WebUltron: An Ultimate Retriever on Webpages Under the Model-Centric Paradigm

Yujia Zhou, Jing Yao, Ledell Wu, Zhicheng Dou, *Member, IEEE*, and Ji-Rong Wen, *Senior Member, IEEE*

***Abstract*—Document retrieval has been extensively studied within the *index-retrieve* framework for decades, which has withstood the test of time. However, this approach inherently segregates the indexing and retrieval processes, preventing a cohesive, end-to-end optimization. To bridge this divide, we introduce WebUltron, a revolutionary model-centric indexer for document retrieval. This system embeds the entirety of document knowledge within the model, striving for seamless end-to-end retrieval. Two primary challenges with this indexer are the representation of document identifiers (docids) and the model's training. Current methods grapple with docids that lack semantic depth and the constraints of limited supervised data, making scaling up to larger datasets challenging. Addressing this, we've engineered two novel docid types imbued with richer semantics that also streamline model inference. Further enhancing WebUltron's capabilities, we've developed a three-stage training regimen, leveraging deeper corpus insights and fortifying query-docid relationships. Experiments on two public datasets demonstrate the superiority of WebUltron over advanced baselines for document retrieval.**

***Index Terms*—Document retrieval, generative model, model-based IR.**

I. Introduction

Turning to search engines to address daily information needs has become a common behavior. In response to a given query, search engines typically employ the established information retrieval (IR) pipeline, specifically the *index-retrieve-rank* strategies [1], [2], to generate a ranked list of documents. Over the past several decades, the inverted index [1] has been foundational to term-based or sparse retrieval methods. With the advent of pre-trained language models (PLMs) [3], [4], [5], [6], sophisticated representation learning approaches [7], [8], [9], [10], [11] have been employed. These techniques are adept at capturing the intricate semantics of both queries and documents, producing superior representation vectors. Such advancements have notably enhanced the search quality within the *index-retrieve-rank* framework.

Both sparse and dense retrieval models have traditionally been studied within the *index-retrieve* framework, which has proven invaluable for document retrieval. However, this pipeline-based approach necessitates a vast pre-computed index encompassing the entire corpus to facilitate subsequent document retrieval. Such a requirement not only imposes significant memory overheads but also constrains the optimization of the distinct indexing and retrieval stages in an end-to-end manner. To overcome these challenges, several recent studies [12], [13], [14], [15], [16] have made preliminary attempts to develop an end-to-end retrieval model. Such models directly yield relevant document identifiers (docids) and supplant the traditional explicit index with a large-scale model, referred to as the differentiable neural search index [12]. This paradigm shift allows for end-to-end document retrieval by leveraging a sequence-to-sequence (seq-to-seq) generative model.

Despite notable advancements in generative retrieval models, two primary challenges persistently undermine their effectiveness in retrieving relevant documents: (1) how to represent docids so that the model can learn the semantics of documents and retrieve the correct docids more easily; (2) how to train the model so as to capture the semantic knowledge of each docid and to learn the mapping relations from queries to relevant docids. Given these challenges, there's a pressing need for a comprehensive solution that enhances both the representation of docids and the training approaches used for these models.

In the context of treating document retrieval as a generative task, the representation of docids poses a significant challenge. Some early studies [12], [14], [15] experimented with various innovative approaches for representing docids, such as atomic identifiers and semantic cluster identifiers. However, the scalability of these docid representations to larger corpora remains an unexplored issue, largely due to limitations in model parameter size and representational capacity. To address this gap, we propose representing each docid as a sequence of shared tokens, embedding richer semantic information into these sequences to enhance the model's generalizability. Specifically, we introduce two types of semantically-rich identifiers. The first, termed *Keyword-based identifiers*, utilizes a sequence of keywords to identify documents. In this approach, the URL

