T}$。

承诺损失。此外，为确保预测的文档标识符能够与某个嵌入建立承诺，并避免模型遗忘先前的文档标识符 $z_{<t}$，我们加入如下承诺损失：

$$\mathcal{L}_{\text{Com}} = -\sum_{t=1}^{T} \log Q(z_t \mid z_{<t}, d). \tag{9}$$

检索损失。对于生成式检索模型 P，我们将其与文档分词模型 Q联合训练，其中 P学习在给定查询 q的情况下生成相关文档 d的文档标识符。具体而言，假设（q, d）为查询与相关文档对；我们将检索模型 P 的学习目标定义为：

$$\mathcal{L}_{\text{Ret}} = -\log \frac{\exp(\mathbf{q}_T \cdot \mathbf{d}_T)}{\sum_{d^- \sim B} \exp(\mathbf{q}_T \cdot \mathbf{d}^-_T)} - \sum_{t=1}^{T} \log P(z_t \mid z_{<t}, q), \tag{10}$$

其中，第一项是面向排序的损失，通过使用（q， d）对来增强模型； d^- 是来自同一训练小批次 B的批内负样本文档； $\mathbf{q}_T$ 和$\mathbf{d}_T$分别表示 q和 d在时间步 T的表征。第二项是根据查询 q 生成文档标识符 z的交叉熵损失。

我们在步骤-T 使用的最终损失是重构损失、承诺损失和检索损失之和：

$$\mathcal{L} = \mathcal{L}_{\text{Rec}} + \mathcal{L}_{\text{Com}} + \mathcal{L}_{\text{Ret}}. \tag{11}$$

3.3.2 多样化聚类技术。为确保生成的文档标识符具有多样性，我们采用了两种多样化聚类技术——码本初始化和每个渐进式训练步骤中的文档标识符重新分配。其中，码本初始化主要旨在提高语义空间划分的均衡性，而文档标识符重新分配主要旨在提高文档标识符分配的均衡性。

码本初始化。 为了初始化我们的模型，我们首先通过将连续表示$\mathbf{d}_T$传递给重建模型来对模型进行预热，而不是

文档标识符表示 $\mathbf{z}_T$，定义见公式 5。在此预热阶段，我们使用重构损失 LRec 和承诺损失 LCom 来优化模型。接下来，我们收集 D 中所有文档的连续表示 $\mathbf{d}_T$，并将它们聚类为 K组。然后使用这些簇的质心作为初始化码本 $\mathbf{E}_T$。为平衡初始化后的文档标识符分布，我们采用多样化约束聚类算法约束 K 均值，该算法通过将簇分配步骤（期望最大化中的 E）表述为最小费用流（MCF）线性网络优化问题来对其进行修改 [2]。

文档标识符重新分配。为了为一批文档分配文档标识符，我们修改公式 3 中的点积查找结果，确保批次中不同文档的文档标识符各不相同 [6, 49]。具体来说，令 $\mathbf{D}_t = \{\mathbf{d}(1)_t, \ldots, \mathbf{d}(B)_t\} \in \mathbb{R}^{B\times D}$ 表示批量大小为 B 的一批文档的连续表示。点积结果表示为 $\mathbf{H} = \mathbf{D}_t \cdot \mathbf{E}_t^\top \in \mathbb{R}^{B\times K}$。为了获得不同的文档标识符，我们计算替代结果 $\mathbf{H}^* = \mathrm{Diag}(\mathbf{u}) \exp(\mathbf{H}_\epsilon)\, \mathrm{Diag}(\mathbf{v})$，其中 u 和 v 分别是 $\mathbb{R}^K$ 和 $\mathbb{R}^B$ 中的重新归一化向量。重新归一化向量通过迭代 Sinkhorn-Knopp 算法 [9] 计算得出。最后，使用 $\mathbf{H}^*$ 代替 H 执行 Softmax（公式 3）和 arg max（公式 4）操作，以获得文档标识符 z_t。

4 实验设置

4.1 数据集

我们在三个知名的文档检索数据集上进行实验：NQ [21], MS MARCO [4],和 BEIR [43]。

NQ320K。 NQ320K 是用于评估生成式检索模型的热门数据集 [42, 46]。它基于 Google 提出的自然问题（NQ）数据集 [21]。NQ320K 包含 32 万个查询-文档对，其中文档取自维基百科页面，查询则是自然语言问题。我们遵循 NCI [46] 中的评估设置，并进一步将测试集拆分为两个子集：已见测试集，其中查询的标注目标文档包含在训练集中；以及未见测试，其中训练集中不包含任何带标签的文档。

MSMARCO。 MS MARCO 是来自 Bing 搜索的查询和网页集合。与 NQ320K 类似，并遵循 [52], ，我们从带标签的文档中抽取一个子集，并使用其对应的查询进行训练。我们使用 MS MARCO 开发集中的查询评估模型，并在抽取的文档子集上评估检索效果。

BEIR。 BEIR 是一个用于异构检索任务的数据集集合。在本文中，我们在 6 个 BEIR 数据集上评估模型，这些数据集涵盖了不同的检索任务和文档集合，数据来源包括 NQ 和 MS MARCO：(i) BEIR-Arg 检索论点的反驳意见；(ii) BEIR-Covid 检索有关 COVID-19 疫情的科学文章；(iii) BEIR-NFC 从 PubMed 检索医学文档；(iv) BEIR-SciFact 检索用于事实核查的科学论文；(v) BEIR-SciDocs 检索科学论文的引用文献；(vi) BEIR-FiQA 检索金融文档。

