Document Aware Contrastive Learning Approach for Generative Retrieval

Kavin R V
kavin19@iiserb.ac.in
Indian Institute of Science Education and Research, Bhopal
India

Maunendra Sankar Desarkar
maunendra@cse.iith.ac.in
Indian Institute of Technology Hyderabad
India

ABSTRACT

The field of neural retrieval models has seen significant advancements with the rise of dense retrieval techniques, where documents are indexed based on embeddings from neural networks, particularly transformer-based models. Generative retrieval, a newer paradigm, employs sequence-to-sequence models to generate unique identifier strings for passages, providing an alternative approach to retrieval. However, these generative models primarily focus on learning the mappings between the queries and the identifiers, which may limit their ability to understand the relationship between queries and passages fully. This work introduces a novel method inspired by dense retrieval to enhance the learning of Document-aware representations, thereby fostering a deeper understanding of the relationship between queries and documents. Additionally, a curriculum-based learning strategy is proposed to optimize contrastive losses effectively. Extensive experiments were conducted using the publicly available Natural Questions dataset to evaluate the proposed approach. The results demonstrate modest improvements in performance across all metrics, highlighting the method's robustness.

1 INTRODUCTION

Information retrieval (IR) is a pivotal field focused on obtaining relevant information from extensive collections of data. By employing various methods and technologies, IR systems aim to locate and present information that meets users' needs from sources such as databases, web pages, text corpora, etc. Traditionally, IR methods have been employed in various applications, including web search engines, recommendation systems, open-domain question answering, and knowledge-grounded conversation [7]. In recent times, the importance of information retrieval has grown, as it forms the backbone of several emerging technologies such as retrieval-augmented generation [8, 13], knowledge-intensive tasks [17], and long-range transformers [3].

Sparse retrieval methods, such as those based on keyword matching and term frequency-inverse document frequency (TF-IDF), have long been foundational in information retrieval, offering quick search times and straightforward implementation. However, these methods struggle with capturing semantic relationships and context, treating terms independently and resulting in sub-optimal retrieval for complex queries. Dense retrieval has gained popularity due to transformer-based models that map queries and documents into continuous, high-dimensional vector spaces using a transformer-based encoder (BERT, RoBERTa), enabling better representation of semantic meanings. This approach addresses some limitations of sparse retrieval, such as the lack of semantic similarity and vocabulary mismatches. Despite its advantages, dense retrieval presents challenges, including fine-grained interactions between passages and queries [24], embedding space bottleneck [11], and the need for large pre-computed indexes that demand substantial memory and resources [27]. Additionally, these models lack end-to-end processing capabilities.

To address the limitations previously discussed, a series of works [4, 5, 22] have proposed alternative approaches in generative retrieval. These end-to-end models utilize sequence-to-sequence language models, such as T5 [19] and BART [12], to retrieve documents by generating their unique document identifier (docID). This method supports fine-grained interaction between queries and docIDs, allowing for comprehensive end-to-end optimization [25]. However, most of these approaches overlook the interaction between the query and the context of its relevant document, where this interaction occurs only within the parameters and at the token prediction stage. This oversight means that the model does not adequately capture the relationship between the context of a document and the query. Consequently, the model may only thoroughly learn some nuanced interactions between them, potentially affecting retrieval performance.

In order to achieve an improved retrieval performance, we propose a Document Aware Generative Retriever (DAGR) model leveraging a novel document-aware learning technique that enhances query representations to better align with document representations. This is accomplished by incorporating a margin-based contrastive loss at the encoder's output stage. We also establish a connection between dense retrieval and the language model head by introducing a method known as document-aware Label Mapping (DALM). DALM leverages the principles of dense retrieval, applying contrastive loss to the document embedding and the language model head's weights associated with the document's respective tokens. Furthermore, we present a curriculum-based learning approach that incrementally increases the difficulty of negative samples used in contrastive loss, fostering a more robust learning process. Additionally, we explore the application of contrastive loss at the token level to further enhance model performance.