Manuscript received 28 December 2022; revised 20 September 2023; accepted 11 November 2023. Date of publication 15 November 2023; date of current version 7 August 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 61872370 and 61832017, in part by Beijing Outstanding Young Scientist Program under Grant BJJWZYJH012019100020098, in part by Beijing Academy of Artificial Intelligence(BAAI), in part by the Outstanding Innovative Talents Cultivation Funded Programs 2021 of Renmin University of China, and in part by the Intelligent Social Governance Platform, Major Innovation and Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. Recommended for acceptance by Y. Tong. *(Corresponding author: Zhicheng Dou.)*

Yujia Zhou, Jing Yao, Zhicheng Dou, and Ji-Rong Wen are with the Gaoling School of Artificial Intelligence, Beijing Key Laboratory of Big Data Management and Analysis Methods, DEKE, Renmin University of China, Beijing 100872, China (e-mail: zhouyujia@ruc.edu.cn; jing_yao@ruc.edu.cn; dou@ruc.edu.cn; jirong.wen@gmail.com).

Ledell Wu is with the Beijing Academy of Artificial Intelligence, Beijing 100084, China (e-mail: wuyu@baai.ac.cn).

Digital Object Identifier 10.1109/TKDE.2023.3332858

and title of a webpage serve as natural keywords that maintain both the uniqueness and semantic value of the identifier. The second type, *Semantic-based identifiers*, represents a document through a series of latent topic tokens. Drawing inspiration from product quantization (PQ) technologies [17], [18], [19] in IR, we consider the PQ code of a document as a form of semantic-based identifier.

In this paper, we introduce WebUltron, an **ult**imate **r**etriever **on web**pages under the model-centric paradigm. WebUltron is built upon generative language models that utilize a transformer-based encoder-decoder architecture. Specifically, we conceptualize document retrieval as a sequence-to-sequence task: the model receives a query as input and outputs a docid. Previous studies [15], [16] reveal that merely relying on limited supervised click data is insufficient for equipping the model with adequate knowledge about each docid. To address this shortcoming, we have developed a three-stage training framework to optimize the WebUltron model. (1) *General Pre-training*. This initial stage aims to align docids and terms within a unified semantic space. To accomplish this, we employ multiple pre-training tasks that bridge the semantic gap between these two elements. (2) *Search-oriented Pre-training*. To improve model performance on search tasks, we focus on enhancing its ability to map short, query-like texts to relevant docids. In this context, we generate pseudo-queries to train the model, thereby adapting it to real-world search scenarios. (3) *Supervised Fine-tuning*. The final stage involves fine-tuning the model using supervised relevance data, enabling it to establish more robust associations between queries and docids. During inference, given a query, our model is capable of directly generating a ranked list of docids through constrained beam search.

To evaluate the performance of our model, we conduct thorough experiments using the widely-accepted MS MARCO and NQ document retrieval datasets. The experimental results validate the effectiveness of our proposed model, including two types of semantic-enhanced docid and the three-stage training pipeline. Furthermore, our detailed analysis of memory usage and computational efficiency substantiates the practical feasibility of our method.

The contributions of this work can be summarized as follows: (1) Along with the blueprint for model-based IR, our primary contribution resides in framing generative retrieval as an integration of two critical components: the representation of docids and the enhancement of training data. (2) We introduce two distinct methods for representing docids, offering greater scalability to larger corpora compared to existing approaches. (3) We develop a three-stage training workflow designed to encode specific knowledge of each docid into the model. This aims to bridge the semantic gap between queries and docids, thereby improving performance in document retrieval tasks.

II. Related Work

A. Index-Based Document Retrieval

In the prevailing *index-retrieve-rank* pipeline, sparse and dense retrieval stand out as the two primary methods for document retrieval.