我们在表1中总结了上述数据集的统计信息。

表 1：实验中使用的数据集统计信息。“# 测试查询”一栏中由 / 分隔的三个数值分别表示完整子集、已见子集和未见子集中的查询数量。在 BEIR 中，测试集中的所有查询均为未见查询。

数据集	# 文档	# 测试查询	# 训练对
NQ320K	109,739	7,830 / 6,075 / 1,755	307,373
MS MARCO	323,569	5,187 / 807 / 4,380	366,235
BEIR-Arg	8,674	1,406	-
BEIR-Covid	171,332	50	-
BEIR-NFC	3,633	323	-
BEIR-SciFact	5,183	300	-
BEIR-SciDocs	25,657	1,000	-
BEIR-FiQA	57,638	648	-

4.2 评估指标

在 NQ320K 上，我们遵循 [46]，使用 Recall@{1,10,100} 和 Mean Reciprocal Rank (MRR)@100 作为评估指标。在 MS MARCO 上，我们遵循 [52]，使用 Recall@{1, 10, 100} 和 MRR@10 作为评估指标。在 BEIR 上，我们使用 nDCG@10 作为主要指标，并计算多个下游子数据集的 nDCG@10 平均值作为总体指标。

4.3 基线

我们考虑三类基线方法：稀疏检索方法、稠密检索方法和生成式检索方法。

稀疏检索基线如下： • **BM25**，使用词频-逆文档频率特征衡量词项权重；我们采用 http://pyserini.io/ 提供的实现。 • **DocT5Query**，利用以文档为输入、经微调的 T5 模型预测可能的查询，并用这些查询扩展文档。

稠密检索基线如下： • **DPR** [19],一种双编码器模型，使用 [CLS] 标记在 BERT 中的表示。 • **ANCE** [47], 采用异步更新的近似最近邻索引器，为基于 RoBERTa 的双编码器模型挖掘训练所需的困难负例。 • **Sentence-T5** [30], 一种使用 T5 生成连续句子嵌入的双编码器模型。我们在自己的数据集上复现了 Sentence-T5（简称 ST5）；该模型基于 T5-Base 编码器-解码器模型，并使用批内负例进行训练。

• **GTR** [31],一种先进的稠密检索模型，通过对比学习在数十亿组配对数据上预训练 Sentence-T5。
• **Contriever** [16], 一种双编码器模型，通过独立裁剪和逆向完形填空任务进行无监督对比学习预训练。

生成式检索基线如下： • **GENRE** [5],一种生成文档标题的自回归检索模型。原始 GENRE 使用 BART在 KILT 数据集上训练，我们使用 T5 在我们的数据集上复现GENRE，以进行公平比较。对于没有标题的数据集，我们使用文档的前 32 个词元作为伪标题。 • **DSI** [42], 使用层次K 均值聚类结果表示文档，并使用前 32 个词元作为伪查询对文档进行索引。由于原始代码未开源，我们使用T5-Base 和 NCI 的文档标识符复现 DSI。[46] • **SEAL** [1] 使用文档中的任意 n 元语法作为文档标识符，并在约束条件下检索文档

使用预构建的FM索引器。我们引用 Wang 等人报告的结果 [46]。 • **CGR-Contra** [23], 是一种标题生成模型，采用上下文化词汇嵌入和对比学习损失。 • **DSI-QG** [53], 使用查询生成模型来扩充文档集合。我们使用 T5 和自己的数据集复现了 DSI-QG 的结果。 • **NCI** [46], 使用前缀感知的权重自适应解码器和多种查询生成策略，包括 DocAs- Query 和 DocT5Query。具体而言，NCI 通过为每篇文档生成 15 个查询来扩充训练数据。 • **Ultron** [52], 采用三阶段训练流程，并使用三种标识符表示文档，包括 URL、PQ 和 Atomic。

我们重点介绍了复现的三个基线方法，它们构成了与所提出方法的公平比较。这些方法均使用 T5 模型和实验设置，但模型输出各不相同：(i) Sentence-T5 输出连续向量，(ii) GENRE 输出文档标题，(iii) DSI-QG 输出聚类 ID，而 GenRet 输出使用所提出的分词方法学习得到的文档标识符。

4.4 实现细节

超参数。 在实验中，我们使用 T5-Base 模型 [35] 作为基础 Transformer，并为每个时间步初始化一个新的码本嵌入 $\mathbf{E}_t$。对于所有数据集，我们将聚类数量设为 $K = 512$，文档标识符的长度 M 则取决于文档总数。对于包含更多候选文档的数据集，我们会设置更大的M，以确保为所有文档分配唯一的文档标识符。在文档标识符重新分配过程中，超参数 ϵ 设为 1。0，并执行 100 次 Sinkhorn-Knopp 算法迭代。

使用查询生成进行索引。 沿用先前的工作 [45,46, 53], ，我们使用查询生成模型生成合成的（查询、文档）对，以扩充数据。具体而言，我们使用 DocT5Query [8] 预训练的查询生成模型来扩充 NQ 和 MS MARCO 数据集。在查询生成过程中，我们使用参数为 $p = 0.8$, $t = 0.8$ 的核采样，并为集合中的每篇文档生成五个查询。对于 BEIR 数据集，我们使用 GPL [45], 生成的查询，这些查询可从其网站下载。[3] GPL 使用在 MS MARCO 上训练的 DocT5Query [8]生成器，为每个 BEIR 数据集生成约 25 万个查询。

训练与推理。所提出的模型和复现的基线模型均使用 PyTorch 1.7.1 和 Hugging Face transformers 4.22.2 实现。我们使用 AdamW 优化模型，并将学习率设为 $5e-4$。批次大小为256，每个时间步最多优化 500k 步。在渐进式训练中，我们先对模型进行 5K 步预热，然后按照第3.3.1节所述，使用聚类质心初始化码本。我们使用约束聚类[4] 来获得多样化聚类结果。推理时，我们使用带约束解码的束搜索 [5] ，束宽为100。

