CorpusLM: Towards a Unified Language Model on Corpus for Knowledge-Intensive Tasks

Xiaoxi Li
Gaoling School of Artificial
Intelligence
Renmin University of China
Beijing, China
xiaoxi_li@ruc.edu.cn

Zhicheng Dou*
Yujia Zhou
Gaoling School of Artificial
Intelligence
Renmin University of China
Beijing, China
{dou,zhouyujia}@ruc.edu.cn

Fangchao Liu
Huawei Poisson Lab
Beijing, China
liufangchao@huawei.com

ABSTRACT

Large language models (LLMs) have gained significant attention in various fields but prone to hallucination, especially in knowledge-intensive (KI) tasks. To address this, retrieval-augmented generation (RAG) has emerged as a popular solution to enhance factual accuracy. However, traditional retrieval modules often rely on large document index and disconnect with generative tasks. With the advent of generative retrieval (GR), language models can retrieve by directly generating document identifiers (DocIDs), offering superior performance in retrieval tasks. However, the potential relationship between GR and downstream tasks remains unexplored. In this paper, we propose **CorpusLM**, a unified language model that leverages external corpus to tackle various knowledge-intensive tasks by integrating generative retrieval, closed-book generation, and RAG through a unified greedy decoding process. We design the following mechanisms to facilitate effective retrieval and generation, and improve the end-to-end effectiveness of KI tasks: (1) We develop a ranking-oriented DocID list generation strategy, which refines GR by directly learning from a DocID ranking list, to improve retrieval quality. (2) We design a continuous DocIDs-References-Answer generation strategy, which facilitates effective and efficient RAG. (3) We employ well-designed unsupervised DocID understanding tasks, to comprehend DocID semantics and their relevance to downstream tasks. We evaluate our approach on the widely used KILT benchmark with two variants of backbone models, i.e., T5 and Llama2. Experimental results demonstrate the superior performance of our models in both retrieval and downstream tasks.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**.

KEYWORDS

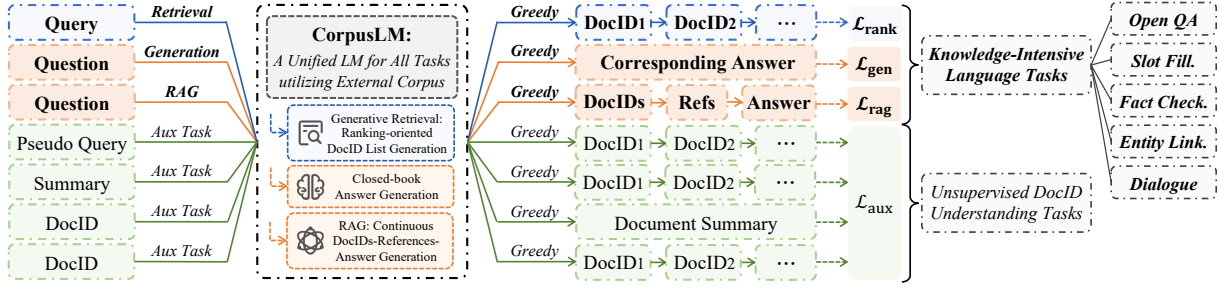Generative Retrieval, RAG, Knowledge-Intensive Language Tasks

*Corresponding author.

1 INTRODUCTION

Large language models (LLMs) have recently revolutionized fields such as question answering (QA), dialogue, and information retrieval, demonstrating impressive capabilities in a variety of language tasks [3, 8, 9, 49, 62]. However, LLMs often face the problem of "hallucination", where the generated text may contain misleading or false information [20]. This issue is particularly severe in knowledge-intensive (KI) tasks, such as slot filling [12] and open-domain question answering [24]. To address this, a popular approach is the use of retrieval-augmented generation (RAG), which involves a retriever to obtain relevant context from a large knowledge corpus, followed by a generator model that synthesizes the retrieved context into coherent responses [19, 27, 66].

Traditionally, the retrieval component follows an index-centric framework [22, 23, 42, 56, 61]. Although this approach is widely used, it has several drawbacks. Notably, the requirement for a large document index to search the entire corpus results in considerable memory footprint. Furthermore, during training, the disparate model structures for retrieval (matching similarity) and generation (auto-regressive) [2, 27, 31, 36] hinder the joint optimization of both models. This limitation restricts the understanding of the relationship between both tasks.

Recently, generative retrieval (GR) has emerged as a promising paradigm [4, 35], which employs auto-regressive generative models to retrieve relevant documents by directly generating document identifiers (DocIDs) [47]. This method has shown improved performance in web search and question-answering (QA) scenarios. While prior research has focused on enhancing model training [46, 53, 63], DocID design [30, 45, 54, 65], and task adaption [6, 7] for better retrieval performance, the unification of generative retriever and downstream generator is often overlooked. Furthermore, the potential of LLMs in the field of generative retrieval remains unexplored.

To address the above problem, we propose **CorpusLM**, a **unified** language model that utilizes external corpus and seamlessly integrates generative retrieval, closed-book generation, and retrieval-augmented generation to handle a wide range of knowledge-intensive

Figure 1: Overview of the CorpusLM framework. We aim to develop a unified language model that utilizes external corpus to handle various knowledge-intensive tasks by integrating generative retrieval, closed-book generation, and RAG. To effectively accomplish all tasks using unified greedy decoding, we propose a ranking-oriented DocID list generation strategy to improve generative retrieval performance; and a continuous RAG strategy to sequentially decode DocID ranking list, references and answer. We also enhance the model's comprehension of DocID semantics through unsupervised DocID understanding tasks.

tasks. These tasks are closely interconnected, necessitating the model to comprehend the relationships between queries, DocIDs, and documents in order to generate accurate answers or DocID ranking lists. To achieve this, we employ a multi-task learning approach, where the generation of both DocIDs and answers is integrated into a unified model training framework. The overview of our framework is illustrated in Figure 1.

Moreover, to facilitate effective retrieval and answer generation in the unified auto-regressive greedy decoding process, we propose the following tailored training and inference strategies:

• **For retrieval**. Existing methods focus on mapping queries to a single DocID, resulting in high accuracy for the top-ranked document but lower accuracy for subsequent documents [63]. To address this, we introduce a novel **ranking-oriented DocID list generation** strategy, allowing the model to learn from a ranking list of DocIDs during training. We apply dynamic constraints during the inference to ensure generating a valid and non-repetitive list of DocIDs from the corpus. This strategy offers three advantages: (1) it enables an end-to-end solution for RAG, as generating a list directly streamlines subsequent steps; (2) training on a query-DocID ranking list can include all relevant DocIDs for a given query, hence it can reduce the training sample confusion problem that exists in existing generative retrieval methods; (3) taking into account the already generated DocIDs during the next DocID decoding can improve the overall ranking capability.

• **For RAG**, retrieved documents often contain irrelevant and redundant information, hindering the accurate generation of answers. To counteract this, we propose a **continuous DocIDs-References-Answer generation strategy**. This strategy guides the CorpusLM to first decode DocIDs and their fine-grained references from the documents before decoding the final answer. By doing so, irrelevant information is efficiently filtered out, leading to more accurate responses. Furthermore, our approach enables a continuous decoding process for RAG, eliminating the need for multiple rounds of interaction for document retrieval and answer generation, thereby improving efficiency.

• **DocID understanding**. It is worth noting that existing pretrained language models do not possess knowledge related to DocIDs. To enhance the model's understanding of DocIDs and establish deeper relationships between DocIDs, queries, and documents, we

introduce a set of auxiliary DocID understanding tasks alongside the main tasks of retrieval, generation, and RAG.