This study aims to address several research questions:

- **RQ1**: Investigate the effectiveness of document-aware curriculum learning and label mapping.
- **RQ2**: Evaluate the impact of curriculum learning on the performance of contrastive loss.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
Gen-IR@SIGIR2024, July 18, 2024, Washington DC, US
© 2024 Copyright held by the owner/author(s).

- **RQ3**: Compare the efficacy of margin loss and listwise loss.
- **RQ4**: Assess the performance of different models on documents not encountered during post-tuning.
- **RQ5**: Examine the advantages of DAGR over token-level contrastive learning.

2 RELATED WORK

2.1 Information Retrieval

Retrieval is a fundamental part of many Knowledge Intensive NLP tasks. The task involves retrieving relevant information from a large corpus of knowledge based on a given query. Formally, given a query q and a corpus C, the goal is to retrieve the top k documents from C that are most relevant to q, where relevance is typically measured using a similarity metric between the query and documents in the corpus.

2.1.1 Sparse Retrieval. Sparse Retrievers are a retrieval method that uses explicit lexical matches between the query and the documents in the corpus. In other words, they retrieve documents that contain specific keywords or phrases that match the query. One common approach is to use an inverted index, a data structure that stores all the terms in the corpus along with the list of documents in which they appear. Given a query, the inverted index can quickly retrieve the documents that contain the query terms. Some examples of sparse retrieval methods include TF-IDF and BM25.

2.1.2 Dense Retrieval. Dense retrieval is a method that leverages transformer-based models to encode queries and documents from a corpus into dense vectors. These methods retrieve documents that closely resemble the query vector, capturing semantic similarity between the query and documents. Unlike sparse retrieval, dense retrieval can find relevant documents even when they do not contain exact query term matches. Dense retrieval methods often involve pre-computed indexing, which converts documents to vectors and stores them for later access during searches. Notable examples include dual-encoder dense passage retrieval techniques [9, 20].

Dense retrieval uses dense vector representations of text for similarity search. In this approach, a neural network encodes the query and the document as dense vectors. The similarity between the two vectors is then computed using distance metrics such as cosine similarity, allowing dense retrievers to capture the semantics of the text for improved retrieval performance over sparse retrievers. This method often employs pre-trained language models like BERT and RoBERTa, which can capture rich and complex semantic relationships for enhanced retrieval tasks. While dense retrieval offers several advantages, such as semantic understanding and improved relevance, it also presents challenges. Computing dense vector representations can be computationally intensive and slow, mainly when dealing with a large number of documents.

2.1.3 Generative Retrieval. Generative retrieval is an end-to-end retrieval technique that generates text sequences as a response to a query rather than retrieving pre-existing text embedding. In this approach, initially, every document is mapped to a unique identifier that represents a passage or document like Passage title [5], substrings [4], URL [27] pseudo queries [14] and sometimes the whole passage [11]. More recent models have explored semantic docid, which is constructed using either hierarchical K-means [22, 24] or based on product quantisation [27].

These models typically rely on sequence-to-sequence transformer architectures [23] built on top of models like T5 [19], BART [12]. They are trained to predict the sequence of tokens in a docid given an input query. The objective of the retrieval can interpreted as $\mathcal{P}(d|q) = \prod_{i=0}^{|d|} \mathcal{P}(d_i|q, d_{<i})$ for a document with docid d and it's respective query q, and d_i is the ith token of the docid d. During inference, these models use beam search to retrieve or generate ranked lists of documents. To mitigate hallucinations and avoid generating unwanted tokens in the docID, constrained beam search is employed, guiding the language model to generate text that follows certain constraints using data structures such as prefix trees or FM indexes.