Sparse Retrieval: Owing to their efficiency and effectiveness, sparse retrieval methods, which largely rely on inverted indexes, are widely used in practice. The classic BM25 model [1] leverages the frequency-based signal tf-idf to weigh terms and calculate matching scores between queries and documents. Additionally, graph-based approaches [20], [21] construct document graphs and employ a PageRank-like mechanism to derive term weights. With the emergence of representation learning [22], [23], a strand of research [24], [25], [26] has emerged that learns term weights automatically from word embeddings with rich semantic and co-occurrence information. However, sparse retrieval faces the challenge of mismatch between query and document words. To tackle this, dense retrieval methods have been employed to overcome the limitations of word mismatching.

Dense Retrieval: These methods leverage deep learning to capture the semantic similarity between queries and documents, thereby overcoming the limitations associated with mere lexical overlap [27]. Typically, these methodologies first apply a neural network to embed all queries and documents into low-dimensional vectors. Then, they calculate the vector similarity between queries and documents to retrieve relevant documents, where ANNS algorithm and PQ [18] are used to achieve a more efficient vector search process. A widely-used framework for dense retrieval is the dual encoder [7], [9], [28]. With advancements in PLMs [3], [4], higher-quality representation vectors are obtained, leading to improved outcomes. In order to further enhance performance, various strategies for hard negative sampling have been proposed to optimize retrieval [27], [29], [30], [31]. Recognizing that retrieval performance can be constrained by the dot product of individual vectors, a line of research introduces lightweight interaction layers to capture more fine-grained matching relationships, such as the multi-vector encoding model [32] and ColBERT [33].

In this paper, we move away from traditional indexes and explore a model-centric paradigm that directly outputs documents on an end-to-end basis.

B. Generative Models for Information Retrieval (IR)

Recently, applying generative models to IR tasks has attracted increasing attention. In earlier stages,seq-to-seq models are used for text generation to assist IR tasks. Ahmad et al. [34] introduced a generative component for query suggestion. Nogueira et al. [35], [36] apply a seq-to-seq model to predict possible queries as an expansion of the corresponding document. Subsequently, generative models were trained to directly produce answers tailored to specific tasks. For instance, Nogueira et al. [37] fine-tuned the T5 model [4] to generate relevance labels for candidate documents. GENRE [38] retrieved relevant entities by generating their names. However, a notable limitation of these generative models is their lack of knowledge about document identifiers. Addressing this, a framework for model-based IR was introduced in [39], which embeds docids within the model. Building upon these innovations, Tay et al. [12] devised the DSI model for retrieval tasks on a small-scale corpus. Inspired by this, Zhuang et al. [16] introduced a query generation module for

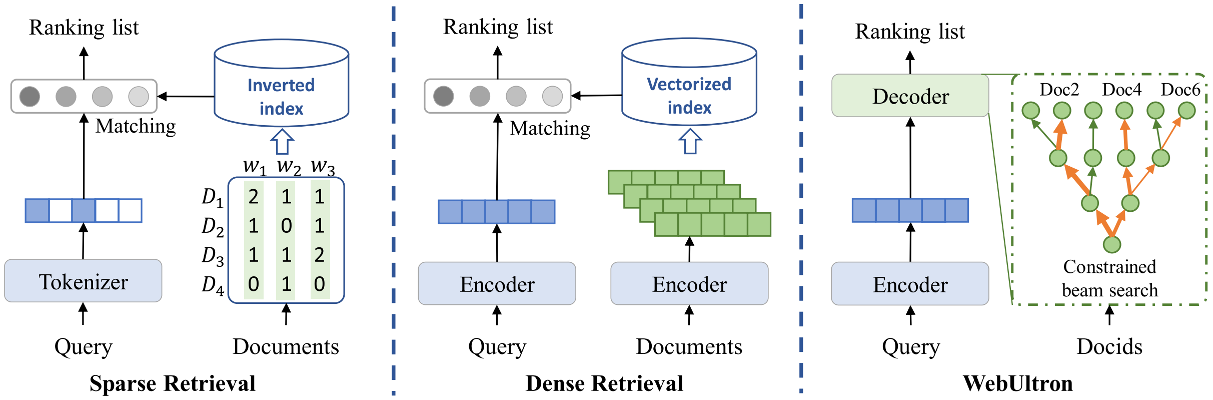


Fig. 1. Comparison between Sparse Retrieval, Dense Retrieval, and WebUltron. Traditional methods follow the indexing-matching workflow, while the WebUltron solely utilizes a unified generative model for document retrieval. The docids (represented as strings) are generated from the query with a seq-to-seq model, and the ranking list is formulated based on the constrained beam search.