We evaluate our model on the KILT benchmark [37], which is a benchmark for knowledge-intensive language tasks and comprises 11 distinct datasets classified into 5 categories of KI tasks. We implement our model with two different backbone models, which are the encoder-decoder T5 model [39] and the open-source decoder-only LLM, Llama2 [50]. The experimental results showcase that our proposed CorpusLM achieves superior performance in both retrieval and downstream generation tasks.

2 RELATED WORK

Generative Retrieval. Generative retrieval (GR) is a novel approach that leverages generative models as differentiable search indices [47], allowing retrieval by directly generating relevant DocIDs. Recent research in this field mainly focuses on the following aspects: (1) Model training. A simple yet effective method involves using generated pseudo queries for training data augmentation. Subsequently, labeled query-DocID pairs are used to further fine-tune the model [29, 46, 53, 64, 67]. To enhance the ranking ability of the GR models, Zhou et al. [63] apply reinforcement learning to train a reward model with various annotators, including sparse&dense retrievers, and LLMs, to provide relevance feedback for GR models. (2) DocID design. Drawing inspiration from DSI [47], existing studies explore various approaches such as atomic identifiers, text fragments [1, 6, 64], semantic clusters and residual quantization [59]. For instance, Ultron [64] utilizes the document URL and title as representations, while SEAL [1] considers all n-grams within a corpus as potential identifiers. Recently, researchers have also explored term-sets as DocIDs [60] and learnable DocIDs [45, 54, 57]. (3) Task adaptation. To better align the GR model with downstream tasks, CorpusBrain Chen et al. [7] propose pre-training methods that enhance retrieval in knowledge-intensive tasks. Furthermore, UGR Chen et al. [6] employ n-gram DocIDs tailored for retrieval at various granularities, from sentences to entire documents. However, these methods generally require an additional generator such as FID [18] to produce the final output.

Knowledge-Intensive Language Tasks. Knowledge-intensive language tasks refer to a range of NLP tasks that require accessing external knowledge sources to provide accurate results [11, 24, 37,

48, 58]. These tasks generally involve two components: a retriever and a reader. The retriever obtains relevant information from a large-scale knowledge source and the reader subsequently utilizes specialized downstream models to read the retrieved context and provide more accurate answers [2, 15, 31, 32, 36].

To enhance the performance of RAG models, various strategies have been developed, which involve training the retriever and reader modules separately using dense retrieval methods, such as DPR [22], MT-DPR [33], DRQA [5] and FID [18]. Other strategies focus on joint training of both modules, either by updating the query encoder while keeping the document index static [27], or by updating both the query and document encoders and asymptotically updating the document index during training [19, 43, 44]. Based on generative retrieval methods, SEAL [1], CorpusBrain [7] and UGR [6] train the GR models and also utilize separate generators to accomplish downstream KILT tasks. Recently, UniGen [28] introduced a unified framework for concurrent learning of GR and QA tasks. However, it uses separate decoders to generate DocIDs and answers, restricting its ability to generalize and scale.

3 METHODOLOGY

3.1 Task Formulation

In this work, we focus on addressing knowledge-intensive language tasks through generative approaches, specifically generative retrieval, closed-book answer generation, and retrieval-augmented answer generation. These tasks can all be accomplished using auto-regressive language models.

Formally, let's consider a document d in a document corpus, and let d' represent the pre-built identifier for document d. For **generative retrieval**, given a query q, we determine the relevance $\mathcal{R}$ between q and each document d using the ranking model f_{rank} with parameters θ:

$$\mathcal{R}(q, d) = f_{\text{rank}}(d'|q; \theta) = \prod_{i=1}^{T} f_{\text{rank}}(d_i'|d_{<i}', q; \theta), \quad (1)$$

where T is the length of the document identifier d', and d_i' refers to the i-th token in d', and $d_{<i}'$ represents the generated identifier up to position i.

For **closed-book answer generation**, given an input query q, we estimate the probability $\mathcal{A}$ of generating a target answer a using the generation model f_{gen} with parameters ϕ:

$$\mathcal{A}_{\text{gen}}(a|q) = f_{\text{gen}}(a|q; \phi) = \prod_{i=1}^{T_a} f_{\text{gen}}(a_i|a_{<i}, q; \phi), \quad (2)$$

where T_a is the total number of tokens in a, and a_i refers to the i-th token in a, and $a_{<i}$ represents the generated answer up to index i.

For **retrieval-augmented answer generation**, the probability of generating the target answer a for input query q, taking into account the set of retrieved documents $\mathcal{D} = \{d_1, d_2, \ldots, d_k\}$, is quantified as follows:

$$\mathcal{A}_{\text{rag}}(a|q, \mathcal{D}) = f_{\text{rag}}(a|q, \mathcal{D}; \mu) = \prod_{i=1}^{T_a} f_{\text{rag}}(a_i|a_{<i}, q, \mathcal{D}; \mu), \quad (3)$$

where f_{rag} represents the model that generates the final answer after considering the retrieved documents, parameterized by μ.

In this paper, we aim to develop a unified language model for all tasks, so we will use a consistent symbol f with parameters θ to denote all tasks defined in Equation (1)-(3) in subsequent sections.

3.2 CorpusLM: the Unified Language Model

CorpusLM is a multi-task learning architecture designed to handle various types of knowledge-intensive tasks. It is a unified language model capable of performing generative retrieval, closed-book generation, and retrieval-augmented generation through the same auto-regressive greedy generation. The model identifies different tasks using specific prefixes.

The overview of CorpusLM is illustrated in Figure 1. The training of CorpusLM involves the following three basic tasks:

- **Generative Retrieval:** retrieving relevant documents to a given query by generating a ranked DocID list, facilitating the model's ranking ability, and can be achieved through greedy decoding.
- **Closed-book Generation:** generating answers solely based on the query input, without relying on external information, similar to classic auto-regressive language models.
- **Retrieval-Augmented Generation:** generating answers by first retrieving relevant content using DocID list generation, and then generate references and final response through continuous greedy decoding, enhancing effective and efficient RAG.

Moreover, in order to effectively integrate generative retrieval and RAG, it is necessary to improve the model's understanding of DocIDs and the relationship between DocIDs and their corresponding knowledge. Therefore, we incorporate a group of unsupervised DocID understanding tasks into the multi-task learning framework to enhance the model's understanding of the meaning behind DocIDs:

- **DocID Understanding:** The model is equipped with auxiliary tasks that deepen its understanding of DocIDs' structure and semantic meaning.

By training on the above four types of tasks that share common patterns and connections, CorpusLM develops a more comprehensive understanding of the relationships between retrieval and downstream tasks, as well as the meaning behind the DocIDs, thereby gaining a more robust grasp of each individual task. Formally, the training of CorpusLM aims to optimize objectives with a combined loss function as below:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{rank}} + \lambda_2 \mathcal{L}_{\text{gen}} + \lambda_3 \mathcal{L}_{\text{rag}} + \lambda_4 \mathcal{L}_{\text{aux}}, \quad (4)$$

where $\mathcal{L}$rank, $\mathcal{L}_{\text{gen}}$, and $\mathcal{L}$rag are corresponding loss functions for generative retrieval, closed-book generation, and retrieval-augmented generation represented by Equations (1)-(3). $\mathcal{L}$aux is the loss function for the DocID Understanding task. The specific forms of these loss functions will be explained in subsequent sections. λ_1, λ_2, λ_3, and λ_4 are weighting coefficients for each task's loss.