3 METHODOLOGY: DOCUMENT AWARE TRAINING FOR GR

In this work, we propose a two-stage training procedure for generative retrieval. Initially, we train a straightforward sequence-to-sequence transformer [23] based on the T5 architecture [19]. This model processes a query as input and produces the most relevant document by generating a sequence of tokens that form the document's identifier (docID). The training utilizes a substantial collection of (query, docID) pairs, allowing the model to map queries to document identifiers effectively. Following recent works [22, 24], we employ a hierarchical k-means algorithm to construct the docIDs. This clustering approach organizes documents hierarchically, facilitating the generation of meaningful docIDs that capture the semantic structure of the corpus. In the second stage, we fine-tune the trained model using a novel document-aware curriculum-based contrastive learning approach, which further enhances the model's performance. Inspired by the dense passage retrieval (DPR) technique [9], this method applies margin-based/listwise contrastive loss to minimize the distance between the encoder/decoder representation of the query and the corresponding document, aiming to achieve similar representations for both. Additionally, we leverage the document embedding to create document-aware label mappings, reducing the distance between the document embedding and the language model head's weights associated with the respective tokens. This approach enables the model to align document embeddings with the output space better, leading to improved retrieval performance.

3.1 Semantic Docid

In our approach, Similar to NCI [24], we create document identifiers based on the input query while integrating document semantic information into the retrieval process. This technique helps manage large corpora by embedding useful semantic content into the identifiers, allowing us to encode the documents' semantics into the learning process. The goal is to generate document identifiers that are semantically aligned so documents with similar content have closely related identifiers, thereby streamlining the retrieval process. We employ the hierarchical k-means algorithm to cluster and encode documents to accomplish this. This process begins by classifying the documents into k clusters using their BERT-encoded

[6] representations. The algorithm is applied recursively for clusters with more than c documents. Once each cluster is limited to c documents or fewer, the documents within each cluster are numbered sequentially from 0 to $c-1$. This method organizes the documents into a tree structure (T) with a root node (r_0). Each document is linked to a leaf node through a path ($l = r0, r1, ..., rm$) from the root, where each internal node represents a cluster at different levels of the tree, and the leaf node corresponds to a specific document. The final identifier for each document is constructed by concatenating the indices from root to leaf, forming a unique semantic identifier for each document. Encoding documents this way ensures that documents with similar semantics have similar identifiers, making the retrieval process more efficient and accurate. Following the implementation of NCI, in our experiments, we use k = 30 and c = 30 as our parameters for clustering.

3.2 Query Generation

Generating semantic document identifiers (docid) based solely on a single input query may lead to an inadequate representation of document semantics. Therefore, following the approaches used in NCI [24] and DSI [22], we integrate each document's content into the model's parameters during training to enhance semantic capture. To achieve this, we augment queries using two methods: "document as queries" and "generated queries" from DocT5Query [16]. For the "document as queries" approach, we use the first 64 terms from a document as queries and randomly sample 10 additional groups of 64 consecutive terms from the entire document. This provides a more comprehensive representation of the document's content and semantics. For the "generated queries" method, we employ a sequence-to-sequence transformer [23] based on the DocT5Query architecture. This model takes the document tokens as input and generates relevant queries through random sampling. Random sampling ensures diversity in the generated queries and captures different aspects of the document's content.

3.3 Base Generative Retriever

The base retriever employs a T5-based architecture [18] and adheres to a traditional learning objective for generative retrieval. This model processes an input query and learns to predict the sequence of tokens representing the relevant document's docid. Given a query q, the model's encoder produces a representation that is used by the decoder to generate the docid $r = r_1, r_2, ...r_{|r|}$. The docids are derived from a hierarchical k-means tree structure, where the i-th token r_i corresponds to the node at the i-th level from the root. It represents the r_i-th child of node r_{i-1}, such as in the docid 23-5-12, where 5 indicates the 5th child of node 23. A crucial aspect to note is that the ordering of tokens within the docid is significant. Thus, docids 23-5-12 and 5-12-23 represent different paths and have distinct meanings. To address this, following NCI [24], each token is uniquely represented at each position within the docid. For instance, in the docids 23-5-12 and 5-12-23, the tokens (23, 1), (5, 2), (12, 3) and (5, 1), (12, 2), (23, 3) respectively, convey different paths. In the training process, each new docid $d = d_1, d_2, ...d_{|d|}$ is calculated as $d_i = r_i + i \cdot c$, where c represents the maximum number of children per node (e.g., $c = 30$). This way, the docid 5-12-23 is transformed into 5-42-83, while 23-5-12 becomes 23-35-72, creating distinct representations for each document. This model is trained using the following loss objective:

$$\mathcal{L}_{tok} = \sum_{k=1}^{|d|} \log p_\theta(d_k | q, d_{<k}) \tag{1}$$

where θ represents the model parameters and $d_{<k}$ denotes the sequence of tokens preceding d_k.