[3]https://public.ukp.informatik.tu-darmstadt.de/kwang/gpl/generated-data/beir/ [4] https://github.com/joshlk/k-means-constrained

5 实验结果

5.1 主要结果

NQ320K 上的结果。在表2中，我们列出了 NQ320K 上的结果。GenRet 优于强大的预训练稠密检索模型 GTR，以及此前表现最佳的生成式检索方法 NCI，从而在 NQ320K 数据集上创下新的最先进水平。此外，我们的结果表明，现有生成式检索方法在已见测试集上表现出色，但在未见测试上不及稠密检索方法。例如，NCI 在已见测试集上的 MRR@100 为76。8，高于 GTR-Base 的 MRR@10065。3。然而，在未见测试数据上，NCI 的表现不如 GTR-Base。相比之下，GenRet 在已见和未见测试数据上均表现出色。这一结果凸显了GenRet 通过端到端优化学习具有语义的离散文档标识符，从而兼具稠密检索和生成式检索优势的能力。

MS MARCO 上的结果。 表 3 展示了在 MS MARCO 数据集上的结果。 GenRet 显著优于以往的生成式检索方法，并取得了与最先进的稠密检索方法 GTR 相当的结果。此外，以往利用标题和 URL 等元数据的生成式检索方法（如 GENRE、Ultron）虽然在 NQ320K 数据集上表现不俗，但在 MS MARCO 数据集上的表现不如此前最佳的稠密检索方法（GTR）和稀疏检索方法（DocT5Query）。这可能是因为 NQ320K 数据集检索的是维基百科文档，其中标题等元数据能够有效捕捉文档的语义。而 MS MARCO 数据集是一个网页搜索数据集，元数据往往无法充分表征文档，导致生成式检索模型的性能下降。相比之下，GenRet 学会生成语义文档标识符，从而有效提升生成式检索模型的性能。

BEIR 上的结果。表4列出了基线模型和GenRet 在 BEIR 的六个数据集上的结果。这些数据集涵盖了多种多样的信息检索场景。平均而言，GenRet优于 BM25 和 GTR-Base 等强基线模型，且与最先进的稀疏和稠密检索方法相比也具有竞争力。与使用相同训练数据和 T5 主干模型的 ST5 GPL 方法相比，Gen- Ret 取得了更好的结果。此外，GenRet 相比此前使用标题作为文档标识符的生成式检索模型 GENRE，取得了显著提升。此外，GENRE 在某些数据集上的表现较差，例如 BEIR-Covid 和 BEIR-SciDocs。这可能是因为这些数据集中各文档的标题无法充分体现其语义内容。

5.2 检索新文档的性能

在本实验中，我们研究了不同文档分词技术对生成式检索模型检索新文档能力的影响。采用不同分词方法的生成式模型在排除未见文档的 NQ320K 数据上进行训练，并在 NQ320K 未见测试集和 BEIR-{Arg, NFC, SciDocs} 数据集上进行评估。对于基线方法而言，它们使用基于规则的文档分词方法，并利用各自的分词技术为目标文档集合生成文档标识符。相比之下，我们提出的方法

表 2：自然问题（NQ320K）上的结果。标有 † 的方法结果来自我们自行的重新实现，其余结果来自各自的官方实现。带有♠的方法在训练期间使用了额外的带注释文档检索数据。*和 **分别表示相较于此前最佳生成式检索基线取得了显著提升，p值分别为 < 0.05 和 < 0.01。♮和 ♯ 分别表示相较于此前最佳稠密检索基线取得了显著提升，p 值分别为 < 0.05 和 < 0.01。每项指标的最佳结果以粗体标示。

	完整测试				已见测试集				未见测试			
方法	R@1	R@10	R@100	MRR@100	R@1	R@10	R@100	MRR@100	R@1	R@10	R@100	MRR@100
稀疏检索												
BM25 [39]	29.7	60.3	82.1	40.2	29.1	59.8	82.4	39.5	32.3	61.9	81.2	42.7
DocT5Query [8]	38.0	69.3	86.1	48.9	35.1	68.3	86.4	46.7	48.5	72.9	85.0	57.0
稠密检索												
DPR [19]	50.2	77.7	90.9	59.9	50.2	78.7	91.6	60.2	50.0	74.2	88.7	58.8
ANCE [47]	50.2	78.5	91.4	60.2	49.7	79.2	92.3	60.1	52.0	75.9	88.0	60.5
Sentence-T5†[30]	53.6	83.0	93.8	64.1	53.4	83.9	94.7	63.8	56.5	79.5	90.7	64.9
GTR-Base♠ [31]	56.0	84.4	93.7	66.2	54.4	84.7	94.2	65.3	61.9	83.2	92.1	69.6
生成式检索												
GENRE† [5]	55.2	67.3	75.4	59.9	69.5	83.7	90.4	75.0	6.0	10.4	23.4	7.8
DSI† [42]	55.2	67.4	78.0	59.6	69.7	83.6	90.5	74.7	1.3	7.2	31.5	3.5
SEAL [1]	59.9	81.2	90.9	67.7	-	-	-	-	-	-	-	-
CGR-Contra [23]	63.4	81.1	-	-	-	-	-	-	-	-	-	-
DSI-QG† [53]	63.1	80.7	88.0	69.5	68.0	85.0	91.4	74.3	45.9	65.8	76.3	52.8
NCI [46]	66.4	85.7	92.4	73.6	69.8	88.5	94.6	76.8	54.5	75.9	84.8	62.4
Ours	**68.1**$^{*\sharp}$	**88.8**$^{*\natural}$	**95.2***	**75.9**$^{*\natural}$	**70.2**$^{\sharp}$	**90.3**$^{\sharp}$	**96.0**$^{\natural}$	**77.7**$^{\sharp}$	**62.5****	**83.6****	**92.5****	**70.4****