data augmentation. Wang et al. [15] designed the neural corpus indexer to further enhance model performance. Taking this a step further, Bevilacqua et al. [13] and Chen et al. [40] extended this paradigm to knowledge-grounded retrieval tasks, achieving better results. Yet, despite these advancements, challenges remain, primarily stemming from semantically deficient docids and limited supervised data. In this paper, our goal is to delve deeper into the model-centric approach, striving to devise an even more efficient retriever.

III. WebUltron: An Ultimate Retriever on Webpages

The traditional *index-retrieve-rank* framework has been a mainstay in IR for decades. This method involves encoding documents into either term-based indexes or dense vector-based indexes and then traversing the index to evaluate the relevance between a provided query and its corresponding candidate documents. However, as outlined in Section I, index-based retrieval methodologies often face optimization challenges due to their inherently pipelined workflows. Inspired by state-of-the-art generative PLMs such as GPT-3 [5] and T5 [4], we propose WebUltron, an ultimate retriever on webpages under the model-centric paradigm that completes document retrieval tasks in an end-to-end generative manner. During the training stage, the model progressively learns the knowledge of all documents, and generates the document ranking list directly for a given query in the inference stage.

A. Backbone of the Model

In alignment with the framework of model-based IR, we attempt to address the document retrieval problem in a generative manner through a seq-to-seq model. As shown in Fig. 1, WebUltron is implemented within an encoder-decoder framework, which encodes the input query and decodes relevant docids using constrained beam search to formulate a ranking list directly. In contrast to traditional sparse and dense retrieval methods, WebUltron transforms the matching task into a generation task, deviating from the conventional indexing-matching paradigm. This transition not only eliminates the need for traditional indexes but also allows for end-to-end optimization of the model.

Sequence-to-Sequence Model. Given the efficacy of seq-to-seq structure across various generation tasks, we leverage the T5 [4] pre-trained language model as our backbone. This model incorporates a Transformer-based [41] encoder-decoder structure. In WebUltron, we define the basic task as a “text-to-docid” format, which means the model receives a textual input and is tasked with generating a relevant docid (represented as a sequence of tokens). To be consistent with the information modeled by the dual encoder, we add a mean-pooling layer after the encoder to represent the query with a single vector q. Based on q, the decoder module tries to predict the relevant docid with the highest auto-regressive score, denoted as:

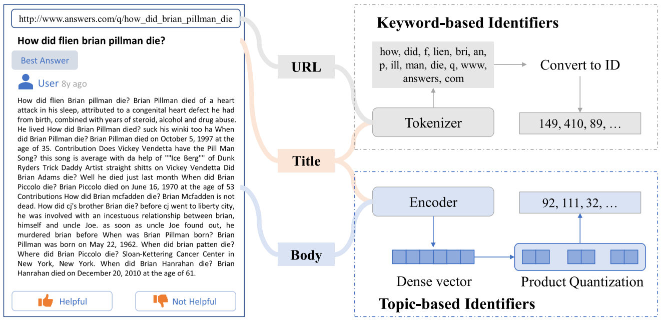
$$\text{score}(d|q) = p_\theta(y|q) = \prod_{i=1}^{N} p_\theta(y_i|y_{<i}, q), \qquad (1)$$

where y is the string identifier of the document d with N tokens, and θ is the parameters of the model. Formally, the workflow of the WebUltron model can be defined as:

$$y = \text{Decoder}\left(\text{Pooling}\left(\text{Encoder}(q)\right)\right). \qquad (2)$$

Contrary to other seq-to-seq tasks like machine translation or dialog systems, the document retrieval task requires the model to generate valid docids within the corpus. Free-form generation might result in an output string that does not match any docids. We address this specific challenge in the subsequent section.