The probability function is defined as:

$$p_\theta = \text{Softmax}(\text{Decoder}(\text{Encoder}(q), d_{<k}) \cdot W^{lm}) \tag{2}$$

where $W^{lm} \in \mathbb{R}^{h \times v}$ denotes the weights of the language modeling head, with h being the model's hidden size and v the decoder's vocabulary size. The query q encompasses all ground truth queries, generated queries, and documents treated as queries. The training process optimizes the model parameters θ jointly across these various types of queries.

3.4 Document Aware Contrastive Learning

Based on the model produced by the above process, we predict a few of the most relevant documents for each query. We accumulate all retrieved documents along with the relevant documents to build a dictionary $\{d : s\}$, where d is the docid and s is the set of most relevant documents according to the model. This set s is a source of soft negatives when sampled from it.

The document-aware loss utilizes the decoder output of the query and the document context dc, whose dot product is calculated at the respective positions and used to build a listwise loss. This dc comprises 64 consecutive terms from the document, ensuring maximal overlap with the query q. The decoder query representation at ith position of the docid is calculated as follows:

$$\mathcal{D}_q^i = \text{Decoder}(\text{Encoder}(q), d_{<i}), \tag{3}$$

Where $\mathcal{D}_q^i$ is the query's decoder representation at the ith step.

The similarity score of query decoder embedding $\mathcal{D}_q^i$ and the decoder embedding of the positive document context ($\mathcal{D}_{dc_{pos}}^i$) is calculated and summed for all the position as shown below:

$$\mathcal{S}_{pos}^d = \sum_{i=0}^{|d|} \mathcal{D}_q^i \cdot (\mathcal{D}_{dc_{pos}}^i)^T \tag{4}$$

The scores for soft Negatives ($\mathcal{S}_{neg}^d$) from the most relevant documents according to the base model and the random negatives ($\mathcal{S}_{neg}^d$), which is taken from within the batch, is calculated in a similar fashion and the decoder document aware loss is calculated as follows:

$$\mathcal{L}_{da}^{dec} = -log \frac{e^{\mathcal{S}_{pos}^d}}{e^{\mathcal{S}_{pos}^d} + e^{\mathcal{S}_{neg}^d} + e^{\mathcal{S}_{rnd}^d}} \tag{5}$$

We also utilize the average pooled output of the Encoder to calculate a margin loss between the query and its corresponding document context dc.

The average pooled output of the Encoder is calculated as follows:

$$\mathcal{E}_q = \text{Linear}(\text{AvgPool}(\text{Encoder}(q))), \tag{6}$$

where $\mathcal{E}_q \in \mathbb{R}^{b \times h}$ represents the encoder output.

The distances between the query embedding $\mathcal{E}q$ and the embeddings of the positive document context ($\mathcal{E}_{dc_{pos}}$), soft negatives ($\mathcal{E}_{dc_{neg}}$), and random negatives ($\mathcal{E}_{dc_{rnd}}$) are calculated as follows:

$$\mathcal{K}_{pos} = ||\mathcal{E}_q - \mathcal{E}_{dc_{pos}}||, \tag{7}$$

$$\mathcal{K}_{neg} = ||\mathcal{E}_q - \mathcal{E}_{dc_{neg}}||, \tag{8}$$

$$\mathcal{K}_{rnd} = ||\mathcal{E}_q - \mathcal{E}_{dc_{rnd}}||. \tag{9}$$

The margin loss can be calculated using the following equation:

$$\mathcal{L}_{neg} = \max(0, \gamma + \mathcal{K}_{pos} - \mathcal{K}_{neg}) \tag{10}$$

$$\mathcal{L}_{rnd} = \max(0, \gamma + \mathcal{K}_{pos} - \mathcal{K}_{rnd}) \tag{11}$$

where γ is a predefined margin that controls the separation between positive and negative samples.