表 3：MS MARCO 上的结果。标记为 † 的方法结果来自我们自行重新实现的版本。标记为 ♠ 的方法使用额外的人工标注检索数据进行训练。*/ 表示相较于先前的生成式检索基线有显著提升，p 值为 $< 0.05/0.01$。每项指标的最佳结果以粗体标示。**

方法	R@1	R@10	R@100	MRR@10
稀疏检索				
BM25 [39]	39.1	69.1	86.2	48.6
DocT5Query [8]	46.7	76.5	90.4	56.2
稠密检索				
ANCE [47]	45.6	75.7	89.6	55.6
Sentence-T5†[30]	41.8	75.4	91.2	52.8
GTR-Base♠ [31]	46.2	79.3	**93.8**	57.6
生成式检索				
GENRE† [5]	35.6	57.6	79.1	42.3
Ultron-URL [52]	29.6	67.8	-	40.0
Ultron-PQ [52]	31.6	73.1	-	45.4
Ultron-Atomic [52]	32.8	74.1	-	46.9
Ours	**47.9****	**79.8****	91.6**	**58.1****

使用分词模型对目标集合中的文档进行分词，从而生成文档标识符。然而，我们的方法可能会生成重复的文档标识符。在这种情况下，所有对应的文档都会被检索出来，并以任意顺序打乱排列。该评估结果总结于表5。

不考虑文档语义信息的文档分词方法，例如 Naive String 和 Atomic，在不借助模型的情况下检索新文档时效果不佳。

表 4：BEIR上的结果。指标为 nDCG@10。标有 † 的方法结果来自我们自行重新实现的版本。ST5 GPL 表示在 GPL 数据集 [45] 上训练的 Sentence-T5。

方法	Arg	Covid	NFC	SciFact	SciDocs	FiQA	**平均**
稀疏检索							
BM25 [39]	29.1	58.9	33.5	67.4	14.8	23.6	37.8
DocT5Query [8]	34.9	71.3	32.8	67.5	16.2	29.1	41.9
稠密检索							
ANCE [47]	31.4	73.3	23.1	50.8	12.2	29.5	36.7
ST5GPL† [30]	32.1	74.4	30.1	58.6	12.7	26.0	39.0
GTR-Base [31]	37.3	61.2	30.0	58.4	14.0	35.1	39.3
Contriever [16]	40.0	68.8	33.5	61.4	16.3	30.7	41.8
生成式检索							
GENRE† [53]	42.5	14.7	20.0	42.3	6.8	11.6	30.0
Ours	34.3	71.8	31.6	63.9	14.9	30.2	41.1

更新。考虑文档语义信息的方法，例如基于标题或 BERT 聚类的方法，取得了一些改进。我们提出的文档分词方法显著优于现有的基于规则的文档分词方法。例如，将在 NQ（基于维基百科文档的事实型问答数据集）上训练的模型应用于不同文档集合 BEIR-SciDocs 上的一项独立检索任务——在科学文章集合上进行的引文检索任务——时，我们提出的文档分词模型仍展现出良好的效果，nDCG@10 达到 12.3，与在目标文档集合上训练的模型相当。这表明

表 5：采用不同文档分词方法检索新文档的零样本评估。第二列表示文档标识符类型，其中 BERT-HC 表示 BERT-层次聚类 [42], Prefix-HC 表示前缀感知的 BERT-层次聚类 [46], , dAE 表示离散自编码。

方法	文档标识符	NQ (R@1) 未见过的	BEIR(nDCG@10) Arg	NFC	Sci文档
DSI-朴素† [42]	朴素字符串	0.0	0.1	1.0	0.1
DSI-Atomic† [42]	原子	0.0	0.2	0.8	0.1
GENRE† [5]	标题	6.0	0.0	2.4	0.6
DSI† [42]	BERT-HC	1.3	1.8	11.1	5.9
NCI [46]	Prefix-HC	15.5	0.9	4.3	1.2
Ours	dAE	**34.2**	**12.1**	**12.1**	**12.3**

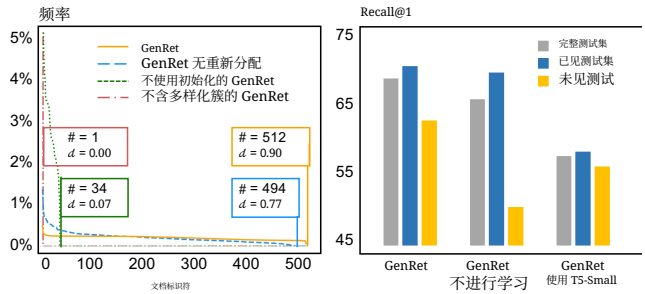


图4：左：NQ320K 上的文档标识符分布。文档标识符按指定频率排序。右：NQ320K 上的消融研究。

我们提出的方法能够有效地将文档的语义信息编码到文档标识符中，从而使文档标识符与生成式检索模型更加匹配。

5.3 分析实验

我们进一步开展分析实验，以研究所提出方法的有效性。

在图4（左）中，我们绘制了不同学习方法在第一个时间步的文档标识符频率。我们用一个方框标注每种方法，其中包含一个文档标识符和一个多样性指标d，其计算方式为：$d = 1 - \frac{1}{2n}\sum_{j=1}^{K}|n_j - n_u|$，其中 $|\cdot|$ 表示绝对值，n 表示文档总数，n_j 表示具有文档标识符 $= j$的文档数，而 $n_u = \frac{n}{K}$表示均匀分布下每个文档标识符对应的预期文档数。

