□大模型外挂知识库优化——负样本样本挖掘篇

来自： AiGC面试宝典

Just do it! 2024年04月29日 22:44

一、为什么需要构建负难样本？

在各类检索任务中，为训练好一个高质量的检索模型，往往需要从大量的候选样本集合中采样高质量的负例，配合正例一起进行训练。

二、负难样本构建方法篇

2.1 随机采样策略（Random Sampling）方法

- 方法：直接基于一均匀分布从所有的候选 Document 中随机抽取 Document 作为负例；

- 存在问题：由于**无法保证采样得到的负例的质量**，故经常会**采样得到过于简单的负例，其不仅无法给模型带来有用信息，还可能导致模型过拟合，进而无法区分某些较难的负例样本**
- 分析随机采样策略（Random Sampling）方法挖掘负例训练时对梯度的影响：

对于随机采样方法，由于其采样得到的负例往往过于简单，其会导致该分数接近于零

$$s_n(q,d) \longrightarrow 0$$

进而导致其生成的梯度均值也接近于零，

$$\nabla_\theta l(q,d) \longrightarrow 0,$$

这样过于小的梯度均值会导致模型不易于收敛。

2.2 Top-K 负例采样策略（Top-K Hard Negative Sampling）方法

- 方法：基于一稠密检索模型对所有候选 Document 与 Query 计算匹配分数，然后**直接选择其中 Top-K 的候选 Document 作为负例**；
- 优点：可以保证采样得到的负例是模型未能较好区分的较难负例；
- 存在问题：**很可能将潜在的正例也误判为负例，即假负例（False Negative）**。如果训练模型去将该部分假负例与正例区分开来，反而会导致模型无法准确衡量 Query-Document 的语义相似度。
- 分析 Top-K 负例采样策略（Top-K Hard Negative Sampling）方法 挖掘负例训练时对梯度的影响：

由于其很容易采样得到语义与正例一致的假负例，其会导致正负样本的右项 $\nabla_\theta s_n(q,d)$ 值相似，但是左项符号相反，这样会导致计算得到的梯度方差很大，同样导致模型训练不稳定。

2.3 困惑负样本采样方法 SimANS 方法

- 动机：在所有负例候选中，与 Query 的语义相似度接近于正例的负例可以同时具有较大的梯度均值和较小的梯度方差，是更加高质量的困惑负样本
- 方法：对与正例语义相似度接近的困惑负例样本进行采样
- 采样方法特点：
    - 与 Query 无关的 Document 应被赋予较低的相关分数，因其可提供的信息量不足；
    - 与 Query 很可能相关的 Document 应被赋予较低的相关分数，因其可能是假负例；
    - 与正例语义相似度接近的 Document 应该被赋予较高的相关分数，因其既需要被学习，同时是假负例的概率相对较低。

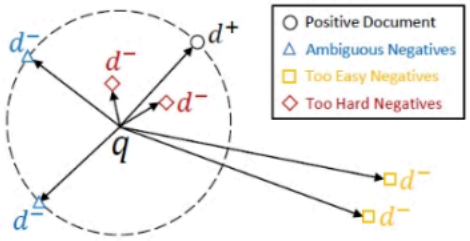

Figure 4: An example of the dense embedding distribution of a query with its positive document, too easy, too hard and ambiguous negatives.

- 困惑样本采样分布

通过以上分析可得，在该采样分布中，随着 Query 与候选 Document 相关分数 s(q,di) 和与正例的相关分数 s(q, d+) 的差值的缩小，该候选 Document 被采样作为负例的概率应该逐渐增大，故可将该差值作为输入，配合任意一单调递减函数 f(·) 即可实现（如 e^{-x}）。故可设计采样分布如下所示：

$$p_i \propto \exp\left(-\alpha(s(q,d_i) - s(q,d^+) - b)^2\right), \forall d_i \in \mathcal{D}^-$$

其中为控制该分布密度的超参数，b 为控制该分布极值点的超参数，$d^+ \in \mathcal{D}^+$是一随机采样的正例样本，D- 是 Top-K 的负例。通过调节 K 的大小，我们可以控制该采样分布的计算开销。以下为该采样方法具体实现的伪代码：

Algorithm 1: The algorithm of SimANS.

