arXiv:2301.04413v2 [cs.IR] 4 Jul 2024

CoSPLADE：用于对话信息检索的上下文化 SPLADE

NamLe Hai[1[0000−0002−9020−8790]]，Thomas Gerald[2]，Thibault Formal[1,3]，Jian-Yun Nie[4]，Benjamin Piwowarski[1[0000−0001−6792−3262]]和 Laure Soulier[1,2[0000−0001−9827−7400]]

[1] 索邦大学、法国国家科学研究中心、ISIR，法国巴黎 F-75005 first.last @sorbonne-universite.fr[2] 巴黎萨克雷大学、法国国家科学研究中心、巴黎萨克雷技术转移公司、LISN，法国奥赛 91405 first.last @lisn.upsaclay.fr[3] Naver 欧洲实验室，法国梅朗 first.last@naverlabs.com[4] 蒙特利尔大学，加拿大蒙特利尔 nie@iro.umontreal.ca

摘要。 对话式搜索是一项具有挑战性的任务，因为它旨在根据当前用户查询以及完整的对话历史检索文档。以往的大多数方法都采用多阶段排序方案，并依赖查询改写这一关键中间步骤，而这可能导致次优检索。其他方法则尝试使用完全神经化的第一阶段信息检索，但要么采用零样本方式，要么依赖基于带伪标签数据集的完整学习排序。在本研究中，我们利用 CANARD 数据集，提出一种创新且轻量的学习技术，用于训练基于 SPLADE 的第一阶段排序器。通过采用 SPLADE 稀疏表示，我们表明，当与基于 T5Mono 的第二阶段排序器结合时，该方法在 TREC CAsT 2020 和 2021 赛道上的结果具有竞争力。源代码可在 https://github.com/xpmir/cosplade 获取

关键词： 信息检索 · 对话式搜索 · 第一阶段排序。

1 引言

随着 Siri、Alexa 和 Cortana 等对话助手的出现，对话信息检索作为即席信息检索的一种变体，已成为一个重要的研究领域 [5,7]。在对话式信息检索中，搜索在一个会话内进行，用户的信息需求通过一系列查询来表达，与自然对话类似——因此，查询与响应之间存在复杂的相互依赖关系。

神经信息检索模型在对话式信息检索任务中表现最佳，这一点并不令人意外 [6,9]。大多数先前研究都依赖历史查询扩展步骤 [40], ，即一种会考虑所有历史查询及其相关答案的查询扩展机制。此类查询扩展模型是在 CANARD 数据集 [10], 上训练的，该数据集由一系列问题及其

相关答案以及一个消歧查询组成，本文将其称为 标准查询 。然而，正如 [16,19] 所强调的，依赖改写步骤计算成本高昂，而且可能并非最优。Krasakis 等人 [16]提出以零样本 [15] 方式使用 ColBERT，即用查询序列替代查询，且不对模型进行任何训练。Lin 等人 [19] 提出学习查询历史的稠密上下文化 表示，并在由弱标签构成的数据集上优化学习排序损失。这使得训练过程复杂（标签并不可靠）且耗时较长。

在这项工作中，我们沿着这一研究方向展开，但为第一阶段排序器提出了一种轻量得多的训练流程：我们专注于查询，不使用任何段落，因此也不进行学习排序训练。此外，这种方法还避开了从 CANARD 数据集中推导弱标签的问题[5]。有了这种强监督，我们便可以考虑更多上下文——也就是说，我们使用系统提供给用户的答案，从而更好地对查询进行上下文化，这一点已在我们的实验中得到验证。我们提出的训练损失函数利用了 SPLADE 模型提供的查询和文档稀疏表示 [11]。简而言之，我们要求查询的表示与消歧查询（即 标准查询）的表示相匹配。我们的第一阶段排序器取得了很高的性能，尤其是在召回率方面——这是多阶段方法中最重要的指标，其表现可与 TREC CAsT [9]，中的最佳系统相媲美；在精确率导向指标上也表现出色，这表明了我们方法的潜力。

最后，为了取得良好效果，第二阶段排序器（即重排序器）也需要考虑对话，这可能需要借助一套启发式方法来选择部分内容，和/或进行查询改写，例如 [21] 中使用的方法。利用第一阶段排序器输出 BERT 词汇表中各词权重这一特点，我们提出了一种简单机制，通过 SPLADE 选出的关键词向重排序器提供对话上下文。

总而言之，我们的贡献如下：

1. 我们提出了 CoSPLADE（上下文化 SPLADE）模型，该模型基于一种新的损失函数来优化第一阶段排序器，从而实现轻量级训练策略，并在召回率方面取得最先进的结果；

2. 我们表明，将基于第一阶段 SPLADE 查询表示所提取上下文的第二阶段排序器与该模型结合后，在 TREC CAsT 2020 和 2021 中取得的结果可与最佳方法相媲美。

2 相关工作

TREC 对话辅助检索赛道（$CAsT$）首届赛事于 2019 年举办， [6] 提出了对话式搜索这一新挑战。其基本原则如下：用户以自然语言向系统提出问题

[5] 请注意，在第二阶段，我们依赖弱标签，因为我们的模型与以往的研究类似。鉴于第一阶段与第二阶段排序器之间的差距持续缩小，未来可能不再需要训练第二阶段排序器。

，并在每次提问后获得系统的答复。与传统搜索系统不同的是，纳入先前的发言（无论是查询还是答案）是更好地理解用户意图的关键。在对话式信息检索中，尤其是在 TREC $CAsT$ [7,6,9] 中，文档集合规模庞大，因此必须设计高效（且有效）的搜索系统。

对话式信息检索与对话式问答密切相关，因为二者都包含自然语言交互轮次。[29,31,30] 然而，两者的目标本质上不同。在对话式问答中，主题或上下文（即包含答案的段落）是已知的；而对话式信息检索旨在从海量文档集合中检索信息，其主题可能更具探索性。基于这一点，以下内容将聚焦于对话式信息检索的文献综述。