This margin loss helps the model learn to differentiate between the query and relevant and non-relevant documents, thereby improving retrieval performance. The overall document-aware loss is calculated as follows:

$$\mathcal{L}_{da}^{enc} = 0.5 \cdot \mathcal{L}_{rnd} + 0.5 \cdot \mathcal{L}_{neg} \tag{12}$$

3.5 Curriculum Learning

Easy to Hard Curriculum learning [2] is a training strategy in which the complexity of the training samples is gradually increased over time to improve the model's learning process. In the context of contrastive learning for document retrieval, we apply curriculum learning to sample negatives of varying difficulty as training progresses. This approach is closely related to the curriculum knowledge distillation done in GripRank [1]. The curriculum learning approach can be described as follows:

(1) **Initial Steps ($0 \leq t \leq T_0$)** In the initial steps of training, the model randomly samples from the top 100 documents retrieved by the model as soft negatives. This introduces diversity in the negative samples and helps the model learn general patterns in the data.

(2) **Intermediate Steps ($T_0 < t \leq T_1$):** As training progresses, the model begins sampling from the top 5 documents with a probability calculated as:

$$p_{top5} = 0.75 \cdot \left(\frac{t}{T_1}\right)^{0.5}, \tag{13}$$

where t represents the current training step and T_1 is the end of the intermediate training phase. This gradual transition increases the difficulty of the negative samples, encouraging the model to focus on more challenging cases.

(3) **Later Steps ($t > T_1$):** After T_1, the model samples from the most relevant documents as the soft negatives for the remaining training steps with a probability of around 0.75. This exposes the model to increasingly difficult negatives, helping it fine-tune its understanding of the most relevant documents.

By gradually increasing the difficulty of the training samples using curriculum learning, the model can better focus on differentiating between relevant and non-relevant documents, thereby enhancing its retrieval performance.

3.6 Document Aware Label Mapping

The Language Model weights W^{lm} at the top of the decoder can be viewed as a scope-reduced document index, drawing similarities to techniques like product quantization. The decoder's output at the i-th position serves as a query representation, while the W^{lm} at the i-th position consists of c(number of child per node) relevant vectors. Essentially, the model aims to find the vector with the highest dot product, corresponding to the respective token in the docid. This process resembles a dense retriever with scope reduction algorithms such as product quantization.

Building on this concept, we aim to maximise the similarity between the weights of a token corresponding to the docid ($W_{d_i}^{lm}$) and the document context representations ($\mathcal{D}_{dc_{pos}}^{i}$). We keep this ($W_{p_i}^{lm}$) as the label map representations $\mathcal{D}_{lm_{pos}}^{i}$ from the language modeling weights $W^{lm} \in \mathbb{R}^{h \times v}$ for docid $p = \{p_0, p_1, ...p_{|p|-1}\}$. where $W_{d_i}^{lm}$ is the d_i-th column of W^{lm}.

The loss is then calculated based on the Similarity between the label map representations $\mathcal{D}_{lm}^{i}$ and the document context representations $\mathcal{D}_{dc_{pos}}^{i}$ to enhance the model's ability to learn meaningful and precise token-context relationships. This looks in equation as follows:

$$\mathcal{S}_{lm_{pos}} = \sum_{i=0}^{|p|} \mathcal{D}_{dc_{pos}}^{i} \cdot (\mathcal{D}_{lm_{pos}}^{i})^T, \tag{14}$$

Similarly the similarity score $\mathcal{S}_{lm_{pos}}$ and $\mathcal{S}_{lm_{pos}}$ is calculate for the negative and random passage token representations $\mathcal{D}_{lm_{neg}}^{i}, \mathcal{D}_{lm_{rnd}}^{i}$ corresponding to docid $n = \{n_0, n_1, ...n_{|n|-1}\}$ and the docid $r = \{r_0, r_1, ...r_{|r|-1}\}$. Using this the Document Aware Label Mapping Loss is calculated as:

$$\mathcal{L}_{dalm} = -log \frac{e^{\mathcal{S}_{lm_{pos}}}}{e^{\mathcal{S}_{lm_{pos}}} + e^{\mathcal{S}_{lm_{neg}}} + e^{\mathcal{S}_{lm_{rnd}}}} \tag{15}$$