3.3 Generative Retrieval: DocID List Generation

To address the limitation of traditional generative retrieval models, which excel at generating the top-1 DocID but often lose accuracy on subsequent DocIDs [63], we introduce a ranking-oriented DocID decoding strategy in CorpusLM. Instead of using single query-DocID training pairs and beam search, our strategy allows the model to learn from an entire DocID ranking list. By using the greedy decoding method, the model can continuously generate a DocID ranking list using natural language expressions. This approach improves the ranking capability of CorpusLM and promotes joint learning between retrieval and downstream generation tasks.
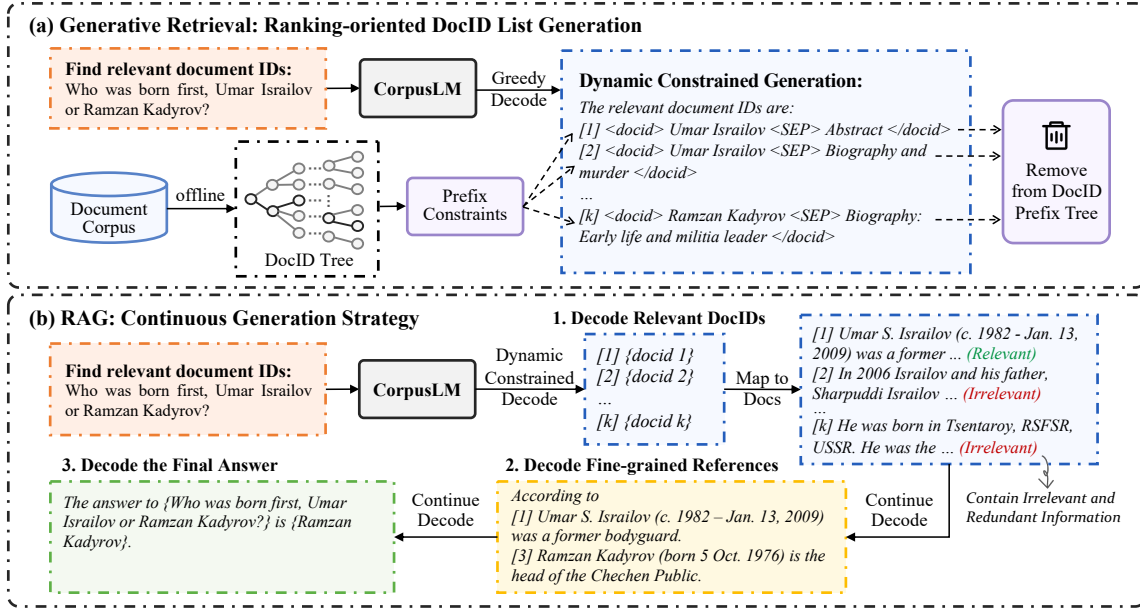
Figure 2: Illustration of the two generation strategies for retrieval and RAG. (a) Ranking-oriented DocID list generation strategy. We utilize a prefix tree built from DocIDs in the corpus to dynamically add constraints for generating valid and non-repeated DocID ranking list. (b) Continuous generation strategy. It comprises three continuous decoding steps: (1) decode DocIDs and map them to the corresponding documents; (2) decode fine-grained references from the documents; (3) decode the final answer.

3.3.1 Training Data Construction. The training process involves learning from a ranked list of relevant DocIDs. However, the challenge lies in the fact that training queries are typically associated with just one or a few relevant documents, which limits the model's capacity to learn effective ranking. To overcome this, we enhance the training data with additional relevant documents. We start with the original training queries and their respective answers. We first use a BM25 [42] retriever to produce an initial list of potentially relevant documents. Following this, we employ a dense re-ranker SimLM [52] to re-rank the retrieved documents. We create a final list of top-k DocIDs by taking the reranked documents and appending them to the original labeled DocIDs, ensuring there are no duplicates in the combined list.

Through this method, each query in the training set is associated with a comprehensive list of top-k relevant DocIDs, enabling the model to understand the relevance between multiple ranked DocIDs and the given query, thus reducing training confusion and improving its ranking capability. The final training data format is depicted in Figure 2, where all of the relevant DocIDs is formatted in a single natural language sequence.

3.3.2 Training Objective. The optimization goal for this task is to maximize the overall relevance of the ranked list of document identifiers d' for each query q, as defined in Equation (1). The optimization objective for generating the ranked list of DocIDs is formalized as: $O_{\text{rank}} = \max_\theta \sum_{d \in \mathbf{D}} \mathcal{R}(q, d)$, where $\mathbf{D}$ represents the set of documents that are to be ranked for the query q.

To optimize O_{rank} and perform ranking in a natural language format, the ranking loss $\mathcal{L}_{\text{rank}}$ of generative retrieval is defined as:

$$\mathcal{L}_{\text{rank}} = -\sum_{i=1}^{T_d} \log f(s_i | s_{<i}, q; \theta), \tag{5}$$

where T_d denotes the length of tokens in the sequence comprised of the target ranked DocID list. The ranking loss is designed to encourage the model to predict the correct sequence of ranked DocIDs by maximizing the log-likelihood of each token s_i, given the preceding tokens $s_{<i}$, and the query q.

This approach helps to ensure that the model can generate a sequence of DocIDs that reflects the correct ranking order, also enable ranking-aware generative retrieval in natural language expressions, ultimately improving the retrieval performance of CorpusLM and unification with other generative NLP tasks.

3.3.3 Inference Constraints. During inference, we employ a DocID prefix tree constructed from the documents in the corpus to add constraints that ensure the generation of valid, non-repetitive DocIDs. As depicted in Figure 2(a), the process involves dynamic activation of constraints during the generation of each token:

(1) If the token is the DocID start symbol "<docid>", we enforce prefix constraints on the succeeding sequence.

(2) If the token is the DocID end symbol "</docid>", we lift the constraints and remove the previously generated DocID from the prefix tree to prevent duplication in subsequent generations.

(3) For all other tokens, we scan backward; if "</docid>" is encountered, scanning exits. If "<docid>" is found, constraints are applied to the sequence following to ensure the current token's validity within the prefix tree.

3.4 Closed-book Answer Generation

For closed-book answer generation, similar to auto-regressive language models like T5 [39] and GPT [38], the model's objective is to

maximize the probability in Equation (2) with the following loss:

$$\mathcal{L}_{\text{gen}} = - \sum\nolimits_{i=1}^{T_a} \log f(a_i | a_{<i}, q; \theta). \tag{6}$$

3.5 RAG: Continuous Generation Strategy

In order to achieve all KI tasks in a unified greedy decoding process and facilitate effective and efficient retrieval-augmented generation, this section introduces the continuous DocIDs-References-Answer decoding strategy. Notably, in traditional RAG methods, the retrieved documents often contain irrelevant and redundant information that can hinder the accurate generation of answers. Therefore, we propose decoding fine-grained references extracted from the documents first before decoding the final answer.

Three-stage Decoding: As shown in Figure 2(b), our RAG decoding method consists of three steps: (1) Decode relevant DocIDs and map them to the corresponding documents in the decoder of CorpusLM. (2) Continue to decode fine-grained references extracted from the documents. (3) Continue to decode the final answer.

Continuous Decoding: We achieve these three decoding steps within a single decoding process, rather than reorganizing the input content and relying on multiple input-output iterations. Instead, after retrieving the documents, we directly continue decoding the references and then proceed to decode the final answer.

Noise Sampling: To prevent the model from overly relying on the previous reference when generating the final answer, we introduce the noise sampling strategy. During training, there is a probability τ of randomly selecting a sentence from the retrieved documents and replacing the original ground-truth reference.

