# From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing

Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps

连晓颖 2019-10-21

# Contents

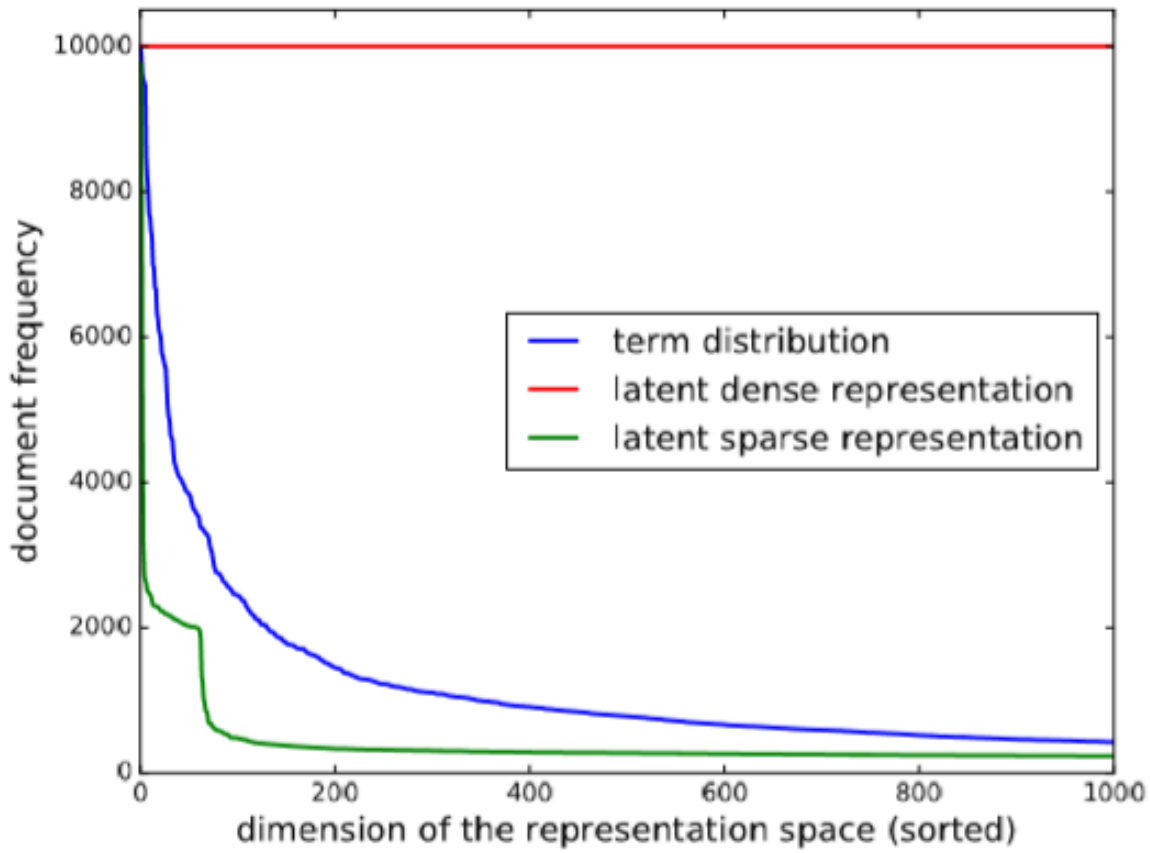
# 1 Introduction

- **Motivation**
- **Main Idea**

# 1.1 Motivation

- 现有的检索模型（Learning to Rank & Nerual Model）
  - 大都是多阶段排序模型，在前一个阶段过滤出的文档基础上进行重排序
  - 前一个阶段成了这些模型的瓶颈，误差在阶段间具有传递性
    - 最相关的文档没能在第一阶段被发现，则之后的阶段都不能弥补
  - 重排序阶段生成的特征都是稠密特征
    - 非0位置少，不适合直接建立倒排索引，同一个文档会几乎会出现在每一个索引上
- 本文提出的模型
  - 单阶段：避免误差传递
  - 高维稀疏：便于构建倒排索引，检索效率几乎和传统词项模型一致
  - 可计算相关性：Query(Q)和Document(D)之间向量内积

# 1.2 Main Idea




- 词项表示（蓝线）
  - 近似符合ZipFian分布
- 稠密表示(红线)
  - 随着表示维度的增加，倒排表的大小还是整个文档集的大小
- 稀疏表示（绿线）
  - 甚至比词项表示的倒排表还要更小些