我们可以将对话式搜索领域的研究分为两条路线。第一条路线[35,36,38,4]侧重于上下文查询改写（CQR），以生成一个（普通或词袋形式的）查询，理想情况下，该查询能够表示脱离上下文的信息需求，并将其输入检索模型。一种 CQR 策略是从先前的话语中选取关键词，依据 word2vec 相似度、 [35], 基于 BM 的词项重要性25 [22], 或分类模型 [36]对关键词进行加权。其他方法 [17,22,21,39,34] 则利用生成式语言模型（例如 GPT-2 或 T5）的能力改写查询。这类方法尤为有效，在 TREC CAsT 2020 赛事中取得了顶尖成绩。 [6]查询改写模型在所选证据来源方面也各不相同。有些模型关注对话的早期阶段， [1],有些关注经过启发式方法 [2] 或分类模型[24], 筛选的一组查询，还有些则同时关注先前的查询和文档。 [37]最后，为避免生成单个查询的问题， [17,23] 有研究提出使用多个生成的查询，并对返回的文档进行聚合。

然而，改写步骤是一个瓶颈，因为无法保证“标准查询”是最优的，因此也无法保证其具有良好的泛化能力 [19,16]。此外，生成文本耗时较长。为避免这些问题，另一类研究旨在将对话历史直接整合到检索模型中，从而绕过查询改写步骤。据我们所知，只有少数研究在对话式搜索中采用了这一路径。Qu 等人 [27] 利用对话中最近的 k 个查询计算查询表示 [18]。类似地，Lin 等人 [19] 对整个查询历史中的上下文化词元嵌入取平均值。该表示通过在带有弱标签的集合上优化学习排序损失来学习，这需要格外谨慎，以确保良好的泛化能力。最后，Krasakis 等人 [16] 使用了一种词汇性更强的神经模型，即 ColBERT [15],，用它对查询及其上下文进行编码，但完全没有对其进行微调。在本研究中，我们进一步采用稀疏模型 SPLADE [11], ，使用专为此类稀疏表示设计的新型损失函数，并采用一种轻量级训练过程，该过程不依赖段落，而只需包含改写后查询的数据集。

3 模型

在 TREC CAsT [6,9]，中，每个检索会话包含约 10 轮对话。每一轮对应一个查询，其对应的规范答案 [6] 会作为后续查询的上下文。现在介绍一些用于描述模型的符号。对于每一轮 $n \leq N$，其中 N 是对话的最后一轮，我们分别用 q_n 和 a_n 表示相应的查询及其回答。最后，轮次 n 中查询 q_n 的上下文包含此前的所有查询和答案，即 q_1、a_1、q_2、a_2、...、q_{n-1}、a_{n-1}。$TREC\ CAsT$ 挑战的主要目标是针对每个查询 q_n 及其上下文，检索相关段落。在接下来的章节中，我们将介绍第一阶段排序器和第二阶段重排序器，以及二者的训练过程；这两种排序器都直接或间接地基于 [11] 中所述的 SPLADE (v2) 模型。SPLADE 在领域内集合上的结果与稠密方法不相上下，同时在零样本设置下展现出更强的泛化能力 [11]。此外，它能在 BERT 词汇表中输出文档或查询的稀疏表示，这一点对于我们模型的训练和推理至关重要。因此，我们没有考虑 ColBERT [33] 等交互模型，也没有考虑稠密方法 [14]。我们使用的 SPLADE 模型包含一个上下文编码函数，后接若干聚合步骤：ReLU、对数饱和，以及对文本中各词元进行最大池化。SPLADE 的输出是一个稀疏向量，其在 BERT 词汇空间中仅包含正值或零值分量 $\mathbb{R}^{|V|}$。在本研究中，我们为同一 SPLADE 架构使用了多组参数，并通过参数 θ 区分各个版本；相应的模型记为 $SPLADE$（...；θ）。

3.1 第一阶段

原始 SPLADE 模型 [11] 通过文档的稀疏表示 (d) 与查询的稀疏表示 ($\hat{}q$) 之间的点积来为文档打分：

$$s(\hat{q}, \hat{d}) = \hat{q} \cdot \hat{d} \tag{1}$$

在我们的工作中，与 [19]，一样，我们假设文档表示已经在标准的即席信息检索任务上得到了充分优化。因此，文档嵌入 d 由预训练 SPLADE 模型生成，即 $d = SPLADE$([CLS] d; θ_{SPLADE})，其中 θ_{SPLADE} 是从 HuggingFace[7] 获取的原始 SPLADE 参数。这些参数在训练过程中不会进行微调。因此，我们可以使用基于原始 SPLADE 文档表示构建的标准索引，高效地检索排名前 k 的文档。接下来，我们将介绍如何利用对话历史对查询表示进行上下文化处理。随后，我们将详细说明训练损失函数，其目标是缩小标准查询表示与上下文化表示之间的差距。

[6] 由组织者选定为基线系统中最相关的答案。[7] 权重可在 https://huggingface.co/naver/splade-cocondenser-ensembledistil 查看

查询表示。与最先进的第一阶段对话排序方法 [19,16], 类似，我们利用先前的查询对当前查询进行上下文化处理。进一步地，我们提出在查询表示过程中纳入答案；借助我们的轻量级训练，这一过程更容易实现。

为了利用这两种上下文，我们采用一个简单的模型：第 n 轮的上下文查询表示记为 $\hat{q}_{n,\ k}$，它由两种表示组合而成：$\hat{q}_n^{queries}$ 编码当前查询在所有先前查询上下文中的信息，$\hat{q}_{n\ k}^{answers}$，则编码当前查询在 k 先前答案[8]上下文中的信息。形式化地，上下文化后的查询表示 $\hat{q}_{n,\ k}$ 为：

$$\hat{q}_{n,k} = \hat{q}_n^{queries} + \hat{q}_{n,k}^{answers} \tag{2}$$

其中，我们使用两个版本的 SPLADE，分别由 $\theta_{queries}$ 参数化，用于完整查询历史；由 $\theta_{answers,\ k}$ 参数化，用于答案。这些参数通过优化式（8）中定义的损失函数进行学习。

按照 [19], ，我们将 $\hat{q}_n^{queries}$ 定义为通过编码当前查询与所有先前查询的串联而得到的查询表示：

$$\hat{q}_n^{queries} = SPLADE([\texttt{CLS}]\ q_n\ [\texttt{SEP}]\ q_1\ [\texttt{SEP}]\ \ldots\ [\texttt{SEP}]\ q_{n-1}; \theta_{queries}) \tag{3}$$