Training Objective: Given a query q, the model's objective is to first generate a reference r after retrieving the relevant documents $\mathcal{D} = \{d_1, d_2, \ldots, d_k\}$. The training process involves minimizing the loss function $\mathcal{L}_{\text{ref}}$, which is defined as:

$$\mathcal{L}_{\text{ref}} = - \sum\nolimits_{i=1}^{T_r} \log f(r_i | r_{<i}, q, \mathcal{D}; \theta), \tag{7}$$

where T_r is the length of the reference r, and r_i denotes the i-th token in the reference sequence. This expects the model to learn to extract fine-grained references after the retrieved passages.

Subsequently, the model generates the final answer a. The loss function for answer generation is defined as:

$$
\begin{aligned}
\mathcal{L}_{\text{ans}} = &- \sum\nolimits_{i=1}^{T_a} \log f(a_i | a_{<i}, q, \mathcal{D}, r; \theta)) \\
&- \tau \sum\nolimits_{i=1}^{T_a} \log f(a_i | a_{<i}, q, \mathcal{D}, r_{\text{noise}}; \theta)),
\end{aligned}
\tag{8}
$$

where T_a is the length of the answer a, $\mathcal{D}$ represents the set of retrieved documents used during training, and r_{noise} denotes the randomly sampled sentence used as the noised reference.

The overall training loss $\mathcal{L}_{\text{rag}}$ is the sum of losses for reference generation and answer generation:

$$\mathcal{L}_{\text{rag}} = \mathcal{L}_{\text{ref}} + \mathcal{L}_{\text{ans}}. \tag{9}$$

This training approach encourages the CorpusLM not only to generate useful references from documents that will aid in the final answer generation, but also to generate the final answer from the documents and the references.

Table 1: Details of datasets in the KILT benchmark.

Dataset	Category	Train Size	Dev Size
FEVER [48]	Fact Checking	104,966	10,444
AIDA CoNLL-YAGO [16]	Entity Linking	18,395	4,784
WNED-WIKI [14]	Entity Linking	-	3,396
WNED-CWEB [14]	Entity Linking	-	5,599
T-REx [12]	Slot Filling	2,284,168	5,000
Zero Shot RE [25]	Slot Filling	147,909	3,724
Natural Questions [24]	Open Domain QA	87,372	2,837
HotpotQA [58]	Open Domain QA	88,869	5,600
TriviaQA [21]	Open Domain QA	61,844	5,359
ELI5 [13]	Open Domain QA	272,634	1,507
Wizard of Wikipedia [11]	Dialogue	63,734	3,054

3.6 Unsupervised DocID Understanding Tasks

Since pre-trained language models lack inherent understanding of DocIDs, we introduce the following tasks to enhance their understanding of DocIDs and align their knowledge with DocIDs:

- **Predicting DocIDs from Pseudo Queries:** The model generates a ranked list of relevant DocIDs given a pseudo query, thereby enhancing ranking performance.
- **Predicting DocIDs from Document Summaries:** The model predicts the ranking list of DocIDs given a document summary, further improving ranking performance.
- **Reciting Content Summary form DocID:** The model generates a summary given a DocID, facilitating better memorization of the main content associated with DocIDs.
- **Predicting Related DocIDs form DocID:** The model generates a ranked list of DocIDs related to a given DocID, enabling the model's learning of document-level relevance.

For these tasks, pseudo queries and summaries are generated using the Llama2-7B model [50] with specific prompts. To enhance the model's ranking ability, the ranking data augmentation is applied to the above DocID prediction tasks, as discussed in Section 3.3.1.

All auxiliary tasks are formulated as seq2seq generative tasks. During training, he model is trained to maximize the likelihood of generating the target sequence. Teacher forcing is used to optimize the cross-entropy loss $\mathcal{L}_{\text{aux}}$ in Equation (4), given by:

$$\mathcal{L}_{\text{aux}} = - \sum\nolimits_{i=1}^{T_c} \log f(c_i | c_{<i}; \theta), \tag{10}$$

where T_c represents the length of the target sequence c for the auxiliary task, c_i represents the ith token in c, and $c_{<i}$ corresponds to the generated content up to position i.

4 EXPERIMENTAL SETTINGS

4.1 Datasets

To evaluate CorpusLM's retrieval and generation performance, we employ KILT (Knowledge-Intensive Language Tasks) benchmark [37], an extensive benchmark that encompasses 11 datasets across 5 knowledge-intensive natural language processing tasks, including fact checking, entity linking, slot filling, open-domain question answering, and dialogue. We evaluate CorpusLM's performance in retrieval, closed-book generation, and retrieval-augmented generation. Detailed statistics can be found in Table 1.

Table 2: Overall retrieval performance on KILT dev set. We report passage-level R-Precision (%). Models are emphasized with the best in bold and the second in <u>underline</u>. The symbol "†" signifies that our model achieved superior results among all baselines in a statistically significant manner (t-test, $p < 0.05$).

Method	FC	Entity Linking			Slot Filling		Open Domain QA				Dial.
	FEVER	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
Sparse&Dense Retrieval											
BM25	30.29	2.82	1.38	3.84	32.04	43.37	12.34	31.31	14.40	1.20	17.20
DPR	59.10	79.51	-	-	60.61	70.91	31.13	39.47	35.48	-	37.66
MT-DPR	64.05	81.69	49.20	46.95	57.64	73.81	32.80	38.42	36.29	10.86	38.00
RAG	66.04	76.40	48.28	46.01	53.57	67.97	38.25	34.61	41.38	10.70	38.04
E5	68.52	79.72	50.47	48.10	54.48	70.01	39.40	37.35	42.62	**11.02**	39.16
SimLM	68.06	80.11	51.98	49.54	55.42	72.11	38.58	36.11	41.80	10.36	38.31
Generative Retrieval											
T5	71.63	86.71	67.34	62.20	64.87	78.51	38.69	38.09	45.73	10.35	42.51
BART	69.90	87.43	67.22	60.71	61.57	76.13	39.84	38.44	47.26	10.09	40.19
SEAL	70.55	82.05	57.09	58.70	55.91	74.89	39.67	<u>40.54</u>	44.16	9.32	41.59
CorpusBrain	72.23	<u>88.79</u>	<u>69.40</u>	63.23	63.42	79.05	40.09	39.45	47.97	10.68	42.19
Llama2	74.39	85.53	66.55	61.45	66.12	77.90	40.59	40.37	48.43	10.66	42.69
CorpusLM (T5)	<u>75.64</u>†	**90.96**†	**70.35**†	**65.43**†	<u>68.89</u>†	**81.08**†	<u>41.46</u>†	39.31	<u>48.80</u>	10.90	**44.96**†
CorpusLM (Llama2)	**76.21**†	88.59	69.39	<u>64.18</u>†	**69.17**†	<u>80.79</u>†	**44.10**†	**42.06**†	**50.62**†	10.88	<u>43.92</u>†

The model training is conducted using the training sets, and evaluations are carried out using the development sets. As the knowledge source, we utilize the pre-processed Wikipedia passages split into sections. The passages are derived from English Wikipedia articles based on the 2019/08/01 Wikipedia dump data, consisting of a total of 5.9 million articles and 16.6 million passages. For both dense and sparse models, we refine the passages by adding the title of each article to its corresponding passage, following [7, 37].

4.2 Evaluation Metrics

For retrieval evaluation, consistent with previous works [1, 4, 27, 34, 37] and the official KILT evaluation metrics, we use R-precision [37] as the evaluation metric, which is by the fraction $\frac{r}{R}$, with R being the number of contexts in the provenance set, and r is the number of relevant contexts within the top-R retrieved contexts.