结果证明，GenRet（以黄色线条表示）在分布均匀性方面表现更优。它利用了所有潜在的文档标识符 $k = 512$，并取得了最高的多样性指标，数值为 $d = 0.90$。未采用文档标识符重新分配的方法也实现了相对均衡的分布，其多样性指标为 $d = 0.77$。然而，未采用多样化码本初始化的方法，其分布极不均匀。这可能是因为大多数随机初始化的码本嵌入在初始训练阶段未被模型选中，因而没有得到更新，也就无法在后续训练中被选中。此外，未采用多样化聚类的模型往往以.....

表6：效率分析。

方法	内存	时间（离线）	前-K	时间（在线）
ANCE	1160MB	145分钟	100	0.69秒
GTR-Base	1430MB	140分钟	100	1.97s
GENRE	**851MB**	**0min**	100	1.41 秒
			10	0.69s
DSI	851MB	310分钟	100	0.32秒
			10	0.21秒
Ours	860MB	220分钟	100	**0.16秒**
			10	**0.10s**

收敛到一个平凡解，即所有文档都被分配相同的文档标识符。

在图 4（右）中，展示了两个消融变体的结果。首先，GenRet w/o learning 是一个直接使用 GenRet 的最终输出文档标识符进行训练的生成式模型，未采用所提出的学习方案。其检索性能在已见测试数据上与 GenRet 相当；然而，在未见测试数据上显著较低。所提出的渐进式自动编码方案对于模型捕获文档的语义信息至关重要，而不仅仅是定义良好的离散文档 ID。其次，GenRet w/ T5-Small 使用较小的模型，其性能不如使用 T5-Base 的 GenRet 。然而，已见测试数据与未见测试数据上的性能差距更小，这可能归因于小型模型有限的拟合能力。

5.4 效率分析

在表6中，我们比较了 GenRet 与 MS MARCO（323,569 篇文档）上的基线模型在内存占用、离线索引时间（不包括神经网络训练时间），以及不同 Top-K 值下的在线检索延迟方面的表现。我们有以下四点观察：（i）生成式检索模型（GENRE、DSI-QG 和所提出的模型）的内存占用小于稠密和稀疏检索方法。生成式检索模型的内存占用仅取决于模型参数，而稠密和稀疏检索方法需要额外的存储空间来保存文档嵌入，这会随文档集合的规模线性增长。（ii）DSI 和 GenRet 的离线索引耗时较长，因为 DSI 涉及使用 BERT 对文档进行编码和聚类，而GenRet 需要使用分词模型对文档进行分词。稠密检索的离线耗时来自文档编码；GENRE 使用标题，因此无需离线计算。（iii）生成式检索模型的在线检索延迟与束宽（即 Top-K ）以及文档标识符的长度有关。GenRet 采用多样化聚类生成更短的文档标识符，因此与 DSI 和 GENRE 相比，在线检索速度更快。

5.5 案例研究

表 7 展示了GENRE、NCI 和 GenRet针对查询“州法院可以下令重新审判什么案件”及其在 NQ320K 中对应文档的输出示例。结果显示，GenRet与基线方法不同，成功返回了目标文档标识符

表7：NQ320K 上模型的输出。 黄色和灰色背景表示 Ret 的第 t=1 或 t=2 步中注意力较高的词语。 文档***D***的文档标识符**表示文档 D 的分词后文档标识符；**查询***Q***生成的文档标识符**表示为查询 Q 生成的文档标识符。**

测试查询（Q）： 州法院可以下令重新审判什么	
目标文档（D）： 美国	
以及提起上诉的规	法院 联邦法院。上
上诉的性质可能因案件类型而大不相同 [...]	
– GENRE:	**文档D的文档标识符：** 美国上诉程序； **查询Q生成的文档标识符：** 加入联邦（）
– NCI:	**文档D的文档标识符：** 22-18-10-1; **查询Q生成的文档标识符：** 14-10-0-4（）
– GenRet:	**文档D的文档标识符：** 95-375-59; **查询Q生成的文档标识符：** 95-375-59 (√)

文档。我们根据 GenRet在不同时间步 t的注意力激活情况，突出显示目标文档中的词语。黄色表示在 t = 1获得较高注意力的词语，而灰色表示在 t = 2获得较高注意力的词语。该示例表明，模型在不同时间步关注不同的词语。GenRet会更多关注与主题相关的词语，例如 上诉，在 t = 1，以及更多关注与国家相关的词语，例如美国，在 t = 2。

6 相关工作

稀疏检索。 传统稀疏检索使用词项匹配指标（如词频-逆文档频率）计算文档得分 [38], 查询似然 [22] 或 BM25 [39]。由于在准确性和效率之间取得了出色的权衡，它在实践中得到了广泛应用。一些方法利用深度神经网络自适应地分配词项重要性 [12, 14, 51]。随着预训练语言模型的最新发展，DeepCT [10] 和 HDCT [11] 利用 BERT 的上下文化文本表示计算词项重要性。Doc2Query [32]和 DocT5Query [8] 则使用 T5 等生成式模型预测相关查询，并在构建 BM25 索引之前扩充文档。稀疏检索通常会受到词汇不匹配问题的影响 [24]。