使用一组特定参数 $\theta_{queries}$。

为了将用户能够获取的答案纳入考虑，我们需要将其加入表示中。按照先前的工作 [2], ，我们可以考虑不同数量的答案 k；具体而言，可以选择 $k = 1$（最后一个答案），也可以选择 $k = n - 1$（所有先前答案）。形式化地，表示 $\hat{q}_{n\ k}^{answers}$，由查询-答案对的表示取均值得到。这使我们能够规避语言模型施加的词元长度限制。形式化地，

$$\hat{q}_{n,k}^{answers} = \frac{1}{k} \sum_{i=n-k}^{n-1} SPLADE(q_n\ [\texttt{SEP}]\ a_i; \theta_{answers,k}) \tag{4}$$

训练 综上，训练旨在为最后提出的查询 q_n获取良好的表示 $\hat{q}_n$，也就是利用先前的查询和答案对 q_n 进行上下文化。为此，我们可以利用标准查询 q_n^*，它（希望如此）是一个经过上下文化且含义明确的查询。我们可以使用原始 SPLADE 模型计算该查询的表示 $\hat{q}_n^*$，即

$$\hat{q}_n^* = SPLADE(q_n^*; \theta_{SPLADE}) \tag{5}$$

例如，对于查询“他多大了？”，相应的标准查询可以是“奥巴马多大了？”。SPLADE 给出的后者的表示如下：

[(”$Obama$” 1.5), (”$Barack$” 1.2), (”age” 1.2), (”old” 1.0), (”$president$” 0.8) ...]

[8] 在实验中，我们还探索了另一种模型，在这种模型中，答案和查询会同时纳入考量。

其中，“奥巴马”和“巴拉克”这两个词明显与当前查询相关的其他词语（“年老”以及语义相关的“年龄”）一同出现。

现在，我们可以定义训练目标：缩小黄金查询表示 $\hat{q}_n^*$ 与模型计算出的表示 $\hat{q}_{n,\ k}$ 之间的差异。一个显而易见的损失函数选择是使用余弦损失来匹配预测表示和黄金表示（因为缩放查询不会影响排序）。然而，正如结果部分所示，我们通过实验发现，采用经过修改的均方误差损失函数能取得更好的结果，其第一部分是标准均方误差损失：

$$Loss_{MSE}(\hat{q}_{n,k}, \hat{q}_n^*) = MSE(\hat{q}_{n,k}, \hat{q}_n^*) \tag{6}$$

在实验中，我们观察到，使用直接均方误差进行训练的模型无法很好地捕捉上下文中的词语，尤其是答案中的词语。原因在于，人工改写的标准查询通常只包含先前轮次中的少数几个附加词语，这些词语可由最后一个查询直接推知。答案中的其他潜在有用词语可能不会被纳入其中。这是一种保守的扩展策略，对于自动查询改写过程而言，未必是最好的范例。因此，我们加入了非对称均方误差，旨在鼓励从先前答案中进行术语扩展，同时通过将术语限制为标准查询 q_n^* 中已有的词语来避免引入噪声。形式上，我们的非对称损失函数为：

$$Loss_{asym}(\hat{q}_{n,k}^{answers}, \hat{q}_n^*) = \left(\max(\hat{q}_n^* - \hat{q}_{n,k}^{answers}, 0)\right)^2 \tag{7}$$

其中最大值是逐分量取的。因此，该损失函数会推动答案偏置表示 $\hat{q}_{n\ k}^{answers}$，纳入标准查询中的词元。与均方误差（MSE）不同，它不会（直接）对 $\hat{q}_{n\ k}^{answers}$表示的各分量施加上界——这会通过下文所述的最终损失函数间接实现。

我们优化的最终损失函数是上述损失函数的简单线性组合，只需计算两个查询表示：

$$Loss(\hat{q}_{n,k}, \hat{q}_n^*) = Loss_{MSE}(\hat{q}_{n,k}, \hat{q}_n^*) + Loss_{asym}(\hat{q}_{n,k}^{answers}, \hat{q}_n^*) \tag{8}$$

全局损失的两个组成部分相互作用。更确切地说，$Loss_{asym}$ 会推动 $\hat{q}_{n\ k}^{answers}$表示尽可能匹配标准查询表示 $\hat{q}_n^*$。而 $Loss_{MSE}$ 则会推动查询偏置表示 $\hat{q}_{n,\ k}$ 在无法匹配时进行补偿。因此，该方法着重从先前答案中提取信息；实验表明，这样做很有益处。

实现细节。对于第一阶段，我们使用预训练的 SPLADE 即席检索模型权重初始化两个编码器（一个对查询进行编码，另一个对先前的答案进行编码）。我们使用 ADAM优化器，训练批次大小为 16；第一个编码器的学习率为 2e-5，第二个为 3e-5。我们仅在 CANARD 数据集上微调 1 个 epoch。

3.2 重排序

我们使用 T5Mono [25] 方法进行重排序，通过第一阶段排序器识别出的关键词来扩充原始查询 q_n。我们的设想是，这些词应能捕捉到为原始查询提供上下文所需的信息。对话轮次 n 的扩充查询 q_n^+ 如下：

$$q_n^+ = q_n.\ Context: q_1\ q_2\ \ldots\ q_{n-1}.\ Keywords: w_1, w_2, ..., w_K \tag{9}$$

其中，w_i 是我们利用第一阶段排序器选出的最重要的前 K 个词，具体方法如下。首先，为了减少噪声，我们只考虑出现在任意查询 q_i 或相关答案 a_i 中的词（针对 $i \leq n-1$）。其次，我们根据组成该词的词元中最大的 SPLADE 权重对词进行排序。[9]

我们将针对该输入进行微调的 T5 模型记为 $T5^+$。与原论文 [25]，一样，文档 d 对查询 q_n 的相关性得分，是在给定提示词 pt(q_n^+, d) = "Query: q_n^+. Document: d. Relevant:" 时生成词元 "true" 的概率：

$$score(q_n^+, d; \theta) = \frac{p_{T5}(\texttt{true}|\text{pt}(q_n^+, d); \theta)}{p_{T5}(\texttt{true}|\text{pt}(q_n^+, d); \theta) + p_{T5}(\texttt{false}|\text{pt}(q_n^+, d); \theta)} \tag{10}$$