# 2 Standalone Neural Ranking Model (SNRM)

- **Objectives**
- **Loss Function**
- **Network Architecture**
  - **Training Time**
  - **Inference Time**
  - **Sub-network**
- **Training Data: Weak Supervision**

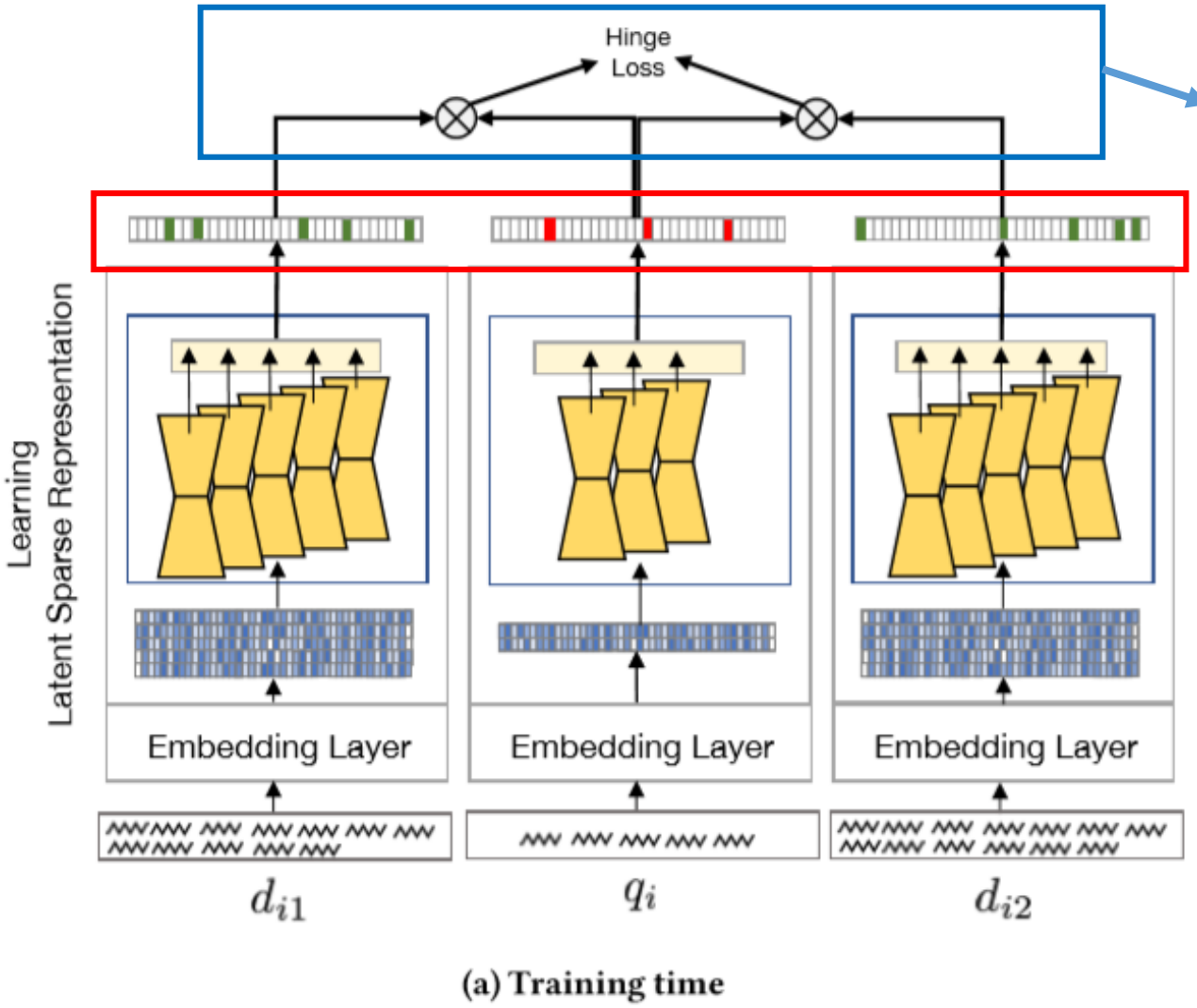
# 2.1 Objectives

- 相关性
  - Pairwise Hinge Loss $\mathcal{L} = \max\{0, \epsilon - y_i[\psi(\phi_Q(q_i), \phi_D(d_{i1})) - \psi(\phi_Q(q_i), \phi_D(d_{i2}))]\}$
- 稀疏性
  - 稀疏比越大，稀疏性越高 $\text{sparsity ratio}(\vec{v}) = \frac{\text{total number of zero elements in } \vec{v}}{|\vec{v}|}$
  - 最大化稀疏比等价于最小化L0（令 $0^0 = 0$） $L_0(\vec{v}) = \sum_{i=1}^{|\vec{v}|} |\vec{v}_i|^0$
  - 由于L0不可导，无法采用端到端的训练方法，因而用L1替代
  - 由于使用了Relu激活函数( Relu(x) = max(0, x) )，会使得很多非正值变为0