For downstream evaluation, we adopt specific metrics for different downstream tasks. Specifically, as suggested in the KILT resource paper [37], we use Accuracy for FEV, AY2, WnWi, WnCw, T-REx and zsRE; Exact Match (*EM*) for NQ, TQA and HoPo; *ROUGE-L* for ELI5; and *F1* for WoW. To ensure a fair comparison with non-finetuned generators, we include the percentage of outputs containing gold answers ("*has_answer*"), in line with [17].

4.3 Baselines

4.3.1 Baselines for Retrieval Tasks. Retrieval tasks are divided into two main approaches: *Sparse&Dense Retrieval* and *Generative Retrieval*. For *Sparse&Dense Retrieval*, we employ the following models: **BM25** [42], a classic sparse retrieval model; **DPR** [22] and its multi-task variant **MT-DPR** [34] for dense passage retrieval; **RAG** [27], which combines dense retrieval with seq2seq models for an enhanced generation; **E5** [51], a state-of-the-art text embedding model; and **SimLM** [52], a dense passage retriever with

effective pre-training methods. The Generative Retrieval methods comprise **T5** [39], a pre-trained encoder-decoder model for multi-task learning, **BART** [26], a denoising autoencoder for text generation; **SEAL** [1], generating sub-strings as document identifiers; **CorpusBrain** [7], incorporating pre-training strategies for KILT retrieval task; and **Llama2** [50], an open-source pre-trained LLM.

The baseline retrieval models are finetuned with labeled retrieval data from KILT datasets. Except the DPR model is finetuned on each specific dataset, other dense and generative retrieval models are multi-task finetuned using retrieval data across all datasets, as multi-task training in KILT leads to improved performance [7, 34].

4.3.2 Baselines for Downstream Tasks. For downstream tasks, we consider two primary streams: Closed-book Generation and Retrieval-augmented Generation. The Closed-book Generation models include **T5** [39], **BART** [26], and the open-source LLM named **Llama2-70B** [50], all leveraging their pre-trained parameters for generative tasks. For Retrieval-augmented Generation, we consider the following models: (1) **DPR+BART** [37], combining dense passage retrieval with generative BART model; (2) **RAG** [27], an end-to-end retriever-generator model; (3) **MT-DPR+FID** [34], integrating multi-task DPR retriever with a fusion-in-decoder generator [18]; (4) **BM25+Llama2-70B** and **E5+Llama2-13B&70B** models reported by [17] that utilize Llama2 [50] for answer generation.

4.4 Implementation Details

In our experiments, we utilize the T5-Base [39] (T5) and Llama2-7B-Chat [50] (Llama2) as our backbone models with pre-trained parameters from Hugging Face [55].

During training, we set the coefficients λ_{1-4} to 1 and the noise sampling probability τ to 0.2. Our training process involves a batch size of 512 and a learning rate of 3e-4. We use 2000 warm-up steps for the learning rate. To improve the training efficiency of the

Table 3: Overall downstream performance on KILT dev set, considering both closed-book and RAG settings. In different settings, models are emphasized with the best in bold and the second in underline. $\diamond$ and $\heartsuit$ indicates results from [37] and [17], respectively. Following [17], the percentage of outputs containing gold answers ("_has_answer_") is noted in parentheses.

Method	FC	Entity Linking			Slot Filling		Open Domain QA				Dial.
	FEVER	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
Closed-book Generation											
T5$^\diamond$	-	81.84	47.35	46.58	47.24	1.58	25.20	12.66	25.79	21.02	13.15
BART$^\heartsuit$	80.67	86.62	47.91	48.01	43.84	3.03	26.15	16.86	32.54	22.69	13.77
Llama2-70B$^\heartsuit$	33.6(74.9)	39.8(54.5)	42.8(53.8)	39.2(55.7)	28.5(40.5)	11.3(13.6)	19.6(37.4)	13.9(25.1)	**67.4(80.8)**	**23.00**	13.30
CorpusLM (T5)	<u>81.93</u>	**87.16**	<u>49.68</u>	**51.65**	<u>49.65</u>	3.61	<u>28.14</u>	<u>17.32</u>	25.02	21.33	<u>13.90</u>
CorpusLM (Llama2)	**85.34**	<u>86.91</u>	**56.66**	<u>50.37</u>	**51.05**	**18.22**	**29.57**	**26.13**	<u>41.34</u>	<u>22.94</u>	**14.85**
Retrieval-augmented Generation											
DPR+BART$^\diamond$	88.11	-	44.96	45.70	56.70	34.96	45.05	25.75	59.28	18.53	15.51
RAG$^\diamond$	87.70	77.40	49.00	46.70	61.48	47.42	48.78	27.68	61.73	16.11	13.28
MT-DPR+FID	88.49	79.77	49.52	47.15	79.43	69.09	50.07	36.50	69.62	15.77	15.60
BM25+Llama2-70B$^\heartsuit$	46.2(86.3)	18.0(35.9)	19.1(32.2)	14.2(30.9)	25.9(43.0)	31.4(37.8)	25.3(34.3)	25.9(33.4)	65.8(80.0)	21.30	12.20
E5+Llama2-13B$^\heartsuit$	66.3(73.5)	51.2(57.9)	48.6(51.4)	45.6(51.4)	17.2(42.3)	31.7(41.1)	36.1(43.3)	14.3(25.5)	56.3(76.2)	20.90	12.30
E5+Llama2-70B$^\heartsuit$	49.9(88.6)	51.2(57.9)	48.6(51.4)	45.6(51.4)	28.9(49.2)	35.0(43.2)	36.4(48.8)	28.1(35.8)	<u>71.1(83.9)</u>	21.50	13.20
CorpusLM (T5)	<u>89.81</u>	**87.09**	<u>50.52</u>	49.77	<u>80.68</u>	<u>70.34</u>	**53.39**	<u>40.96</u>	70.94	<u>22.13</u>	<u>16.65</u>
CorpusLM (Llama2)	**90.22**	<u>85.03</u>	**56.54**	**50.32**	**81.57**	**72.79**	<u>55.38</u>	**42.23**	**72.43**	**23.46**	**16.96**

CorpusLM (Llama2) model, we employ QLoRA [10] and Deep-Speed [40, 41] technologies. We incorporate a maximum of top-10 and top-3 ranked passages for training the DocID decoding and answer decoding, respectively. Each unsupervised DocID understanding task consists of approximately 300k training pairs.

During inference, we use dynamic constrained greedy decoding for generative retrieval tasks and greedy decoding for downstream tasks. For retrieval tasks, we limit the number of generated DocIDs to a maximum of 10. For RAG tasks, we retrieve and utilize 3 passages as the contexts for downstream tasks. All experiments are conducted on 8 NVIDIA Tesla A100 40GB GPUs.

5 EXPERIMENTAL RESULTS

5.1 Retrieval Performance

In this section, we evaluate the retrieval performance of our proposed CorpusLM models based on T5 and LLama2, compared with a range of baseline models, as presented in Table 2.

Comparison with Sparse&Dense Retrievers: Despite the competitive performance of dense retrievers like MT-DPR, E5, RAG, and SimLM, which have been fine-tuned through multi-task training, **our CorpusLM models achieve superior performance in most datasets by a significant margin (t-test, $p < 0.05$).** Specifically, on the FEVER dataset, T5 and LLama2-based CorpusLM outperform MT-DPR by 18.10% and 18.99% respectively. Similarly, on the zsRE dataset, CorpusLM (T5) and CorpusLM (LLama2) surpass SimLM by 12.44% and 12.03% respectively. We also observe that the bag-of-words method BM25, a sparse retrieval model, falls behind neural retrieval models in all tasks, highlighting the need for neural models due to the complexity of these tasks. Notably, joint training of the DPR model on multiple datasets (MT-DPR) outperforms individual dataset training in almost all cases, indicating the effectiveness of joint training for improved retrieval performance.