其中，θ 是 T5Mono 模型的参数。

与第一阶段训练不同，我们通过对齐文档的得分来微调排序器，而非对齐查询的权重（显然，使用 T5 模型无法做到这一点）。这里，文档的"标准"得分是使用原始 T5Mono 和标准查询 q_n^* 计算的。T5 模型使用原作者公开的权重进行初始化[10]，，记为 θ_{T5}。更具体地说，我们使用 MSE-Margin 损失 [13] 微调预训练的 T5Mono 模型。重排序器在对话轮次 n、给定文档 d_1 和 d_2 时的损失函数计算如下：

$$\mathcal{L}_R = \left[\left(s(q_n^+, d_1; \theta_{T5+}) - s(q_n^+, d_2; \theta_{T5+})\right) - \left(s(q_n^*, d_1; \theta_{T5}) - s(q_n^*, d_2; \theta_{T5})\right)\right]^2$$

我们通过保留原始内容来优化 θ_{T5+} 参数，以评估标准查询的得分 θ_{T5}。

实现细节。我们将 θ_{T5+} 初始化为 θ_{T5}，并进行 3 个 epoch 的微调，批次大小为 8，学习率为 1e-4。我们使用第一阶段排名前 1000 的文档采样样本对 (d_1, d_2)：从排名前三的文档中采样 d_1，从其余 997 篇文档中采样 d_2，以促使模型关注分数中的重要差异。

[9] 为提高连贯性，我们选择让关键词按照其在上下文中的出现顺序排列，但没有改变这一实验设置。[10] 我们使用了 HuggingFace 检查点 https://huggingface.co/castorini/monot5-base-msmarco

4 实验方案

我们设计了评估方案，以满足两个主要评估目标：(i) 分别评估 CoSPLADE 模型第一阶段和第二阶段排序组件的效果；(ii) 将 CoSPLADE 模型的效果与 TREC CAsT 2020 和 2021 年的参赛系统进行比较。

4.1 数据集

为训练模型，我们使用了 CANARD语料库[11], ，这是一个专注于基于上下文进行查询改写的对话数据集。具体而言，CANARD 数据集由一系列对话历史组成，每段历史包含一系列查询、简短答案（由人工撰写）和改写后的查询（已结合上下文）。训练集、开发集和测试集分别包含 31.538、3.418 和 5.571 条结合上下文的改写后查询。

为评估模型，我们使用了 TREC CAsT 2020 和 2021 数据集，分别包含 25 和 26 个信息需求（主题），以及一个由 MS MARCO数据集、来自 KILT基准的维基百科更新转储和《华盛顿邮报》V4文档集组成的文档集合。每个主题都提供一段对话，其中问题与响应交替出现（从文档集合中人工挑选的段落，即规范答案）。对于每个问题（总计分别为 216 和 239 个），数据集提供其人工改写形式，以及一组约 20 篇相关文档。我们使用前者来定义上限基线（**SPLADE_黄金查询**）。

4.2 指标与基线

我们采用了 TREC CAsT 2020 和 2021 中使用的官方评估指标，即 nDCG@3、平均倒数排名、Recall@X、MAP@X 和 nDCG@X，其中 CAsT 2020 的截断值设为 1000，CAsT 2021 的截断值设为 500。对于每项指标，我们计算数据集中不同查询的性能均值和方差。基于此，下面介绍用于评估模型各个组件的不同基线和场景。

第一阶段排序场景。 为评估我们的第一阶段排序模型（第 3.1 节）的有效性，我们将基于公式 (2) 中查询表示的 CoSPLADE 方法与不同变体进行比较（在我们的所有实验中，文档编码器均设为原始 SPLADE 编码器）：**SPLADE_原始查询** （下界）：SPLADE [12] 仅使用原始且存在歧义的用户查询 q_n；**SPLADE_黄金查询** （某种意义上的上界）：SPLADE 使用人工改写的查询 q_n^*；**CQE** [19]，一种最先进的稠密上下文化查询表示，通过在带有伪标签的数据集上进行学习排序来学习。前两者旨在评估.....程度，然而

[11] https://sites.google.com/view/qanta/projects/canard

我们的模型能够捕捉上下文信息；后者是 TREC CAsT 的参赛系统，与我们的工作密切相关。

为使用 $\hat{q}_{n\,k}^{answers}$, 表示查询时对答案进行建模，我们采用了两种历史范围（**“全部”** $k = n-1$ 个答案，以及**“最后一个”**，即仅使用最后一个答案 $k = 1$），并使用三种类型的答案输入：**答案**，其中答案为规范答案；**短答案**，其中按照表现最佳的 TREC CAsT 方法过滤句子 [21]。这样可以保持输入长度一致，但代价是丢失信息；**长答案** 由于 CANARD 中的答案较短（从维基百科提取的几个句子，与 CAsT 中的答案不同），我们对其进行扩展，以缩小训练与推理之间的差异。对于每个句子，我们找到它所在的维基百科段落（如果该段落存在于 ORConvQA [26] 中），并从中抽取由 3 个相邻句子组成的短片段。

最后，我们还通过修改历史上下文的使用方式或训练损失函数，对上述表现最佳的变体进行了消融研究：**flatContext** 这是我们 SPLADE 方法的单编码器版本，将上下文中的所有信息拼接起来，再应用 SPLADE，以获得查询的单一表示（而非公式 2 和 3 中的两个表示 $\hat{q}_n^{queries}$和 $\hat{q}_{n\,k}^{answers}$,）；由于不再有两个表示，该版本使用均方误差损失函数进行训练（公式 6）。**MSE** 这是使用均方误差损失（公式 6）而非所提出的损失函数（公式 8）训练的 SPLADE 方法版本；**余弦** 这是使用余弦损失而非所提出的损失函数（公式 8）训练的 SPLADE 方法版本。余弦损失很有意义，因为对于保持文档排序不变的缩放因子（公式 1），它具有不变性。

第二阶段排序场景。我们考虑第二阶段排序模型的不同场景：**T5Mono_原始查询**将 T5Mono 排序模型 [25]应用于原始查询；**T5Mono GoldQuery**将 T5Mono 排序模型¯应用于标准查询；**T5Mono CQR**将 T5Mono 排序模型应用于¯使用预训练T5（基于 CANARD 数据集）生成的查询改写；**CoSPLADE_[上下文]_[数量]**：第二阶段排序模型输入（公式 9）的不同版本，其中变化因素包括：1）改写中是否包含历史查询；2）第一阶段排序器识别为相关的关键词数量：5、10、20。