Constrained Beam Search: Beam search is a commonly used decoding algorithm that enhances the capabilities of greedy search by expanding the search space, thereby facilitating the discovery of globally optimal solutions. However, in our specific use case, where it is imperative to generate docids that already exist within the corpus, conventional beam search falls short. Motivated by [38],we employ a constrained beam search. This approach directs the decoder to navigate within a confined token space at each step, thereby generating valid docids from a predefined candidate set. Specifically, we define the constraint by constructing a prefix tree built from all docid strings. Each



http://www.answers.com/q/how_did_brian_pillman_die

How did flien brian pillman die?

Best Answer

User 8y ago

How did flien Brian pillman die? Brian Pillman died of a heart attack in his sleep, attributed to a congenital heart defect he had from birth, combined with years of steroid, alcohol and drug abuse. He lived How did Brian Pillman died? suck his winki too ha When did Brian Pillman die? Brian Pillman died on October 5, 1997 at the age of 35. Contribution Does Vickey Vendetta have the Pill Man Song? this song is average with da help of ""Ice Berg"" of Dunk Ryders Trick Daddy Artist straight shitts on Vickey Vendetta Did Brian Adams die? Well he died just last month When did Brian Piccolo die? Brian Piccolo died on June 16, 1970 at the age of 53 Contributions How did Brian mcfadden die? Brian Mcfadden is not dead. How did cj's brother Brian die? before cj went to liberty city, he was involved with an incestuous relationship between brian, himself and uncle Joe. as soon as uncle Joe found out, he murdered brian before When was Brian Pillman born? Brian Pillman was born on May 22, 1962. When did brian patten die? Where did Brian Piccolo die? Sloan-Kettering Cancer Center in New York, New York. When did Brian Hanrahan die? Brian Hanrahan died on December 20, 2010 at the age of 61.

Fig. 2. Two ways of representing document identifiers as strings. We devise keyword-based and semantic-based identifiers to involve document semantics from different perspectives.

node in this tree has child nodes that contain all valid subsequent tokens for a given prefix sequence. By decoding along the prefix tree, the model ensures that the generated docids are valid and exist within the corpus. Finally, the model yields the top-k docids as the ranking results based on their auto-regressive scores during the beam search.

B. Design of Document Identifiers

A natural attribute of document identifiers (docid) is to distinguish different documents. Intuitively, previous works have tried to identify documents with a random integer, called atomic identifiers [12], [14]. However, they lead to gigantic embedding parameters and lack semantics. To alleviate this problem, we represent each docid as a sequence of shareable tokens satisfying two characteristics: uniquely referring to a document and reflecting the semantic information of the document. Following the ideas of sparse retrieval and dense retrieval, as shown in Fig. 2, we attempt to represent docids from two perspectives: keyword-based identifiers and semantic-based identifiers.

Keyword-Based Identifiers: Using keywords to represent the document content is a hallmark of sparse retrieval. Inspired by this, we aim to uniquely identify a document using meaningful keywords. Interestingly, we find that the URL of a web page naturally has such abilities. For example, the URL "*http://www.answers.com/q/how_did_brian_pillman_die*" reflects that the main content of this page is related to the answer of "how did brian pillman die". This observation inspires us to *generate the document's URL directly for a given query*. To streamline the model's prediction process, we rearrange each segment of the URL (delimited by '/') in reverse sequence. This ensures the prediction starts with the semantically-rich portion, followed by the domain name. However, not all URLs provide sufficient semantic information. To navigate this, we incorporate the document title as keyword-based information of a webpage. Specifically, we integrate the URL and the title of the webpage together to represent docids, defined as:

$$\mathrm{docid}_{\mathrm{URL}} = \begin{cases} \text{title} + \text{domain}, & \text{if title length} > L, \\ \text{reverse URL}, & \text{otherwise.} \end{cases}$$

Here L is set to 2 in our experiments. Finally, we can get a sequence of tokens by T5 tokenizer to represent the docid.