# 2.2 Loss Function

- 对于第i个训练样本的Loss为 $\mathcal{L}(q_i,d_{i1},d_{i2},y_i)+\lambda L_1(\phi_Q(q_i)||\phi_D(d_{i1})||\phi_D(d_{i2}))$
  - 第一项：pairwise hinge Loss
  - 第二项：L1, “||” 代表向量连接Q和D1，D2的向量表示
- $\lambda$控制向量的稀疏性，越大则越稀疏
  - 但不能过大，过大会导致出现很多0元素，使检索模型的性能下降
  - 换言之，只要有足够的0元素保证模型的性能稳定就行
- Q，D1和D2的表示维度都比较高，在本文的实验中达到了20000维

# 2.3 Network Architecture (Training Time)

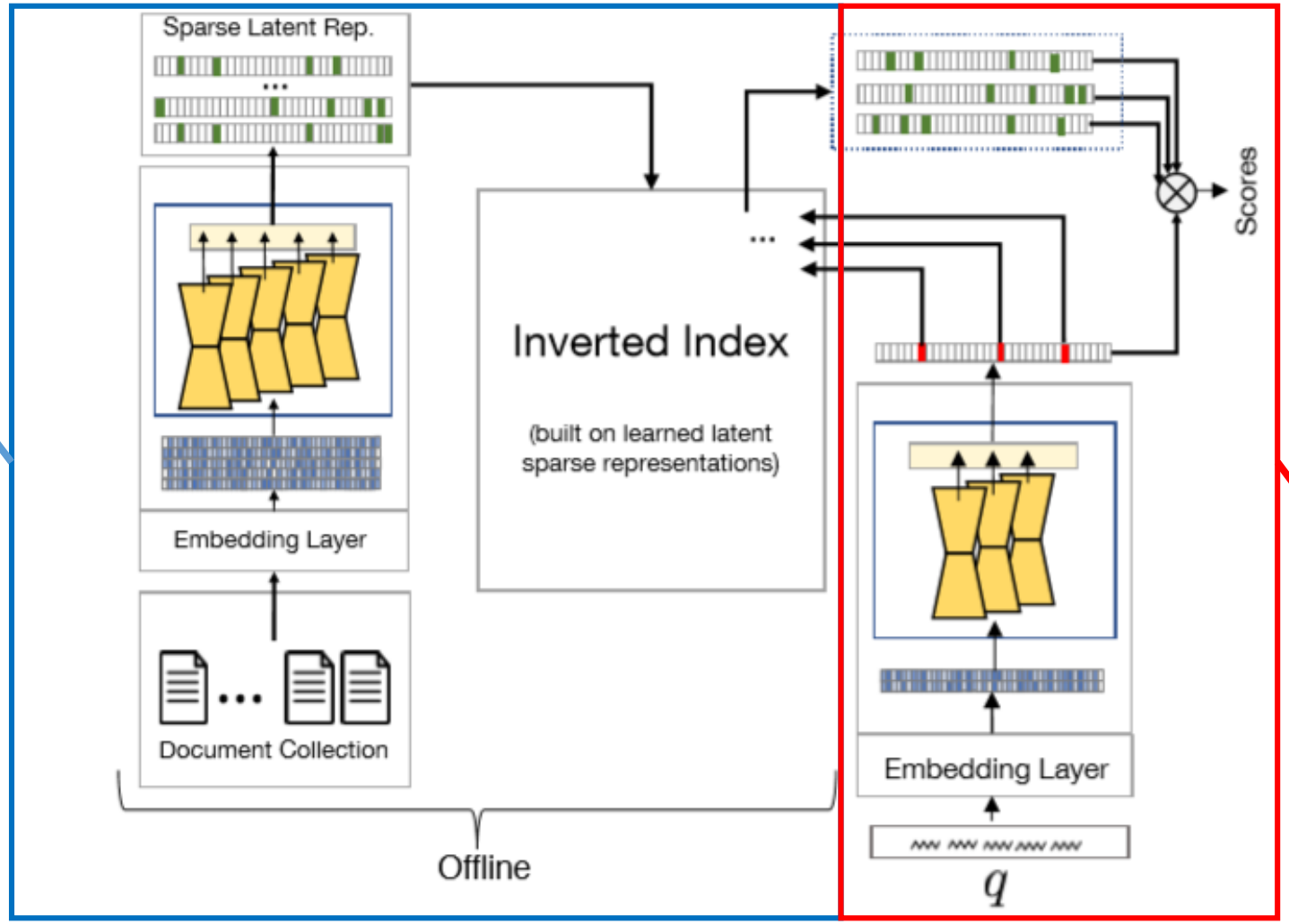



(a) Training time

- Hinge Loss： 区分相似和不相似
- 高维稀疏向量表示

# 2.3 Network Architecture (Inference Time)

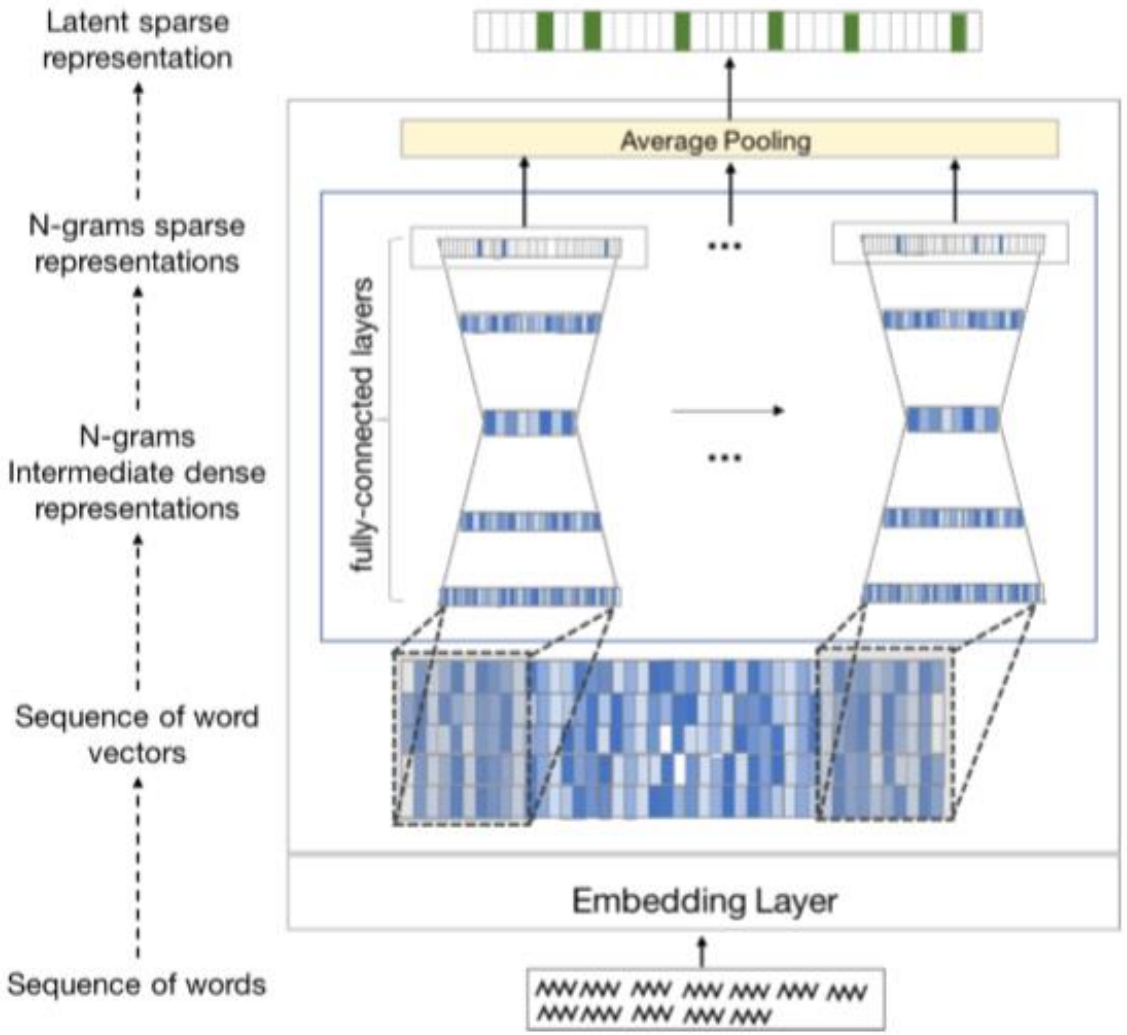
- 离线计算




(b) Inference time

- 线上查询

# 2.3 Network Architecture (Sub-network)




- Q应该比D含有更少的非0项，更少的非零项使得查询时合并的次数更少
  - 与输入长度挂钩, N-gram, 均值池化聚合