TREC参与者基线。对于每次评测活动（2020年和2021年），我们还将模型与两个概述中列出的 TREC CAsT 参赛者中的最佳者、中位数水平者和最低者进行比较 [6,9], ，其中参赛者按照 nDCG@3指标排名。请注意，由于我们报告的是 TREC概述 [8,9]中提供的有效性指标，因此无法判断结果是否具有统计显著性。

	@500召回率 @500平均准确率		MRR	@500归一化折损累计增益 nDCG@3	
基线					
SPLADE _原始查询	30.8±2.7	5.5±0.9	21.3±2.9	17.8±1.8	13.1±2.1
SPLADE _黄金查询	68.8±2.0	16.1±1.2	55.5±3.3	42.8±1.7	38.3±2.8
CQE [20]来自 [9]	79.1	28.9	60.3	55.7	43.8
答案处理的影响：CoSPLADE _...					
所有答案	79.5±2.2	28.8±1.7	61.7±3.1	55.3±2.0	46.5±2.9
所有答案-简短	72.8±2.6	25.7±1.9	54.4±3.3	49.5±2.3	40.1±3.0
所有答案-详细	80.4±2.1	29.3±1.8	62.0±3.2	55.6±2.1	**48.9±3.0**
最终答案	83.4±2.0	31.2±1.8	61.8±3.1	58.1±2.0	47.4±3.0
最后一个短答案	79.2±2.2	28.1±1.8	61.4±3.3	54.3±2.1	46.4±3.0
LastAnswer-long	**85.2±1.8**	**32.0±1.7**	**64.3±03.0**	**59.4±1.9**	48.6±3.0
CoSPLADE _LastAnswer-long 变体					
flatContext	77.0±2.0	26.0±2.0	55.0±3.0	52.0±2.0	42.0±3.0
均方误差损失	70.9±2.4	21.6±1.7	48.7±3.4	45.2±2.3	36.9±3.1
余弦损失	70.4±2.5	22.6±1.7	52.5±3.3	46.9±2.2	39.0±3.0

表1. 我们的第一阶段排序模型在 TREC CAsT 2021 不同场景下的效果。

5 结果

5.1 第一阶段排序效果

本节重点介绍 CoSPLADE 模型的第一阶段排序组件。为此，我们设计了不同的实验场景，旨在评估所设计的损失函数（公式 8）以及证据来源的建模方式和效用（公式 3 和 4）所带来的影响。表1给出了这些不同基线和实验场景在 TREC CAsT 2021 数据集上的结果；在 CAsT 2020 上也观察到了类似趋势。详细结果（查询级别）可在 GitHub 仓库中查看。

总体而言，可以看出，我们方法的所有变体（CoSPLADE_* 模型）都优于在原始查询上应用初始版本 SPLADE 的场景，更重要的是，也优于在标准查询上应用该模型的场景。这令人十分鼓舞，因为后一种场景可以视为预言机，即查询经过了人工消歧。最后，在所有指标上，我们的结果都优于 CQE [19]，这表明我们简单的学习机制与 SPLADE 相结合，能够实现最先进性能。

更好地利用查询与答案历史，可以为当前查询提供更充分的上下文。flatContext 场景的结果（相对于 SPLADE_黄金查询）让我们能够比较与对话相关的证据来源所产生的影响，因为两者使用相同的架构（SPLADE）。我们可以观察到，它的结果优于 SPLADE_黄金查询（例如，Recall@500 指标为 77，而后者为 68.8），这凸显了上下文对于更好地理解信息需求的作用。

答案越详细，效果越好。由于答案比问题更冗长，将其纳入其中会更复杂，因此我们需要研究不同的可能性（CoSPLADE_AllAnswers* 和 CoSPLADE_LastAnswer*）。我们可以

看到：1）将答案精简为几个关键词（*-short）的效果不如考虑规范答案，但 2）与对应的维基百科段落（*-long）结合时，可能会有一定效果。此外，与考虑对话历史中的全部答案相比，仅考虑最后一个答案似乎更有效[12]。综合来看，这些观察凸显了在改写过程中纳入答案信息的方式至关重要。

采用非对称损失的双查询表示利用了稀疏查询表示。flatContext 场景的结果表明，同时考虑历史查询和答案的效果更好（与均方误差损失场景相比，两者可以直接比较）。然而，如果将表示分开并采用非对称损失函数，结论就会改变。此外，将我们的最佳场景 CoSPLADE_LastAnswer-long 与仅使用均方误差或余弦损失训练的类似场景进行比较，揭示了非对称均方误差（公式 7）的有效性。请注意，这种非对称损失会促使模型在查询编码时考虑先前的答案。这进一步印证了我们的直觉：对话上下文，尤其是冗长的答案，对于对话式搜索任务十分重要。这也表明，上下文应在架构的不同层面（输入和损失函数）中纳入考虑。

5.2 第二阶段排序效果

本节中，我们使用 CoSPLADE_LastAnswer-long 模型作为第一阶段排序器，并评估基于 T5Mono 模型的第二阶段排序方法的不同变体。为进行公平比较，我们还列出了将 T5Mono 排序模型应用于原始查询和标准查询，以及应用于经 T5 生成模型改写的查询时所得的结果。TREC CAsT 2021 数据集上的结果见表 2。

对 CoSPLADE 模型不同变体的分析揭示了关于上下文可用性以及添加到查询中的关键词数量的不同观察结果。首先，在提示词中将先前的问题加入当前查询（即“上下文”），似乎能够改善查询理解，从而对检索效果产生积极影响。例如，添加 5 个关键词时，使用上下文可使 nDCG@3 达到 51.5%，而不使用上下文时为 45.9%。其次，随着额外关键词数量的增加，性能往往会提升，尤其是在不使用上下文的场景中，这合乎情理。对于使用上下文的场景，这一趋势则不太明显，因为最高分数分别出现在使用 10 个或 20 个关键词时。值得注意的是，添加 10 或 20 个关键词比仅使用 5 个关键词更有效（例如，nDCG@3指标为 54.4% 对 51.5%）。因此，关键词似乎有助于重述初始信息需求，但当关键词过多并与其他信息结合时，效果可能会趋于饱和。