And the main model is trained by optimising all the loss joints as mention in the following equation:

$$\mathcal{L}_{dagr} = \mathcal{L}_{dalm} + \mathcal{L}_{tok} + \mathcal{L}_{da}^{dec} + \mathcal{L}_{da}^{enc} \tag{16}$$

4 EXPERIMENTAL SETUP

4.1 Dataset

4.1.1 Natural Questions. We conduct training for the retrieval model using a widely recognized open-source question answering dataset called Natural Questions (NQ) [10]. Introduced by Google in 2019, Natural Questions consists of approximately 320,000 query-document pairs, providing a substantial dataset for training and evaluating retrieval models. The documents in the dataset are sourced from Wikipedia pages, offering a broad and diverse range of topics and domains. The queries in NQ are naturally phrased questions, mirroring real-world information-seeking behavior and presenting a practical challenge for retrieval systems. This diversity and realism in the dataset allow models to generalize better to a variety of queries and document contexts.

Table 1: Results on Natural Questions compared strong baselines across Sparse, Dense and Generative Retrieval Models

Model	R@1	R@10	R@100	MRR@100
BM25 [21]	15.11	32.48	50.54	21.07
BM25 + DocT5Query [16]	35.43	61.83	76.92	44.47
DPR [9]	50.2	77.7	90.09	59.9
ANCE [26]	52.63	80.38	91.31	62.84
DSI_{atomic} [22]	49.40	65.30	76.1	55.2
GENRE [5]	55.2	67.3	75.4	59.9
SEAL (Large) [4]	59.93	81.24	90.93	67.70
Ultron_{URL} [27]	61.2	77.4	85.9	67.50
NCI [24]	65.86	85.20	92.42	73.12
T5 (Baseline)	64.04	83.36	90.73	71.31
DAGR (Ours)	**66.12**	**84.44**	**92.15**	**73.26**

4.2 Implementation Details

For the hierarchical semantic identifier, the k-means algorithm is applied to the document embeddings obtained from a 12-layer BERT model. The number of centroids (k) at each layer is set to 30, and the maximum number of clusters for the terminal condition is set to 30 as well. For training the baseline model, we use the T5 implementation from the Hugging Face Transformers library. We initialize the weights of the encoder using the pre-trained T5 model [18], while the decoder weights and other components are randomly initialized. The initial base model is trained with a learning rate of 1×10^{-4} on two NVIDIA V100 GPUs with 32 GB of memory each, and a batch size of 128 per device, resulting in a total batch size of 256. The maximum token length for all inputs is set to 64. For the document-aware model, we train on two NVIDIA V100 GPUs with a batch size of 32 per device and a gradient accumulation of 4. The margin value (λ) is set to 1 for all margin loss calculations, and the token size for both the query and document context is set to 64. Regarding curriculum learning, we define T_0 as the number of steps in one epoch and set T_1 as the total number of steps in 15 epochs. Base model is trained on the dataset with queries from ground truth queries, pseudo queries from DocTTTTTQuery and document as queries for 850k steps. For the DAGR model we train on the top base model with $\mathcal{L}_{dagr}$ on only the ground truth queries for 50k steps. We follow the same steps provided in the NCI's[24] GitHub page. The NQ dataset consist of around 100k unique documents and around 300k unique queries.

4.3 Baselines

We compare our models to strong baselines across all types of retrieval models. For sparse retrieval, we compare our models to BM25 using raw documents and an approach using augmented queries from DocT5Query [16]. For dense retrieval models, we compare our models with DPR [9] and ANCE [26]. For generative retrieval models, we use DSI [22], GENRE [5], SEAL [4], Ultron [27], and NCI [24] as baselines for comparison. All the results are sourced either from [24] or their respective original papers.