Input: Queries and their positive documents $\{(q, \mathcal{D}^+)\}$, document pool $\mathcal{D}$, pre-learned dense retrieval model M

Build the ANN index on $\mathcal{D}$ using M.
Retrieve the top-k ranked negatives $\widetilde{\mathcal{D}}^-$ for each query with their relevance scores $\{s(q, d_i)\}$ from $\mathcal{D}$.
Compute the relevance scores of each query and its positive documents $\{s(q, \mathcal{D}^+)\}$.
Generate the sampling probabilities of retrieved top-k negatives $\{p_i\}$ for each query using Eq. 3.
Construct new training data $\{(q, \mathcal{D}^+, \widetilde{\mathcal{D}}^-)\}$.
while M *has not converged* **do**
    Sample a batch from $\{(q, \mathcal{D}^+, \widetilde{\mathcal{D}}^-)\}$.
    Sample ambiguous negatives for each instance from the batch according to $\{p_i\}$.
    Optimize parameters of M using the batch and sampled negatives.
end

2.4 利用 对比学习微调 方式构建负例方法

对比学习是优化向量化模型的常用训练方法：

- 目的：**优化向量化模型，使其向量化后的文本，相似的在向量空间距离近，不相似的在向量空间距离远**。文档召回场景下，做对比学习（有监督）需要三元组（问题，文档正例，文档负例）。文档正例是和问题密切相关的文档片段，文档负例是和问题不相关的文档片段，可以是精挑细选的，也可以是随机出来的。
- 构建方法：如果是随机出来的话，**完全可以用同一个 batch 里，其他问题的文档正例当作某一个问题的文档负例，如果想要效果好，还需要有比较大的 batch size**。

损失函数是基于批内负样本的交叉熵损失，如下公式所示,q、d 分别表示问题和文档正例对应的向量，r 为温度系数,sim 函数可以是 cos 相似度或者点积。

论文：SimCSE: Simple Contrastive Learning of Sentence Embeddings

$$\ell_i = -\log \frac{e^{\mathrm{sim}(q_i, d_i^+)/r}}{\sum_{j=1}^{N} e^{\mathrm{sim}(q_i, d_j^+)/r}}$$

- 实现方法：

分别将 B1 个问题，和 B2 个文档片段通过向量化模型变成向量形式，然后通过矩阵乘积计算每个问题和文档的相似度，最后通过交叉熵损失进行优化。如果文档负例仅来自于同一个 batch 的其他样本的文档正例，那么 B1=B2；如果人工的给每个样本陪 k 个文档负例（比如可以通过难例挖掘得到），那么 B2 =(k+1)*B1。

```
q_reps = self.encode(query) # 问题矩阵 维度(B1, d)
d_reps = self.encode(doc) # 文档矩阵 维度(B2, d)
score = torch.matmul(q_reps, d_reps.transpose(0, 1)) #计算相似度矩阵 维度:(B1, B2)
scores = scores / self.temperature
target = torch.arange(scores.size(0), device=scores.device, dtype=torch.long) ## 得交叉熵损失函数的标签
# 考虑文档负例不仅来自于batch内其他样本的文档正例，也可能人工的给每个样本构造一些文档负例。
target = target * (p_reps.size(0) // d_reps.size(0))
```

```
loss = cross_entropy(scores, target) //交叉熵损失函数
```