[12] 这可能是因为使用先前答案的方式过于简单，即式（4），但我们尝试的其他所有变体都没有取得更好的效果

	@500召回率	@500平均准确率	MRR	@500归一化折损累计增益 nDCG@3	
基线					
T5Mono _原始查询	78.4±2.3	21.0±1.8	39.6±3.2	45.9±2.1	28.4±3.0
T5Mono _黄金查询	86.1±1.7	44.1±1.9	78.7±2.7	68.5±1.8	64.6±2.8
T5Mono _CQR	80.4±2.2	30.0±1.9	58.2±3.4	55.3±2.1	44.6±3.2
基于 CoSPLADE 的第二阶段变体					
CoSPLADE _无上下文 _5	84.3±1.8	31.7±2.0	61.6±3.3	58.1±2.0	45.9±3.1
CoSPLADE _无上下文 _10	83.1±1.9	32.0±1.7	66.0±3.1	59.1±1.9	49.8±2.9
CoSPLADE _无上下文 _20	84.8±1.7	33.4±1.8	66.0±3.0	60.4±1.8	49.6±2.9
CoSPLADE _上下文 _5	**85.0±1.7**	35.0±1.8	68.4±3.0	61.7±1.9	51.5±02.9
CoSPLADE _上下文 _10	84.8±1.7	**36.5±1.9**	67.8±3.1	**63.0±1.9**	53.3±3.1
CoSPLADE _上下文 _20	84.9±1.7	35.5±1.8	**69.8±3.0**	62.2±1.9	**54.4±2.9**

表2. 我们的第二阶段排序模型在 TREC CAsT 2021 不同场景下的效果。

TREC CAsT 2020	Recall@1000	MAP@1000	平均倒数排名	nDCG@1000	nDCG@3
TREC 参赛者（最佳）	63.3	30.2	59.3	52.6	45.8
TREC 参赛者（中位数）	52.1	15.1	42.2	36.4	30.4
TREC 参赛者（较低）	27.9	1.0	5.9	11.1	2.2
CoSPLADE	82.4±2.0	26.9±1.5	58.1±2.9	54.2±1.8	44.0±2.7
TREC CAsT 2021	@500召回率	@500平均准确率	MRR	@500归一化折损累计增益	nDCG@3
TREC 参赛者 1（最佳）	85.0	37.6	67.9	63.6	52.6
TREC 参赛者 2（中位数）	36.4	17.6	53.4	33.6	37.7
TREC 参赛者 3（较低）	58.9	7.6	27.0	31.4	15.4
CoSPLADE	84.9±1.7	35.5±1.8	69.8±3	62.2±1.9	54.4±2.9

表 3. TREC CAsT 2020 和 2021 年参赛者的表现

将表现最佳的模型场景与更基础的场景进行比较——后者在原始查询和标准查询上应用 T5Mono 第二阶段排序器——我们可以发现，我们的方法能够提升初始查询的检索效果，但仍不足以达到 T5Mono_GoldQuery 的性能。然而，在由 T5 重述的查询上应用 T5Mono 后得到的结果表明，对初始查询进行语境化是一项困难的任务。事实上，T5Mono_CQR 场景的效果不如 T5Mono_GoldQuery；具体而言，不同指标的分数相差 6 至 20 个百分点。

此外，值得注意的是，在原始查询和标准查询上应用的 SPLADE 模型（表1中的第一阶段排序）所得结果，低于在相同数据上使用的 T5Mono 模型（表2中的第二阶段排序）。这可以用两种架构的不同用途来解释：SPLADE 是一种稀疏模型，侧重于查询/文档扩展；而 T5Mono 则专门用于提高精确率。不过，值得指出的是，在实验评估中，将 SPLADE 和 T5Mono 分别用作第一阶段和第二阶段排序器，取得了最高的效果。结果证明，CoSPLADE 方法既能为查询提供上下文信息，也能有效地对文档排序。

5.3 与 TRECCAsT 参赛者相比的有效性

最后，我们将本方法与 TREC CAsT 参赛者在 2020年和 2021年评测活动中的表现进行比较（表3）。在这两年中，我们的表现都与最佳参赛者十分接近，甚至更胜一筹。具体而言，在 2020年评测活动中，CoSPLADE 在 Recall@1000 和 nDCG@1000 方面优于最佳 TREC 参赛者。2021年，我们的模型在平均倒数排名和 nDCG@3指标方面取得了更好的结果。这两年的最佳参赛者都是 h2oloo 团队 [21,9]，他们使用了基于 T5 的查询改写技术。我们的结果表明，采用稀疏第一阶段排序模型的方法，可以在单个模型中结合查询扩展与文档排序的优势，进而帮助最终的重排序步骤。换言之，如果只改写查询，却不通过联合学习进行文档排序，可能会损害搜索任务的整体表现。

5.4 效率

第一阶段排序器基于 SPLADE（一种稀疏检索模型），因此效率很高。关于浮点运算次数的详细分析见 [11]。由于我们的损失函数不包含用于保持稀疏性的正则化损失，因此值得考察其变化情况，并与 SPLADE 进行比较。非零项的平均数量仅在我们的所有答案变体中有所增加（从 gold/raw 的约 60 增至 80），这表明未来我们或许可以通过控制稀疏性来提高 CoSPLADE 的效率。有趣的是，LastAnswers 变体的稀疏性略高，且表现更好。

6 结论

本文展示了如何将稀疏检索神经信息检索模型 SPLADE [11], 与轻量级学习过程结合，从而获得最先进的第一阶段排序器。我们还展示了这一第一阶段排序器可用于为第二阶段排序器提供上下文，其结果可与表现最佳的系统相媲美。未来的工作可以探索更好地捕捉上下文信息的策略，或显式处理评测数据集中包含的用户反馈。我们还计划在其他对话式问答数据集上评估该方法，例如 CoQA[32], 、OR-ConvQA [28], 或 ConvMix [3]。

7 致谢

我们谨此感谢 ANR 通过以下资助项目支持本研究：ANR JCJC SESAMS (ANR-18-CE23-0001) 和 ANR COST (ANR-18-CE23- 0016-003)。本研究获准使用 IDRIS 的高性能计算资源，分配编号为 AD011012847。