4.4 Evaluation Metrics

Recall@n is the standard evaluation metric for the retrieval task, this is formulated as follows:

$$Recall = \frac{|(relevant\ docs) \cap (retrieved\ docs)|}{|(relevant\ docs)|} \tag{17}$$

In this case $|(relevant\ article)| = 1$ for all the query will makes $Recall = |(relevant\ article) \cap (retrieved\ article)|$, and this recall will be "1" if the model retrieves the relevant article and "0" otherwise. R@k accounts for total article retrieved by the model which k, it is controlled by user. Recall in this case can also be consider as accuracy as for R@k the model will retrieve k articles and we score 1 if the relevant article is present in the k article retrieved and "0" other wise this similar to the accuracy metric.

Mean Reciprocal Rank (MRR) is a metric used to evaluate the effectiveness of search engines, recommendation systems, and information retrieval systems. It is particularly common in the context of ranking-based evaluations. MRR is defined as the average of the reciprocals of the ranks of the first correct item.

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{18}$$

Where:

- $|Q|$ the total number of queries or instances.
- $rank_i$ is the rank of the first correct item for the i-th query.

5 RETRIEVAL PERFORMANCE ON NQ

From table 1 our model DAGR performs modestly better compared to most of the models from respective class of retrievers across all 4 different metrics on Natural questions. We can notice that NCI is second best performing according to R@1 and MRR@100. DAGR also out performs our Base T5 model suggesting the importance of the document aware representations. The table 2 show the result of ablation studies. we do experiments by removing $\mathcal{L}_{DALM}$ or curriculum and study the performance of the model. we also replace the $\mathcal{L}_{ca}^{enc}$ with Listwise loss similar to $\mathcal{L}_{da}^{dec}$ and compare the performance to the token based learning to rank.

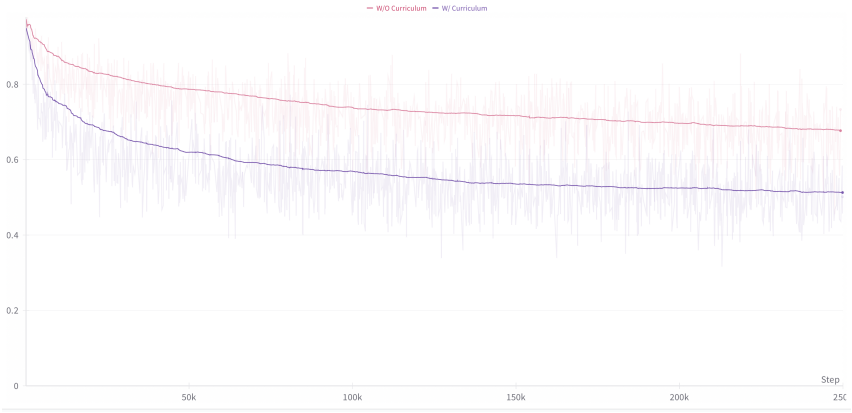

Figure 1: $\mathcal{L}_{neg}$ logged at every step, compared with and without curriculum learning.

Table 2: Ablation Results on Natural Question. All of these have $\mathcal{L}^{tok}$ in the loss component.

Model	R@1	R@10	R@100	MRR@100
DAGR (Ours)	**66.12**	**84.44**	**92.15**	**73.26**
T5 (Baseline)	64.04	83.36	90.73	71.31
w/o $\mathcal{L}_{dalm}$	66.02	84.19	91.12	73.13
w/ $\mathcal{L}_{da}^{enc}$ only	65.84	83.36	90.73	71.31
w/o Curriculum	65.79	84.18	91.21	72.73
Learning to Rank[15]	65.20	81.24	88.35	71.20
w/ $\mathcal{L}_{da}^{enc}$ as Listwise	65.52	83.74	90.85	72.33

Without DALM Loss (RQ1): Excluding the $\mathcal{L}_{DALM}$ loss from the model training results in a performance decrease of approximately 0.36%. This suggests that incorporating the document-aware loss to modify the language model head is beneficial for the model to learn token weights that are closer to the embeddings related to the context of the document.