  - |q|-n+1 << |d|-n+1

$$\phi_D(d) = \frac{1}{|d|-n+1} \sum_{i=1}^{|d|-n+1} \phi_{\text{ngram}}(w_i, w_{i+1}, \cdots, w_{i+n-1})$$

- 训练参数应足够被放入GPU中
  - FC层维度先减小再增加
- Q和D的高维稀疏向量应该在同一个语义空间里
  - 参数共享$\phi_{n-gram}$

# 2.4 Training Data: Weak Supervision

- 弱监督：用程序生成弱标签数据
  - 收集大量的Query
  - 用现有的检索模型检索每一个Query（eg: Query likelihood）
  - 从查询列表里采样一个相关文档，从整个数据集里负采样一个不相关的文档，得到一个训练样本$(q_i,\ d_{i1},\ d_{i2},\ y_i,)$
  - 标签是Query likelihood概率值之差的符号

$$y_i = \text{sign}\left(p_{QL}(q_i|d_{i1}) - p_{QL}(q_i|d_{i2})\right)$$

# 3 Experiments

- **Dataset**
- **Sparsity and Efficiency**
- **Effectiveness**
- **Robustness to Collection Growth**

# 3.1 Dataset

- 两个数据集：Robust(250个查询)、ClueWeb（200个查询）
- 每个数据集上采用二折交叉验证优化参数