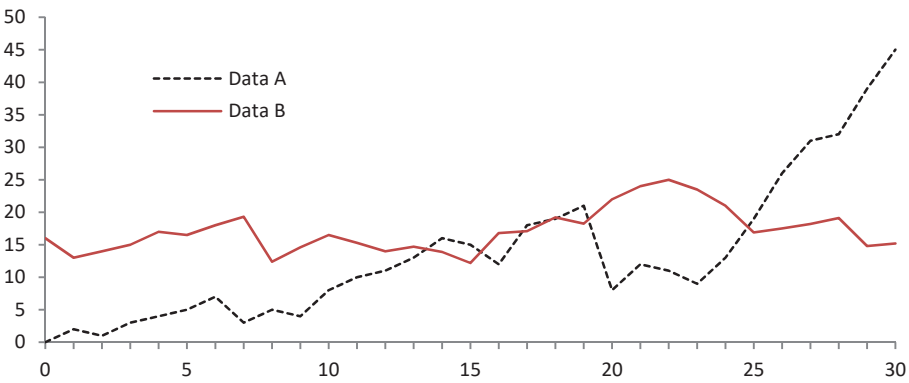
参考文献

1. Aliannejadi, M., Chakraborty, M., Ríssola, E.A., Crestani, F.：利用多轮对话的演变实现有效的答案检索，第 33–42 页。https://doi.org/10.1145/3343413.3377968, http://arxiv.org/abs/1912 10554 2 . . Arabzadeh, N., Clarke, C.L.A.：Waterlooclarke 在 TREC 2020 对话助手赛道中的表现 (2020) 3. Christmann, P., Roy, R.S., Weikum, G.：异构来源上的对话式问答。载于：Amigó, E., Castells, P., Gonzalo, J., Carterette, B., Culpepper, J.S., Kazai, G. (编.) SIGIR '22：第 45 届国际 ACM 信息检索研究与开发会议，西班牙马德里，2022 年 7 月 11 日至 15 日。第 144–154 页。ACM (2022)。https://doi.org/10.1145/3477495.3531815, https://doi.org/10 3531815 4 1145 3477495 . / . . Clarke, C.L.A.：Waterlooclarke 在 TREC 2019 对话助手赛道中的表现。载于：Voorhees, E.M., Ellis, A. (编.) 第二十八届文本检索会议论文集，TREC 2019，美国马里兰州盖瑟斯堡，2019 年 11 月 13 日至 15 日。NIST 特别出版物，第 1250 卷。美国国家标准与技术研究院 (NIST) (2019), https://trec.nist.gov/pubs/trec28/papers/WaterlooClarke.C.pdf 5. Culpepper, J.S., Diaz, F., Smucker, M.D.：信息检索研究前沿：第三届洛恩信息检索战略研讨会报告 (SWIRL 2018)。SIGIR Forum **52**(1), 34–90 (2018)。https://doi.org/10.1145/3274784.3274788, https://doi.org/10 3274788 6 1145 3274784 2020 . / . . Dalton, J., Xiong, C., Callan, J.：CAsT：对话辅助赛道概述，第 10 页 7. Dalton, J., Xiong, C., Callan, J.：TREC CAsT 2019：对话辅助赛道概述 http://arxiv.org/abs/2003 13624 8 2019. . Dalton, J., Xiong, C., Callan, J.：TREC CAsT：对话辅助赛道概述 http://arxiv.org/abs/2003 13624 9 2021. . Dalton, J., Xiong, C., Callan, J.：TREC CAsT：对话辅助赛道概述，第 7 页 (2021) 10. Elgohary, A., Peskov, D., Boyd-Graber, J.：你能把它说清楚吗？学习改写上下文中的问题。载于：2019年自然语言处理实证方法会议与第九届国际自然语言处理联合会议 (EMNLP-IJCNLP) 论文集。第 5918–5924 页。计算语言学协会，香港，中国 (2019 年 11 月)。https://doi.org/10.18653/v1/D19-1605, https://aclanthology.org/D19 1605 11 - . Formal, T., Lassance, C., Piwowarski, B., Clinchant, S.：从知识蒸馏到硬负样本采样：提升稀疏神经信息检索模型的效果。载于：第 45 届国际 ACM 信息检索研究与开发会议论文集。第 2353–2359 页。SIGIR '22，计算机协会，美国纽约州纽约市 (2022 年 7 月)。https://doi.org/10.1145/3477495.3531857, http://doi.org/10 3531857 12 1145 3477495 . / . . Formal, T., Piwowarski, B., Clinchant, S.：SPLADE：用于第一阶段排序的稀疏词汇扩展模型。载于：第 44 届国际