注：bge2论文里，做基于批内负样本的对比学习时同时考虑了多任务问题。之前也介绍了，不同任务加的prompt是不同的，如果把不同任务的样本放到一个batch里，模型训练时候就容易出现偷懒的情况，有时候会根据pormpt的内容来区分正负例，降低任务难度，这是不利于对比学习效果的。因此，可以通过人为的规定，同一个batch里，只能出现同一种任务的样本缓解这个问题。（实际应用场景下，如果任务类别不是非常多的话，最好还是一个任务训练一个模型，毕竟向量化模型也不大，效果会好一些）

2.5 基于批内负采样的对比学习方法

- 本质：随机选取文档负例，**如果能有针对性的，找到和文档正例比较像的文档负例（模型更难区分这些文档负例），加到训练里，是有助于提高对比学习效果的**。就好比我们只有不断的做难题才能更好的提高考试水平。
- 论文方法：在文档向量空间找到和文档正例最相近的文档片段当作文档负例，训练向量化模型。模型更新一段时间后，刷新文档向量，寻找新的文档负例，继续训练模型。

参考论文：

【1】Approximate nearest neighbor negative contrastive learning for dense text retrieval

【2】Contrastive learning with hard negative samples

【3】Hard negative mixing for contrastive learning

【4】Optimizing dense retrieval model training with hard negatives

【5】SimANS: Simple Ambiguous Negatives Sampling for Dense Text Retrieval

2.6 相同文章采样方法

- 思路：文档正例所在的文章里，其他文档片段当作难负例，毕竟至少是属于同一主题的，和随机样本比起来比较难区分。
- 存在问题：实际应用场景下，如果你的数据比较脏，难例挖掘用处可能不大。

2.7 LLM辅助生成软标签及蒸馏

- 方法：根据用户问题召回的相关文档片段最终是要为 LLM 回答问题服务的，因此 LLM 认为召回的文档是否比较好很重要，以下介绍的方法是 bge2 提出的。对于向量化模型的训练，可以让 LLM 帮忙生成样本的辅助标签，引导向量化模型训练。辅助标签的生成可用如下公式表示。在已知 LLM 需要输出的标准答案下，分别将问题和各个文档片段 C 放入 LLM 的 prompt 中，看 LLM 生成标准答案的概率 r 大小，当作辅助标签。r 越大，表示起对应的文档片段对生成正确答案的贡献越大，也就越重要。
- 存在问题：
    - 打标要求有点太高；
    - 很多实际应用场景，我们并没法拿到 LLM 回答的标准答案，同时对每个问题的候选文档片段都计算一个 r，开销貌似有点大。

$$r_{C|O} = \prod|_{i=1}^{|O|} \text{LLM}(o_i|C, O_{:i-1})$$

利用以上 LLM 生成的标签以及 KL 散度（笔者认为论文里这个形式的公式不能叫做 KL 散度吧。。），对模型进行优化。P 为某个问题 q 对应的候选文档片段 p 的集合，e 表示向量，<.,.>表示相似度操作，w 是对所有候选文档 p 对应的辅助标签值 r 经过 softmax 变换后的值。本质是，如果 LLM 认为某个文档片段越重要，给它的优化权重越大。为了进一步稳定蒸馏效果，还可以对候选文档片段根据 r 进行排序，只用排名靠后的样本进行优化。

$$\min. \sum_{p} -w_i * \log \frac{\exp(\langle e_q, e_p \rangle / \tau)}{\sum_{p' \in P} \exp(\langle e_q, e_{p'} \rangle / \tau)}.$$

辅助知识

附一：梯度计算方法

以稠密检索常用的 BCE loss 为例，正例与采样的负例在计算完语义相似度分数后，均会被 softmax 归一化，之后计算得到的梯度如下所示：

$$\nabla_\theta l(q,d) = \begin{cases} (s_n(q,d) - 1) \nabla_\theta s_n(q,d) & if\ d \in \mathcal{D}^+ \\ s_n(q,d) \nabla_\theta s_n(q,d) & if\ d \in \mathcal{D}^- \end{cases}$$

注：

$^1 s_n(q,d)$：经过 softmax 归一化后的语义相似度分数