- 弱监督数据集：AOL query logs （600万不同的查询）
- 评价标准：MAP@1000, P@20, nDCG@20，Recall@1000

| ID | collection | queries (title only) | #docs | avg doc length | #qrels |
|---|---|---|---|---|---|
| Robust | TREC Disks 4 & 5 minus CR | TREC 2004 Robust Track, topics 301-450 & 601-700 | 528k | 254 | 17,412 |
| ClueWeb | ClueWeb 09 - Category B | TREC 2009-2012 Web Track, topics 1-200 | 50m | 1,506 | 18,771 |

# 3.1 Sparsity and Efficiency

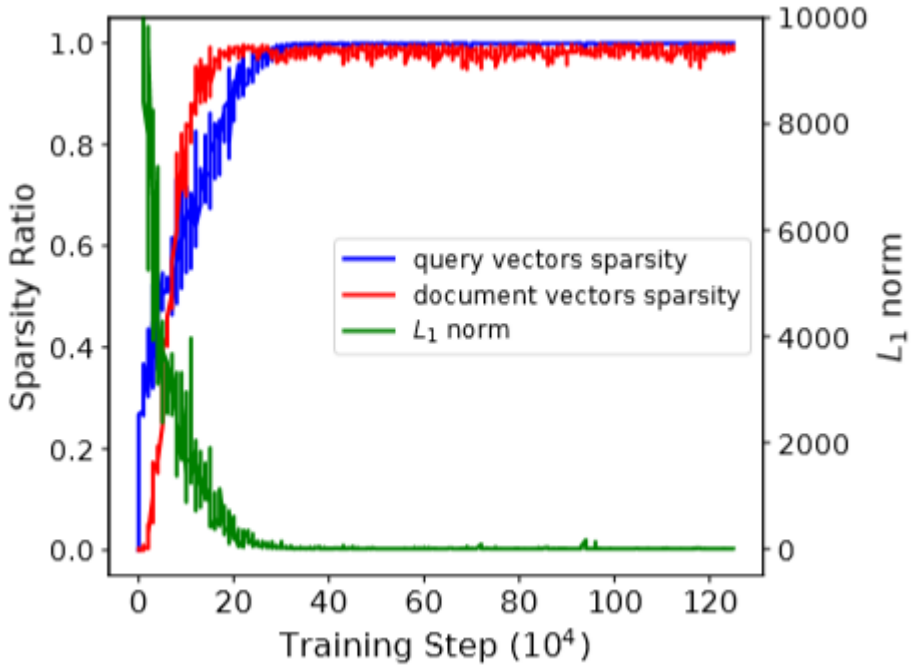
- 输出10000维时非0位的个数，的确生成了高维稀疏向量，Q要比D更稀疏

| # Unique latent terms... | Robust | | ClueWeb | |
|---|---|---|---|---|
| | Mean | Std. dev. | Mean | Std. dev. |
| per document | 97.96 | 447.57 | 130.24 | 561.53 |
| per query | 3.37 | 3.04 | 3.87 | 4.51 |