Comparison with Generative Retrievers: Generative retrievers generally outperform dense retrievers, indicating their effectiveness in knowledge-intensive tasks. However, they still fall short compared to our CorpusLM models. T5-based CorpusLM and LLama2-based CorpusLM models generally exhibit similar retrieval performance. LLama2-based CorpusLM performs better in fact-checking and open domain QA tasks, while T5-based CorpusLM outperforms in entity linking and dialogue tasks. **Compared to other baseline generative retrievers, our CorpusLM models consistently achieve superior performance across most datasets.** For example, on the WoW dataset, CorpusLM (T5) and CorpusLM (LLama2) outperform T5 by 5.76% and 3.31% respectively, and surpass CorpusBrain by 6.57% and 4.11% respectively. On the T-REx dataset, CorpusLM (T5) and CorpusLM (LLama2) outstrip SEAL by 23.21% and 23.72%, and surpass LLama2 by 4.19% and 4.61% respectively.

5.2 Downstream Generation Performance

In this section, we analyze the downstream generation performance of the models within the closed-book and open-retrieval settings on the KILT dev set. The results are shown in Table 3.

In the closed-book setting, **our CorpusLM models surpass traditional language models like T5 and BART, and even outperform the larger model Llama2-70B, across most datasets.** While both CorpusLM variations demonstrate similar performance, the larger parameter scale of CorpusLM (Llama2) provides a slight edge over CorpusLM (T5) across most datasets. The T5-based CorpusLM notably outshines T5 model by 10.88%, 11.67%, and 36.81% on the WnCw, NQ, and HoPo datasets respectively, and even surpasses the Llama2-70B by 31.76%, 43.57%, and 24.60%. Furthermore, on the FEVER and T-REx datasets, the Llama2-7B-based CorpusLM surpasses the Llama2-70B by 153.99% and 79.12%, respectively. They also excel in the _has_answer_ metric by 13.94% and 26.05%. These

Table 4: Ablation studies on retrieval performance. The best results are in bold for T5 and Llama2-based models, respectively.

Method	FC	Entity Linking			Slot Filling		Open Domain QA				Dial.
	FEVER	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
CorpusLM (T5)	**75.64**	**90.96**	70.35	65.43	**68.89**	**81.08**	**41.46**	**39.31**	**48.80**	**10.90**	**44.96**
w/o DocID Understanding Tasks	74.25	89.23	67.32	63.02	68.45	80.30	39.23	37.24	46.28	8.89	42.75
w/o Ranking-oriented Decode	74.03	90.67	**70.65**	**65.97**	67.98	80.78	40.17	37.13	46.92	10.06	44.70
CorpusLM (Llama2)	**76.21**	**88.59**	**69.39**	**64.18**	**69.17**	**80.79**	**44.10**	**42.06**	**50.62**	**10.88**	**43.92**
w/o DocID Understanding Tasks	75.28	87.58	66.13	62.75	69.07	79.37	42.83	40.82	49.24	10.08	42.59
w/o Ranking-oriented Decode	72.79	84.20	64.50	60.95	66.59	75.59	41.20	39.20	47.50	9.10	40.90

Table 5: Ablation studies on downstream performance in RAG setting. The best results are in bold for T5 and Llama2-based models, respectively.

Method	FC	Entity Linking			Slot Filling		Open Domain QA				Dial.
	FEVER	AY2	WnWi	WnCw	T-REx	zsRE	NQ	HoPo	TQA	ELI5	WoW
CorpusLM (T5)	**89.81**	**87.09**	**50.52**	**49.77**	**80.68**	**70.34**	**53.39**	**40.96**	**70.94**	22.13	**16.65**
w/o Decode Reference	87.21	84.17	49.09	47.97	77.95	67.92	50.47	36.42	68.50	21.16	15.85
w/o Noise Sampling	88.31	86.48	49.41	48.53	79.14	68.90	52.78	38.81	70.29	20.17	15.05
w/o Pipeline Decode	89.18	86.12	48.38	47.07	78.51	67.83	52.56	39.06	69.82	**22.39**	16.20
CorpusLM (Llama2)	**90.22**	85.03	**56.54**	**50.32**	**81.57**	**72.79**	**55.38**	**42.23**	**72.43**	**23.46**	**16.96**
w/o Decode Reference	87.40	84.49	55.14	47.89	79.61	70.53	51.88	38.25	69.58	21.61	15.70
w/o Noise Sampling	88.04	**85.43**	55.42	48.65	79.77	71.51	54.25	39.66	71.79	21.31	15.66

results demonstrate the effectiveness of our multi-task learning framework for closed-book generation tasks.

In the RAG setting, **our CorpusLM models also take the lead, with CorpusLM (Llama2) demonstrating the most substantial improvements.** Among small language models, our T5-based CorpusLM outperforms RAG by 12.52%, 47.98%, and 37.37% on the AY2, HoPo, and ELI5 datasets respectively. It even exceeds MT-DPR+FID, which utilizes a stronger fusion-in-decoder generator, by 9.17%, 12.21%, and 40.36% on these datasets. Among LLMs, our Llama2-based CorpusLM surpasses E5+Llama2-13B by 36.08%, 129.64%, and 37.92% on the FEVER, zsRE, and WoW datasets respectively. It even outperforms E5+Llama2-70B by 80.80%, 107.99%, and 28.51% on these datasets. The above results validate the effectiveness of our multi-task learning framework and continuous RAG strategy.

5.3 Ablation Studies

In this section, we conduct experiments on ablations for both retrieval and downstream tasks, to assess the effectiveness of each proposed component for T5 and Llama2-based CorpusLM.

5.3.1 Ablations for Retrieval Tasks. For retrieval tasks, we investigate the impact of the DocID understanding tasks and the DocID list decoding strategy. The results are shown in Table 4.

(1) Removing DocID understanding tasks reduces retrieval performance across all datasets for both T5 and LLama2-based CorpusLM. Specifically, we see notable decreases on the NQ, HoPo, and ELI5 datasets by 5.37%, 5.53%, and 18.42% for CorpusLM (T5), respectively. For CorpusLM (Llama2), we also see notable declines on the WnWi and ELI5 datasets, with drops of 4.70% and

7.40%, respectively. This underlines the crucial role of DocID understanding in enhancing CorpusLM's retrieval capability. **(2) Changing the ranking-oriented decoding to the traditional beam decoding leads to lower R-Precision in most datasets**. Notably, for CorpusLM (T5), TQA and ELI5 saw reductions of 3.84% and 7.74% in R-Precision, highlighting the importance of this strategy for retrieval. For CorpusLM (Llama2), we also observe a consistent performance drop without the DocID understanding tasks,

5.3.2 Ablations for Downstream Tasks. We further examine the effectiveness of the proposed reference decoding, noise sampling, and continuous decoding. The results are detailed in Table 5.