Semantic-Based Identifiers: Dense retrieval maps documents into a latent semantic space using dense vectors. In extreme cases, each dense vector can be used as a unique identifier to distinguish documents. However, the space of dense vectors is too large to decode. This promotes us to look for a way to preserve dense vector semantics in a smaller topic space. As a classic vector compression method, Product Quantization [17], [18], [19] just meets our needs for designing docids. For all D-dimensional vectors, it first divides the D-dimensional space into m groups, and then performs K-means clustering on each group to obtain k cluster centers. Finally, each vector can be represented as a set of m cluster ids. Similarly, for the document d, its semantic-based identifier can be defined as:

$$\mathrm{docid}_{\mathrm{PQ}} = \mathrm{PQ}\left(\mathrm{Encoder}(d)\right), \tag{3}$$

where $\mathrm{Encoder}(\cdot)$ is implemented by a pre-trained T5 encoder. For cluster ids of all groups, we regard them as $m \times k$ new tokens and add them into the vocabulary. However, a disadvantage of the PQ code is that it may not uniquely refer to a document. Thus, for repeated PQ codes, we add an incremental number after the PQ code to ensure the uniqueness of docids.

IV. Three-Stage Training Workflow

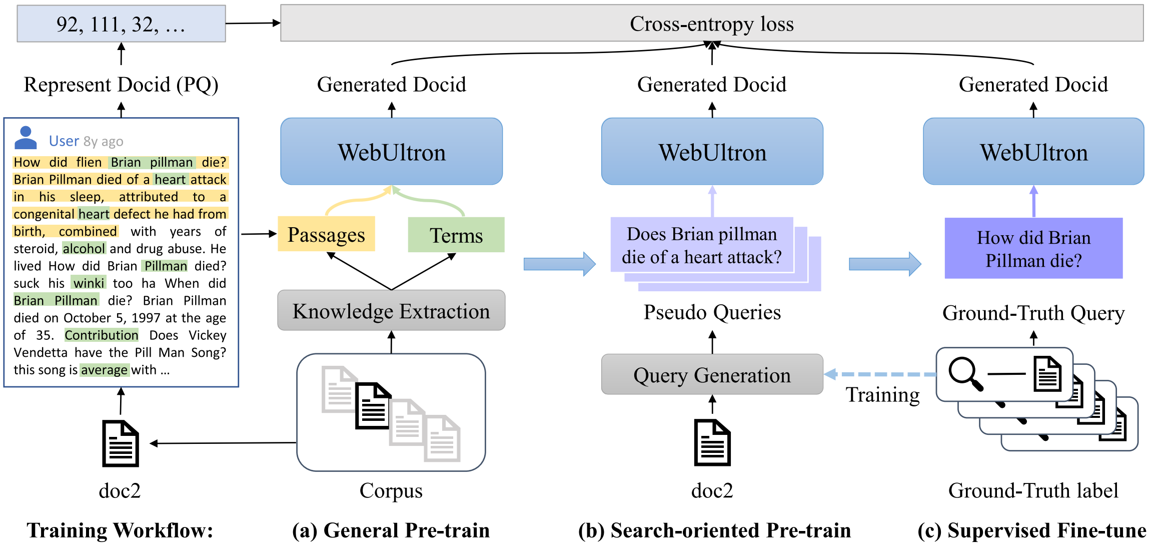
As we discuss in Section I, the training phase of WebUltron can be likened to the indexing stage in classic IR systems. Through this process, we expect the model to encode rich semantics over docids and learn the mapping relations from queries to relevant docids. However, insufficient supervised click data makes it hard for the model to learn associations between queries and docids. This realization motivates us to construct more self-supervised training data to adapt the model to search scenarios.