Without Curriculum Learning (RQ2): When not using curriculum learning and instead using only the top 5 most relevant documents as negative samples, performance decreases by around 0.33 in Recall@1%. Figures 1 illustrate that contrastive loss is lower when curriculum learning is used compared to not using it. Furthermore, even several steps after the curriculum phase, the loss remains consistently lower. This attributes to the efficacy of curriculum learning to allow model some time learn better representations in initial steps and lets it do well for harder negatives in later stages

Listwise Loss (RQ3) and $\mathcal{L}_{ltr}$(Learning to Rank) Loss (RQ5): Comparing the model performance using listwise and LTR loss, the DAGR model achieves improvements of 0.6% and 0.9% in Recall@1 metric respectively. Performance of margin loss at encoder level is also better than both the Listwise loss and $\mathcal{L}_{ltr}$.

Performance on Unseen Documents While Post-tuning (RQ4): We evaluate the model's performance on documents that were not present during post-tuning. These documents don't have ground truth queries but are included in the test set, allowing us to assess the model's ability to generalize and avoid overfitting. As shown in Table 3, the zero-shot performance of DAGR is slightly better than that of the baseline T5 model, suggesting that the model's generalization capability is on par with or slightly better than the base model. This indicates that our approach effectively balances training on the available data while still maintaining strong performance on unseen documents.

Table 3: Zero Shot Results on Natural Question

Model	R@1	R@10	R@100	MRR@100
DAGR (Ours)	**66.12**	**84.44**	**92.15**	**73.26**
T5 (Baseline)	64.04	83.36	90.73	71.31
Zero Shot DAGR	50.48	70.88	81.54	57.99
Zero Shot Base	49.28	68.72	80.06	56.24

6 CONCLUSION

This work delves into the one possible way to interact query and document during the training. Since this interactive training is done after the initial generative retrieval model any model could be further fine-tuned with this technique. The experiment on using this fine-tuning approach on models with PAWA [24] and other generative retrieval models could be considered for a future work.

REFERENCES

[1] Jiaqi Bai, Hongcheng Guo, Jiaheng Liu, Jian Yang, Xinnian Liang, Zhao Yan, and Zhoujun Li. 2023. GripRank: Bridging the Gap between Retrieval and Generation via the Generative Knowledge Improved Passage Ranking. *arXiv preprint arXiv:2305.18144* (2023).

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, Vol. 382. 41–48.

[3] Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. 2024. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems* 36 (2024).

[4] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Wen-tau Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *arXiv preprint arXiv:2204.10628* (2022).

[5] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904* (2020).

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241* (2018).

[8] Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282* (2020).

[9] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).

[10] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. https://doi.org/10.1162/tacl_a_00276

[11] Hyunji Lee, Sohee Yang, Hanseok Oh, and Minjoon Seo. 2022. Generative Retrieval for Long Sequences. *arXiv preprint arXiv:2204.13596* (2022).

[12] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[13] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[14] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview identifiers enhanced generative retrieval. *arXiv preprint arXiv:2305.16675* (2023).

[15] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2024. Learning to rank in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 8716–8723.

[16] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[17] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. 2020. KILT: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252* (2020).

[18] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[19] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[20] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[21] Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

[22] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems* 35 (2022), 21831–21843.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[24] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, et al. 2022. A neural corpus indexer for document retrieval. *Advances in Neural Information Processing Systems* 35 (2022), 25600–25614.

[25] Zihan Wang, Yujia Zhou, Yiteng Tu, and Zhicheng Dou. 2023. NOVO: Learnable and Interpretable Document Identifiers for Model-Based IR. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (<conf-loc>, <city>Birmingham</city>, <country>United Kingdom</country>, </conf-loc>) *(CIKM '23)*. Association for Computing Machinery, New York, NY, USA, 2656–2665. https://doi.org/10.1145/3583780.3614993

[26] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*.

[27] Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, Peitian Zhang, and Ji-Rong Wen. 2022. Ultron: An ultimate retriever on corpus with a model-based indexer. *arXiv preprint arXiv:2208.09257* (2022).