稠密检索。 稠密检索（DR）将查询和文档表示为稠密向量，并通过内积或余弦相似度对其相似度进行建模 [19]。与稀疏检索相比，稠密检索缓解了词汇不匹配问题。人们提出了多种技术来改进 DR 模型，例如困难负例挖掘 [34, 47], 后期交互 [20, 41], 和知识蒸馏 [15, 26]。近期研究表明，在大规模语料库上使用对比学习预训练 DR 模型是有效的 [16, 31, 37]。尽管取得了成功，DR 方法仍有若干局限 [5, 28]: （i）DR 模型采用索引-检索流水线，并使用固定的搜索流程（MIPS），因此难以对模型进行端到端优化 [42, 46]。（ii）DR 模型的训练依赖对比学习 [19] 来区分正例和负例，这与大型语言模型的训练目标不一致 [3]，且未能充分利用预训练语言模型的能力 [1]。

生成式检索。 生成式检索正受到越来越多的关注。它通过使用 T5 等生成式模型生成文档标识符来检索文档。生成式检索为文档检索任务提供了端到端的解决方案 [28, 42]，并能更好地发挥大型生成式语言模型的能力 [1]。Cao 等人 [5] 首次提出了一种自回归实体检索模型通过生成标题来检索文档。Tay 等人 [42] 提出了可微搜索索引（DSI），并将文档表示为原子ID、朴素字符串或语义字符串。Bevilacqua 等人 [1] 建议使用文档的任意跨度作为文档标识符。此外，近期研究还探索了多阶段预训练 [7, 52], 查询生成 [46, 52, 53],上下文化嵌入 [23], 和持续学习 [27], 。然而，现有的生成式检索模型存在一个局限，即它们依赖固定的文档分词来生成文档标识符，而这通常无法捕捉文档的语义信息 [42]。应如何定义文档标识符仍是一个开放性问题。为了进一步捕捉文档标识符中的文档语义，我们提出了文档标记化学习方法。所提出的离散自编码学习方案能够以端到端的方式自动生成语义文档标识符。

离散表示学习。 使用神经网络学习离散表示是机器学习中的一个重要研究领域。对于图像，Rolfe [40] 提出了离散变分自编码器，而 VQ-VAE [44]通过向量量化学习量化表示。Dall-E [36] 使用自回归模型生成离散图像表示，以实现文本到图像生成。近年来，表示学习在自然语言处理任务中引起了广泛关注，应用于机器翻译 [54], 、对话生成 [50], 和文本分类 [17, 48]等任务。对于文档检索，RepCONC [49] 采用基于约束聚类的离散表示学习方法进行向量压缩。我们提出了一种面向生成式检索的文档标记化学习方法，通过渐进式训练捕捉文档标识符的自回归特性，并利用多样化聚类技术增强文档标识符的多样性。

7 结论

本文提出了一种面向生成式检索的文档标记化学习方法，名为 GenRet。该方法采用离散自编码方法，将文档标记化为简短的离散表示（即文档标识符），从而确保生成的文档标识符具有语义。我们提出了一种渐进式训练方法和两种多样化聚类技术，以增强模型训练。在多个文档检索数据集上的实证结果表明，该方法有效。尤其是，GenRet 在未见文档上表现出色，并且能够很好地泛化到多种检索任务。未来，我们希望将这一方法扩展至大型文档集合。我们还计划探索使用大规模语言模型进行文档标记化的生成式预训练。此外，我们打算研究渐进式训练中文档标识符前缀的动态适配。

参考文献

[1] Michele Bevilacqua、Giuseppe Ottaviano、Patrick Lewis、Wen tau Yih、Sebastian Riedel 和 Fabio Petroni。2022年。自回归搜索引擎：将子字符串生成为文档标识符。发表于 *NeurIPS*。[2] Paul S. Bradley、Kristin P. Bennett 和 Ayhan Demiriz。2000年。受约束的 K-均值聚类。微软研究院（2000年）。[3] Tom B. Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared Kaplan、Pra- fulla Dhariwal、ArvindNeelakantan、PranavShyam、Girish Sastry、Amanda Askell、Sandhini Agarwal、Ariel Herbert-Voss、Gretchen Krueger、T. J. Henighan、Rewon Child、Aditya Ramesh、Daniel M. Ziegler、Jeff Wu、Clemens Winter、Christopher Hesse、Mark Chen、Eric Sigler、Mateusz Litwin、Scott Gray、Benjamin Chess、Jack