- 每个查询的平均运行时间(ms)，包括生成Q向量和检索评分的时间，和词项匹配模型的速度差不多

| Method | Robust | | ClueWeb | |
|---|---|---|---|---|
| | Mean | Std. dev. | Mean | Std. dev. |
| QL | 35.14 | 18.43 | 662.86 | 746.68 |
| SNRM | 46.12 | 23.11 | 612.73 | 640.98 |

- L1 norm下降，D和Q的稀疏比均上升



# 3.2 Effectiveness

| Method | Robust | | | | ClueWeb | | | |
|---|---|---|---|---|---|---|---|---|
| | MAP | P@20 | nDCG@20 | Recall | MAP | P@20 | nDCG@20 | Recall |
| QL | 0.2499 | 0.3556 | 0.4143 | 0.6820 | 0.1044 | 0.3139 | 0.2294 | 0.3286 |
| SDM | 0.2524 | $0.3679^{1}$ | $0.4242^{1}$ | 0.6858 | 0.1078 | 0.3141 | 0.2320 | $0.3385^{1}$ |
| RM3 | $0.2865^{12}$ | $0.3773^{12}$ | $0.4295^{12}$ | $0.7494^{12}$ | 0.1068 | 0.3157 | 0.2309 | 0.3298 |
| FNRM | $0.2815^{12}$ | $0.3752^{12}$ | $0.4327^{12}$ | $0.7234^{12}$ | $0.1329^{123}$ | $0.3351^{123}$ | $0.2392^{13}$ | $0.3426^{123}$ |
| CNRM | $0.2801^{12}$ | $0.3764^{12}$ | $0.4341^{123}$ | $0.7183^{12}$ | $0.1286^{123}$ | $0.3317^{123}$ | $0.2337^{1}$ | $0.3345^{13}$ |
| SNRM | $0.2856^{12}$ | $0.3766^{12}$ | $0.4310^{12}$ | $0.7481^{1245}$ | $0.1290^{123}$ | $0.3336^{123}$ | $0.2351^{13}$ | $0.3393^{135}$ |
| SNRM with PRF | $\mathbf{0.2971}^{123456}$ | $\mathbf{0.3948}^{123456}$ | $\mathbf{0.4391}^{123456}$ | $\mathbf{0.7716}^{123456}$ | $\mathbf{0.1475}^{123456}$ | $\mathbf{0.3461}^{123456}$ | $\mathbf{0.2482}^{123456}$ | $\mathbf{0.3618}^{123456}$ |

- SNRM取 Top 1000 的结果要比FNRM [Dehghani et al., SIGIR '17] 和CNRM ( 卷积FNRM ) 取 Top 2000 还要好（Recall）
- SNRM + PRF（伪相关度反馈扩展查询）比所有的模型都要更好

# 3.3 Robustness to Collection Growth

- 在训练时随机从Robust数据集里去掉文档

| % removal | MAP | P@20 | nDCG@20 | Recall |
|---|---|---|---|---|
| no removal | 0.2971 | 0.3948 | 0.4391 | 0.7716 |
| 1% removal | 0.2953 | 0.3953 | 0.4401 | 0.7691 |
| 5% removal | $0.2776^{\nabla}$ | $0.3807^{\nabla}$ | $0.4227^{\nabla}$ | $0.7349^{\nabla}$ |

- 去掉1%的文档（超过5k个）：模型表现基本不受影响
- 去掉5%的文档（超过26k个）：模型表现严重下降（可能是缺失了很多词汇导致）
- 说明在数据更新的情况下，模型需要定期增量训练

# 4 Conclusion

- SNRM是单阶段检索模型，有可能通过训练进一步提升Recall
- SNRM可以端到端训练
- SNRM可以通过PRF(伪相关度反馈)扩展查询，提升性能

Thank you !