(1) Removing reference decoding, there's a performance decline across all tasks. This is particularly pronounced on NQ and HoPo datasets, with a decrease of 2.92% and 4.54%, respectively. For CorpusLM (Llama2), removing reference decoding notably affects the accuracy in the Open Domain QA domain, especially for TQA and ELI5, with reductions of 2.85% and 1.85%, respectively. This indicates that reference decoding plays a critical role in filtering out irrelevant information and generating more accurate answers. **(2) Without noise sampling, we also observe a reduction in most datasets**, with decreases of 0.61% and 1.44% on AY2 and on zsRE for T5-based CorpusLM, respectively. This suggests that introducing noise during training helps improve RAG performance. **(3) The removal of continuous decoding leads to drops in most datasets for CorpusLM (T5)**, with a decline of 2.14% on WnWi and 1.12% on TQA, showcasing the importance of a streamlined decoding process.
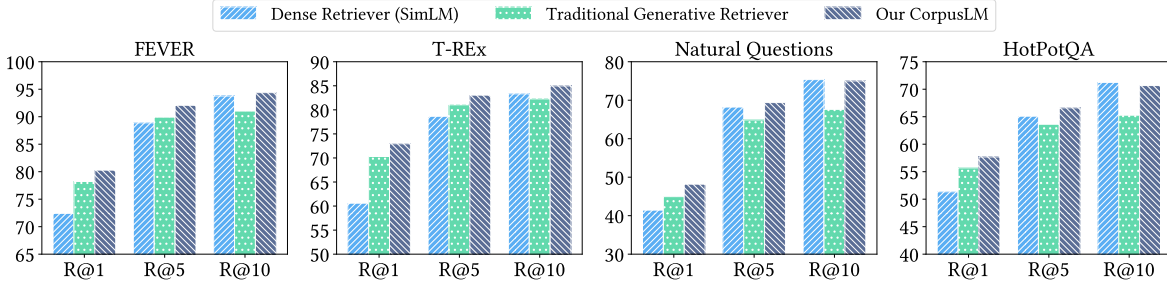
Figure 3: Analysis of ranking capability of CorpusLM for retrieval tasks. We compare our T5-based CorpusLM with dense retriever SimLM and another T5-based generative retriever with traditional beam search, focusing on Recall@{1, 5, 10}.

Table 6: Analysis of the efficiency of RAG models.

Method	Parameters	Storage	Latency
RAG	626M	59.3G	106.7ms
MT-DPR+FID	440M	51.2G	160.9ms
CorpusLM (T5)	**220M**	**426.1M**	**78.4ms**
w/ Pipeline Decode	220M	426.1M	102.9ms

5.4 Analysis of Ranking Capability

To evaluate the effectiveness of the proposed ranking-oriented DocID decoding strategy, we conduct a series of experiments to compare the dense retriever (DR) SimLM [52], generative retriever (GR), and our T5-based CorpusLM, with a focus on the Recall metric. Detailed comparisons are shown in Figure 3, where the generative retriever utilizes our CorpusLM model with the traditional constrained beam decoding method.

For the four datasets in the figure, we observe that Dense Retriever is consistently outperformed by Generative Retriever for Recall@1, but it surpasses GR at Recall@10. Notably, on the FEVER and NQ datasets, DR lags behind GR by 7.41% and 7.81% at Recall@1, yet it leads by 3.01% and 11.44% at Recall@10, respectively. This trend suggests that traditional GR methods are firmly focused on the top-1 ranked DocID, often at the expense of accuracy for other rop-ranked DocIDs. **Our CorpusLM, however, demonstrates superior performance to GR at all levels of Recall**. With improvements of 3.68% and -10.94% in Recall@10 for FEVER and NQ, our model showcases an advanced capability for ranking enhancement. Moreover, CorpusLM consistently outpaces the DR at Recall@5 and is competitive at Recall@10. These results underline the strength of the ranking strategy employed by CorpusLM, leading to improved ranking performance.

5.5 Efficiency Analysis for RAG Tasks

We then analyze the efficiency of our continuous RAG strategy, evaluating model parameters, memory footprint and query latency, as shown in Table 6. Our CorpusLM (T5) showcases a significant reduction in the number of parameters, memory usage, and query latency compared to other methods such as RAG and MT-DPR+FID. Specifically, the CorpusLM model has approximately **2.8-fold fewer parameters** than the RAG [27] model, resulting in a substantially lower computational cost. Notably, it implies a **139.1-fold decrease in memory footprint**, primarily due to the

model only requiring storage for titles and sections as DocIDs rather than a large-scale dense document index. Moreover, compared with MT-DPR+FID [34], the CorpusLM model exhibits a **2.1-fold reduction in query latency**. The table also highlights that even when utilizing a pipeline decode approach, our model maintains its efficiency advantage, with a 1.3-fold faster decoding speed compared to the multi-round input-output RAG method. Our CorpusLM model utilizes continuous decoding to combine generative retrieval and answer generation into a single step, which can accelerate response times and cut down on the computational costs of repeated interaction cycles in traditional RAG approaches.

6 CONCLUSION AND FUTURE WORKS

In this work, we propose CorpusLM, a unified language model that handles various knowledge-intensive tasks by integrating generative retrieval, closed-book generation, and RAG. By unifying all tasks and incorporating auxiliary DocID understanding tasks, we facilitate joint learning of retrieval and generation, thus enhance the model's performance in KI tasks. This work opens up the potential for unifying various IR tasks (especially the retrieval task that cannot be easily performed by LLMs) into a single generative language model. Furthermore, our model is enhanced with two well-designed generation strategies: (1) a ranking-oriented DocID list generation strategy, which refines GR by directly learning from a DocID ranking list, to improve retrieval quality. (2) a continuous DocIDs-References-Answer generation strategy, which facilitates effective and efficient RAG. These generation strategies enable the model to perform retrieval and RAG in a continuous auto-regressive decoding process. Experiments conducted on 11 datasets in the KILT benchmark have shown our CorpusLM's superior performance in both retrieval and downstream tasks.

REFERENCES

[1] Michele Bevilacqua, Giuseppe Ottaviano, Patrick S. H. Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive Search Engines: Generating Substrings as Document Identifiers. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). http://papers.nips.cc/paper_files/paper/2022/hash/cd88d62a2063fdaf7ce6f9068fb15dcd-Abstract-Conference.html

[2] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving Language Models by Retrieving from Trillions of Tokens. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 2206–2240. https://proceedings.mlr.press/v162/borgeaud22a.html

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[4] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=5k8F6UU39V

[5] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, 1870–1879. https://doi.org/10.18653/V1/P17-1171

[6] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2023. A Unified Generative Retriever for Knowledge-Intensive Language Tasks via Prompt Learning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1448–1457. https://doi.org/10.1145/3539618.3591631

[7] Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2022. CorpusBrain: Pre-train a Generative Retrieval Model for Knowledge-Intensive Language Tasks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, Mohammad Al Hasan and Li Xiong (Eds.). ACM, 191–200. https://doi.org/10.1145/3511808.3557271

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2023. PaLM: Scaling Language Modeling with Pathways. *J. Mach. Learn. Res.* 24 (2023), 240:1–240:113. http://jmlr.org/papers/v24/22-1144.html

[9] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. *CoRR* abs/2210.11416 (2022). https://doi.org/10.48550/ARXIV.2210.11416 arXiv:2210.11416

[10] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. *CoRR* abs/2305.14314 (2023). https://doi.org/10.48550/ARXIV.2305.14314 arXiv:2305.14314

[11] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of Wikipedia: Knowledge-Powered Conversational Agents. In *International Conference on Learning Representations*.

[12] Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *LREC*.

[13] Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long Form Question Answering. In *ACL*. Association for Computational Linguistics, Florence, Italy, 3558–3567. https://doi.org/10.18653/v1/P19-1346

[14] Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web* 9, 4 (2018), 459–479.

[15] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*. PMLR, 3929–3938.

[16] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *EMNLP*. 782–792.