Clark、Christopher Berner、Sam McCandlish、Alec Radford、Ilya Sutskever 和 Dario Amodei。2020年。语言模型是少样本学习者。发表于 NeurIPS。[4] Daniel Fernando Campos、Tri Nguyen、Mir Rosenberg、Xia Song、Jianfeng Gao、Saurabh Tiwary、Rangan Majumder、Li Deng 和 Bhaskar Mitra。2016年。MS MARCO：由人工生成的机器阅读理解数据集。arXiv abs/1611.09268（2016年）。[5] Nicola De Cao、Gautier Izacard、Sebastian Riedel 和 Fabio Petroni。2021年。自回归实体检索。发表于 ICLR。[6] Mathilde Caron、Ishan Misra、Julien Mairal、Priya Goyal、Piotr Bojanowski 和 Armand Joulin。2020年。通过对比簇分配进行视觉特征的无监督学习。arXiv abs/2006.09882（2020年）。[7] Jiangui Chen、Ruqing Zhang、Jiafeng Guo、Y. Liu、Yixing Fan 和 Xueqi Cheng。2022年。CorpusBrain：为知识密集型语言任务预训练生成式检索模型。发表于 CIKM。[8] David R. Cheriton。2019年。从 Doc2query 到 DocTTTTTquery。在线预印本。[9] Marco Cuturi。2013年。Sinkhorn 距离：以闪电般的速度计算最优传输。发表于 NIPS。[10] Zhuyun Dai 和 Jamie Callan。2019年。面向第一阶段检索的上下文感知句子/段落术语重要性估计。arXiv abs/1910.10687（2019年）。[11] Zhuyun Dai 和 Jamie Callan。2020年。面向即席搜索的上下文感知文档术语加权。发表于 WWW。[12] Mostafa Dehghani、Hamed Zamani、Aliaksei Severyn、Jaap Kamps 和 W. Bruce Croft。2017年。弱监督下的神经排序模型。发表于 SIGIR。[13] Daniel Gillick、Alessandro Presta 和 Gaurav Singh Tomar。2018年。连续空间中的端到端检索。arXiv abs/1811.08008（2018年）。[14] Jiafeng Guo、Yixing Fan、Qingyao Ai 和 W. Bruce Croft。2016年。一种用于即席检索的深度相关性匹配模型。发表于 CIKM。[15] Sebastian Hofstätter、Sheng-Chieh Lin、Jheng-Hong Yang、Jimmy J. Lin 和 Allan Hanbury。2021年。通过均衡且具备主题感知能力的采样，高效地训练有效的稠密检索器。发表于 SIGIR。[16] Gautier Izacard、Mathilde Caron、Lucas Hosseini、Sebastian Riedel、Piotr Bo- janowski、Armand Joulin 和 Edouard Grave。2022年。通过对比学习进行无监督稠密信息检索。发表于 TMLR。[17] Shuning Jin、Sam Wiseman、Karl Stratos 和 Karen Livescu。2020年。用于低资源文本分类的离散潜变量表示。发表于 ACL。[18] Jeff Johnson、Matthijs Douze 和 Herve Jegou。2017年。利用 GPU 进行十亿规模的相似性搜索。《IEEE 大数据汇刊》7（2017年），535–547。[19] Vladimir Karpukhin、Barlas Oğuz、Sewon Min、Patrick Lewis、Ledell Yu Wu、Sergey Edunov、Danqi Chen 和 Wen tau Yih。2020年。面向开放域问答的稠密段落检索。发表于 EMNLP。[20] Omar Khattab 和 Matei Zaharia。2020年。ColBERT：通过 BERT 上的上下文后期交互，实现高效且有效的段落搜索。发表于 SIGIR。[21] Tom Kwiatkowski、Jennimaria Palomaki、Olivia Redfield、Michael Collins、Ankur P. Parikh、Chris Alberti、Danielle Epstein、Illia Polosukhin、Jacob De- vlin、Kenton Lee、Kristina Toutanova、Llion Jones、Matthew Kelcey、Ming-Wei Chang、Andrew M. Dai、Jakob Uszkoreit、Quoc V. Le 和 Slav Petrov。2019年。自然问题：问答研究的一个基准。《TACL》7（2019年），453–466。[22] John Lafferty 和 ChengXiang Zhai。2001年。用于信息检索的文档语言模型、查询模型与风险最小化。发表于 SIGIR。[23] Hyunji Lee、Jaeyoung Kim、Hoyeon Chang、Hanseok Oh、Sohee Yang、Vladimir Karpukhin、Yi Lu 和 Minjoon Seo。2022年。上下文生成式检索。arXiv abs/2210.02068（2022年）。[24] Jimmy Lin、Rodrigo Nogueira 和 Andrew Yates。2022年。用于文本排序的预训练 Transformer：BERT 及其后续模型。Springer Nature（2022年）。[25] Yiding Liu、Guan Huang、Jiaxiang Liu、Weixue Lu、Suqi Cheng、Yukun Li、Dait- ing Shi、Shuaiqiang Wang、Zhicong Cheng 和 Dawei Yin。2021年。用于百度搜索网页规模检索的预训练语言模型。发表于 SIGKDD。[26] Yuxiang Lu、Yiding Liu、Jiaxiang Liu、Yunsheng Shi、Zhengjie Huang、Shi Feng、Yu Sun、Hao Tian、Hua Wu、Shuaiqiang Wang、Dawei Yin 和 Haifeng Wang。2022年。ERNIE-Search：通过在线自蒸馏，将交叉编码器与双编码器结合，用于稠密段落检索。arXiv a bs/2205.09153（2022年）。[27] Sanket Vaibhav Mehta、Jai Gupta、Yi Tay、Mostafa Dehghani、Vinh Quang Tran、Jinfeng Rao、Marc-Alexander Najork、Emma Strubell 和 Donald Met- zler。2022年。DSI++：利用新文档更新 Transformer 记忆。arXiv abs/2212.09744（2022年）。[28] Donald Metzler、Yi Tay 和 Dara Bahri。2021年。重新思考搜索：让门外汉成为领域专家。发表于 SIGIR。