际 ACM SIGIR信息检索研究与发展会议。第2288–2292页。SIGIR '21，计算机协会，纽约，纽约州，美国（2021年7月）。https://doi.org/10/gm2tf2, https://doi.org/10 3463098 13 1145 3404835 . / . . Hofstätter, S., Althammer, S., Schröder, M., Sertkan, M., Hanbury, A.：通过跨架构知识蒸馏改进高效神经排序模型。arXiv **abs/2010.02666**（2020年）14. Hofstätter, S., Lin, S.C., Yang, J.H., Lin, J., Hanbury, A.：通过平衡主题感知采样，高效地训练高效稠密检索器。载于：第44届ACM SIGIR信息检索研究与发展会议论文集。第113–122页。SIGIR '21，计算机协会，纽约，纽约州，美国（2021年）。https://doi.org/10.1145/3404835.3462891, https://doi.org/10 3462891 15 1145 3404835 . / . . Khattab, O., Zaharia, M.：ColBERT：通过BERT上的上下文化晚期交互实现高效且有效的段落检索 http://arxiv.org/abs/2004 12832 16 . . Krasakis, A.M., Yates, A., Kanoulas, E.：面向对话式搜索的零样本查询上下文化。载于：第45届ACM SIGIR信息检索研究与发展会议论文集。第1880–1884页。SIGIR '22，计算机协会，纽约，纽约州，美国（2022年7月）。https://doi.org/10.1145/3477495.3531769, https://doi.org/10 3531769 17 1145 3477495 . / . . Kumar, V., Callan, J.：让信息寻求更轻松：改进的对话式搜索流程，第10页 18. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., S oricut, R.：ALBERT：一种用于语言表征自监督学习的精简BERT。载于：第8届国际学习表征会议（ICLR 2020），埃塞俄比亚亚的斯亚贝巴，2020年4月26日至30日。OpenReview.net（2020年），https://openreview.net/forum?id=H1eA7AEtvS 19. Lin, S.C., Yang, J.H., Lin, J.：用于对话式搜索的上下文化查询嵌入 http://arxiv.org/abs/2104 08707 20 . . Lin, S.C., Yang, J.H., Lin, J.：面向稠密检索的紧密耦合教师模型知识蒸馏中的批内负例。载于：第6届自然语言处理表征学习研讨会（RepL4NLP-2021）论文集。第163–173页。计算语言学协会。https://doi.org/10.18653/v1/2021.repl4nlp-1.17, https://aclanthology.org/2021 17 21 4 1 2020 .repl nlp- . . Lin, S.C., Yang, J.H., Lin, J.：TREC笔记：CAsT赛道。技术报告，TREC（2021年12月）22. Lin, S.C., Yang, J.H., Nogueira, R., Tsai, M.F., Wang, C.J., Lin, J.：多阶段对话式段落检索：融合词项重要性估计与神经查询改写的方法 http://arxiv.org/abs/2005 02230 23 . . Lin, S., Yang, J., Nogueira, R., Tsai, M., Wang, C., Lin, J.：在对话式搜索中利用查询历史进行段落检索的查询改写。CoRR**abs/2005**02230 2**402230** 2020 2005.（），https://arxiv.org/abs/ . . Mele, I., Muntean, C.I., Nardini, F.M., Perego, R., Tonellotto, N.：通过搜索对话中的话语依赖关系发现上下文。技术报告（2021年）25. Nogueira, R., Jiang, Z., Pradeep, R., Lin, J.：使用预训练序列到序列模型进行文档排序。载于：计算语言学协会论文集：EMNLP 2020 Findings。第708–718页。计算语言学协会。https://doi.org/10.18653/v1/2020.findings-emnlp.63, https://www.aclweb.org/anthology/2020.findings-emnlp.63

26. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.：开放检索式对话问答，第539–548页。https://doi.org/10.1145/3397271.3401110, http://arxiv.org/abs/2005.11364 27. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.：开放检索式对话问答。载于：第43届ACM SIGIR信息检索研究与发展会议论文集。第539–548页。SIGIR '20，计算机协会，纽约，纽约州，美国（2020年）。https://doi.org/10.1145/3397271.3401110, https://doi.org/10.1145/3397271.3401110 28. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.：开放检索式对话问答。载于：Huang, J.X., Chang, Y., Cheng, X., Kamps, J., Murdock, V., Wen, J., Liu, Y.（编）《第43届ACM SIGIR信息检索研究与发展会议论文集》，SIGIR 2020，2020年7月25日至30日，中国虚拟会议。第539–548页。ACM（2020年）。https://doi.org/10.1145/3397271.3401110, https://doi.org/10.1145/3397271.3401110 29. Qu, C., Yang, L., Qiu, M., Croft, W.B., Zhang, Y., Iyyer, M.：用于对话式问答的BERT历史答案嵌入。载于：第42届ACM SIGIR信息检索研究与发展会议论文集。第1133–1136页。SIGIR'19，计算机协会，纽约，纽约州，美国（2019年）。https://doi.org/10.1145/3331184.3331341, https://doi.org/10.1145/3331184.3331341 30. Qu, C., Yang, L., Qiu, M., Zhang, Y., Chen, C., Croft, W.B., Iyyer, M.：用于对话式问答的注意力历史选择。载于：第28届ACM信息与知识管理国际会议论文集。第1391–1400页（2019年）31. Reddy, S., Chen, D., Manning, C.D.：CoQA：对话式问答挑战赛。《计算语言学协会汇刊》**7**，249–266（2019年）。https://doi.org/10.1162/tacl_a_00266, https://aclanthology.org/Q19-1016 32. Reddy, S., Chen, D., Manning, C.D.：CoQA：对话式问答挑战赛。《计算语言学协会汇刊》**7**，249–266（2019年）。https://doi.org/10.1162/tacl_a_00266, https://doi.org/10.1162/tacl_a_0026633. Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.：ColBERTv2：通过轻量级晚期交互实现有效且高效的检索。载于：2022年计算语言学协会北美分会会议：人类语言技术论文集。第3715–3734页。计算语言学协会，美国西雅图（2022年7月）。https://doi.org/10.18653/v1/2022.naacl-main.272, https://aclanthology.org/2022.naacl-main.272 34. Vakulenko, S., Longpre, S., Tu, Z., Anantha, R.：面向对话式问答的查询改写。载于：第14届ACM网络搜索与数据挖掘国际会议论文集。第355–363页。ACM。https://doi.org/10.1145/3437963.3441748, https://dl.acm.org/doi/10.1145/3437963.3441748 35. Voskarides, N., Li, D., Panteli, A., Ren, P.：ILPS在TREC 2019对话助手赛道上的工作，第4页

36. Voskarides, N., Li, D., Ren, P., Kanoulas, E., de Rijke, M.：有限监督条件下对话式搜索的查询解析，第 921–930 页。https://doi.org/10.1145/3397271.3401130, http://arxiv.org/abs/2005 11723 37 . . Yan, X., Clarke, C.L.A., Arabzadeh, N.：Waterlooclarke 在 TREC 2021 对话式助手赛道中的表现（2021）38. Yang, J.H., Lin, S.C., Wang, C.J., Lin, J.J., Tsai, M.F.：基于对话历史的查询和答案扩展。载于：TREC（2019）39. Yu, S., Liu, J., Yang, J., Xiong, C., Bennett, P., Gao, J., Liu, Z.：少样本生成式对话查询改写 http://arxiv.org/abs/2006 05009 40 . . Zamani, H., Trippas, J.R., Dalton, J., Radlinski, F.：对话式信息寻求（2022年1月）。https://doi.org/10.48550/arXiv.2201.08808，http://arxiv.org/abs/2201.08808, arXiv:2201.08808 [cs]


50
45
40
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
Data A
Data B