[17] Yasuto Hoshi, Daisuke Miyashita, Youyang Ng, Kento Tatsuno, Yasuhiro Morioka, Osamu Torii, and Jun Deguchi. 2023. RaLLe: A Framework for Developing and Evaluating Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023 - System Demonstrations, Singapore, December 6-10, 2023*, Yansong Feng and Els Lefever (Eds.). Association for Computational Linguistics, 52–69. https://aclanthology.org/2023.emnlp-demo.4

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

[19] Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot Learning with Retrieval Augmented Language Models. *J. Mach. Learn. Res.* 24 (2023), 251:1–251:43. http://jmlr.org/papers/v24/23-0037.html

[20] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *Comput. Surveys* 55, 12 (2023), 1–38.

[21] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *ACL*. Association for Computational Linguistics, Vancouver, Canada, 1601–1611.

[22] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*. 6769–6781.

[23] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 39–48. https://doi.org/10.1145/3397271.3401075

[24] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[25] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *CoNLL*. 333–342.

[26] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL*. 7871–7880.

[27] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[28] Xiaoxi Li, Yujia Zhou, and Zhicheng Dou. 2023. UniGen: A Unified Generative Framework for Retrieval and Question Answering with Large Language Models. *arXiv preprint arXiv:2312.11036* (2023).

[29] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Learning to Rank in Generative Retrieval. *CoRR* abs/2306.15222 (2023). https://doi.org/10.48550/ARXIV.2306.15222 arXiv:2306.15222

[30] Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview Identifiers Enhanced Generative Retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, 6636–6648. https://doi.org/10.18653/V1/2023.ACL-LONG.366

[31] Jiongnan Liu, Jiajie Jin, Zihan Wang, Jiehan Cheng, Zhicheng Dou, and Ji-Rong Wen. 2023. RETA-LLM: A Retrieval-Augmented Large Language Model Toolkit. *arXiv preprint arXiv:2306.05212* (2023).

[32] Xiao Liu, Hanyu Lai, Hao Yu, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. WebGLM: Towards An Efficient Web-Enhanced Question Answering System with Human Preferences. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, Ambuj K. Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Ozcan, and Jieping Ye (Eds.). ACM, 4549–4560. https://doi.org/10.1145/3580305.3599931

[33] Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 2021. Multi-Task Retrieval for Knowledge-Intensive Tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, 1098–1111. https://doi.org/10.18653/V1/2021.ACL-LONG.89

[34] Jean Maillard, Vladimir Karpukhin, Fabio Petroni, Wen-tau Yih, Barlas Oguz, Veselin Stoyanov, and Gargi Ghosh. 2021. Multi-Task Retrieval for Knowledge-Intensive Tasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, 1098–1111. https://doi.org/10.18653/V1/2021.ACL-LONG.89

[35] Donald Metzler, Yi Tay, Dara Bahri, and Marc Najork. 2021. Rethinking search: making domain experts out of dilettantes. In *ACM SIGIR Forum*, Vol. 55. ACM New York, NY, USA, 1–27.

[36] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. WebGPT: Browser-assisted question-answering with human feedback. *CoRR* abs/2112.09332 (2021). arXiv:2112.09332 https://arxiv.org/abs/2112.09332

[37] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a Benchmark for Knowledge Intensive Language Tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 2523–2544. https://doi.org/10.18653/v1/2021.naacl-main.200

[38] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[39] 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. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67. http://jmlr.org/papers/v21/20-074.html

[40] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–16.

[41] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3505–3506.

[42] Stephen E. Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (2009), 333–389. https://doi.org/10.1561/1500000019

[43] Devendra Singh Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-End Training of Neural Retrievers for Open-Domain Question Answering. (2021), 6648–6662. https://doi.org/10.18653/V1/2021.ACL-LONG.519

[44] Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems* 34 (2021), 25968–25981.

[45] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. 2023. Learning to Tokenize for Generative Retrieval. *CoRR* abs/2304.04171 (2023). https://doi.org/10.48550/ARXIV.2304.04171 arXiv:2304.04171

[46] Yubao Tang, Ruqing Zhang, Jiafeng Guo, Jiangui Chen, Zuowei Zhu, Shuaiqiang Wang, Dawei Yin, and Xueqi Cheng. 2023. Semantic-Enhanced Differentiable Search Index Inspired by Learning Strategies. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, Ambuj K. Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Ozcan, and Jieping Ye (Eds.). ACM, 4904–4913. https://doi.org/10.1145/3580305.3599903

[47] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. 2022. Transformer Memory as a Differentiable Search Index. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). http://papers.nips.cc/paper_files/paper/2022/hash/892840a6123b5ec99ebaab8be1530fba-Abstract-Conference.html

[48] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 809–819.

[49] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[50] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288* (2023).

[51] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text Embeddings by Weakly-Supervised Contrastive Pre-training. *CoRR* abs/2212.03533 (2022). https://doi.org/10.48550/ARXIV.2212.03533 arXiv:2212.03533

[52] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2023. SimLM: Pre-training with Representation Bottleneck for Dense Passage Retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, 2244–2258. https://doi.org/10.18653/V1/2023.ACL-LONG.125

[53] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Hao Sun, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Allen Sun, Weiwei Deng, Qi Zhang, and Mao Yang. 2022. A Neural Corpus Indexer for Document Retrieval. *CoRR* abs/2206.02743 (2022).

[54] 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, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, Ingo Frommholz, Frank Hopfgartner, Mark Lee, Michael Oakes, Mounia Lalmas, Min Zhang, and Rodrygo L. T. Santos (Eds.). ACM, 2656–2665. https://doi.org/10.1145/3583780.3614993

[55] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).

[56] 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 *ICLR*.

[57] Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. 2023. Auto Search Indexer for End-to-End Document Retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, 6955–6970. https://aclanthology.org/2023.findings-emnlp.464

[58] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *EMNLP*. Association for Computational Linguistics, Brussels, Belgium, 2369–2380. https://doi.org/10.18653/v1/D18-1259

[59] Hailin Zhang, Yujing Wang, Qi Chen, Ruiheng Chang, Ting Zhang, Ziming Miao, Yingyan Hou, Yang Ding, Xupeng Miao, Haonan Wang, et al. 2023. Model-enhanced Vector Index. *arXiv preprint arXiv:2309.13335* (2023).

[60] Peitian Zhang, Zheng Liu, Yujia Zhou, Zhicheng Dou, and Zhao Cao. 2023. Term-Sets Can Be Strong Document Identifiers For Auto-Regressive Search Engines. *arXiv preprint arXiv:2305.13859* (2023).

[61] Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. 2023. Retrieve Anything To Augment Large Language Models. *CoRR* abs/2310.07554 (2023). https://doi.org/10.48550/ARXIV.2310.07554 arXiv:2310.07554

[62] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *CoRR* abs/2303.18223 (2023). https://doi.org/10.48550/ARXIV.2303.18223 arXiv:2303.18223

[63] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2023. Enhancing Generative Retrieval with Reinforcement Learning from Relevance Feedback. In *Proceedings*

of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, 12481–12490. https://aclanthology.org/2023.emnlp-main.768

[64] 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. *CoRR* abs/2208.09257 (2022).

[65] Yujia Zhou, Jing Yao, Ledell Wu, Zhicheng Dou, and Ji-Rong Wen. 2023. WebUltron: An Ultimate Retriever on Webpages Under the Model-Centric Paradigm.

IEEE Transactions on Knowledge and Data Engineering (2023).

[66] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107* (2023).

[67] Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. 2022. Bridging the Gap Between Indexing and Retrieval for Differentiable Search Index with Query Generation. *CoRR* abs/2206.10128 (2022). https://doi.org/10.48550/ARXIV.2206.10128 arXiv:2206.10128