[29] Arvind Neelakantan、Tao Xu、Raul Puri、Alec Radford、Jesse Michael Han、Jerry Tworek、Qiming Yuan、Nikolas A. Tezak、Jong Wook Kim、Chris Hallacy、Jo- hannes Heidecke、Pranav Shyam、Boris Power、Tyna Eloundou Nekoul、Girish Sastry、Gretchen Krueger、David P. Schnurr、Felipe Petroski Such、Kenny Sai-Kin Hsu、Madeleine Thompson、Tabarak Khan、Toki Sherbakov、Joanne Jang、Peter Welinder 和 Lilian Weng。2022年。通过对比式预训练生成文本和代码嵌入。arXiv abs/2201.10005（2022年）。[30] Jianmo Ni、Gustavo Hern'andez 'Abrego、Noah Constant、Ji Ma、Keith B. Hall、Daniel Matthew Cer 和 Yinfei Yang。2022年。Sentence-T5：基于预训练文本到文本模型的可扩展句子编码器。发表于 ACL 论文集。[31] Jianmo Ni、Chen Qu、Jing Lu、Zhuyun Dai、Gustavo Hern'andez ' Abrego、Ji Ma、Vincent Zhao、Yi Luan、Keith B. Hall、Ming-Wei Chang 和 Yinfei Yang。2021。大型双编码器是可泛化检索器。arXiv abs/2112.07899（2021）。[32] Rodrigo Nogueira、Wei Yang、Jimmy Lin 和 Kyunghyun Cho。2019。通过查询预测进行文档扩展。arXiv abs/1904.08375（2019）。[33] Fabio Petroni、Aleksandra Piktus、Angela Fan、Patrick Lewis、Majid Yazdani、Nicola De Cao、James Thorne、Yacine Jernite、Vassilis Plachouras、Tim Rock- taschel 和 Sebastian Riedel。2021。KILT：知识密集型语言任务基准。在 NAACL。[34] Yingqi Qu、Yuchen Ding、Jing Liu、Kai Liu、Ruiyang Ren、Xin Zhao、Daxiang Dong、Hua Wu 和 Haifeng Wang。2021。RocketQA：面向开放域问答的稠密段落检索优化训练方法。在 NAACL。[35] Colin Raffel、Noam M. Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter Liu。2020。借助统一的文本到文本转换器探索迁移学习的极限。JMLR（2020）。[36] Aditya Ramesh、Mikhail Pavlov、Gabriel Goh、Scott Gray、Chelsea Voss、Alec Rad- ford、Mark Chen 和 Ilya Sutskever。2020。零样本文本到图像生成。在 ICML。[37] Nils Reimers 和 Iryna Gurevych。2019。Sentence-BERT：使用孪生 BERT 网络的句子嵌入。在 EMNLP。[38] Stephen E. Robertson 和 Steve Walker。1997。关于相关信息有限时的相关性权重。在 SIGIR。[39] Stephen E. Robertson 和 Hugo Zaragoza。2009。概率相关性框架：BM25 及其拓展。《信息检索基础与趋势》（2009）。[40] Jason Tyler Rolfe。2017。离散变分自编码器。在 ICLR。[41] Keshav Santhanam、Omar Khattab、Jon Saad-Falcon、Christopher Potts 和 Matei Zaharia。2022。ColBERTv2：通过轻量级后期交互实现高效检索。在 NAACL。[42] Yi Tay、Vinh Quang Tran、Mostafa Dehghani、Jianmo Ni、Dara Bahri、Harsh Mehta、Zhen Qin、Kai Hui、Zhe Zhao、Jai Gupta、Tal Schuster、William W. Cohen 和 Donald Metzler。2022。作为可微搜索索引的 Transformer 记忆。在 NeurIPS。[43] Nandan Thakur、Nils Reimers、Andreas Ruckl'e、Abhishek Srivastava 和 Iryna Gurevych。2021。BEIR：用于零样本评估信息检索模型的异质基准。在 NeurIPS 数据集与基准赛道（第二轮）。[44] Aäron van den Oord、Oriol Vinyals 和 Koray Kavukcuoglu。2017。神经离散表示学习。在 NIPS。[45] Kexin Wang、Nandan Thakur、Nils Reimers 和 Iryna Gurevych。2022。GPL：用于稠密检索无监督领域自适应的生成式伪标签。在 NAACL。[46] Yujing Wang、Ying Hou、Hong Wang、Ziming Miao、Shibin Wu、Hao Sun、Qi Chen、Yuqing Xia、Chengmin Chi、Guoshuai Zhao、Zheng Liu、Xing Xie、Hao Sun、Weiwei Deng、Qi Zhang 和 Mao Yang。2022。用于文档检索的神经语料索引器。在 NeurIPS。[47] Lee Xiong、Chenyan Xiong、Ye Li、Kwok-Fung Tang、Jialin Liu、Paul Bennett、J unaid Ahmed 和 Arnold Overwijk。2021。用于稠密文本检索的近似最近邻负对比学习。在 ICLR。[48] Erxin Yu、Lan Du、Yuan Jin、Zhepei Wei 和 Yi Chang。2022。通过主题引导的离散潜变量学习语义文本相似度。arXiv abs/2211.03616（2022）。[49] Jingtao Zhan、Jiaxin Mao、Yiqun Liu、Jiafeng Guo、M. Zhang 和 Shaoping Ma。2022。通过约束聚类学习离散表示，实现高效的稠密检索。在 WSDM。[50] Tiancheng Zhao、Kyusong Lee 和 Maxine Eskénazi。2018。用于可解释神经对话生成的无监督离散句子表示学习。在 ACL。[51] Guoqing Zheng 和 Jamie Callan。2015。利用分布式表示学习词项重新加权。在 SIGIR。[52] Yujia Zhou、Jing Yao、Zhicheng Dou、Ledell Yu Wu、Peitian Zhang 和 Ji rong Wen。2022。Ultron：借助基于模型的索引器实现语料上的终极检索器。arXiv abs/2208.09257（2022）。[53] Shengyao Zhuang、Houxing Ren、Linjun Shou、Jian Pei、Ming Gong、G. Zuccon 和 Daxin Jiang。2022。通过查询生成弥合可微搜索索引中索引与检索之间的差距。arXiv abs/2206.10128（2022）。[54] Łukasz Kaiser、Aurko Roy、Ashish Vaswani、Niki Parmar、Samy Bengio、Jakob Uszkoreit 和 Noam M. Shazeer。2018。使用离散潜变量加速序列模型的解码。在 ICML。