As shown in Fig. 3, the complete training process is divided into three stages. The first stage is general pre-training, designed to learn the general semantics of docids and to establish relationships between texts and docids. The second stage, search-oriented pre-training, focuses on generating pseudo queries to enhance the model performance on search tasks. The final stage involves supervised fine-tuning, which is applied to further improve model's capabilities for document retrieval using supervised data. The details of the three-stage training are introduced in the following sections.

A. General Pre-Training

The semantic information contained in the document is a basic knowledge of the docid, which is useful in general IR tasks. To learn such knowledge, we conduct general pre-training by extracting self-supervised signals from the corpus. While existing PLMs have captured semantic dependencies through term-to-term relations, the introduction of docids brings forth two novel types of relationships that merit consideration during the model's pre-training phase.

Terms-Docid: Establishing a connection between the term and docid spaces is crucial for the model to learn the knowledge of each docid. Specifically, multiple term sequences can be extracted from the document content to construct the mapping



User 8y ago

How did flien Brian pillman die? Brian Pillman died of a heart attack in his sleep, attributed to a congenital heart defect he had from birth, combined with years of steroid, alcohol and drug abuse. He lived How did Brian Pillman died? suck his winki too ha When did Brian Pillman die? Brian Pillman died on October 5, 1997 at the age of 35. Contribution Does Vickey Vendetta have the Pill Man Song? this song is average with ...

Fig. 3. Three-stage training workflow of the model. First, general pre-training is undertaken to acquaint the model with the basic knowledge of each docid. Then, search-oriented pre-training is conducted to adapt the model to search scenarios. Finally, the model is fine-tuned using supervised data to learn relationships between queries and docids.

relations from text to docid. There are two simple but effective strategies to achieve this.

First, inspired by previous studies that use passage-level evidence for document ranking [42], we segment the document text into passages with fixed-size windows, constructing *passage-to-docid* samples for model training. Formally, given a document containing n terms, i.e., $\{t_1, t_2, \ldots, t_n\}$, we can extract multiple training pairs using windows of size s, such as:

$$\text{passage} : \{t_i, t_{i+1}, \ldots, t_{i+s}\} \longrightarrow \text{docid}, \tag{4}$$

where i is any starting position and is set at intervals of s.

Second, the importance of each word within a document for its semantic representation is different. Recognizing this, we attempt to highlight some important words to reflect the basic semantics of the document. TF-IDF [1] weight is a typical indicator to measure the term importance, which can be used to generate training pairs in the form of *terms-to-docid*. Based on term weights, we select several important terms as a set to reflect the document semantics. We have:

$$\text{terms} : \{t_i, \ldots, t_j, \ldots, t_k\} \longrightarrow \text{docid}, \tag{5}$$

where t_i, t_j, t_k are important terms selected from the document.

Docid-Docid: In our design, we utilize two methods to represent the docid: keyword-based and semantic-based identifiers. Each method offers a unique lens through which to understand the semantics of a document. This differentiation prompts us to explore the relationships between the PQ code and the URL of a document, enabling them to mutually enhance each other. Concretely, for the model with semantic-based identifiers, we can extract knowledge from keyword-based identifiers, i.e., *URL-to-PQ*. In reverse, semantic-based identifiers can also provide information for the model with keyword-based identifiers, i.e., *PQ-to-URL*. Our hypothesis is that allowing these two identifiers to predict one another could bolster the model's reasoning capabilities.

B. Search-Oriented Pre-Training

After general pre-training, the model already attains a basic understanding of the semantics of each docid. However, our observations indicate that this base knowledge isn't sufficient for excelling at document retrieval tasks. Specifically, beyond merely comprehending the semantic knowledge within documents, the model needs to further learn the interrelations between queries and docids. To adapt the model to search scenarios, we further conduct search-oriented pre-training. This step entails the generation of pseudo queries derived from the corpus and the establishment of mapping relations from these pseudo queries to docids.

Following [36], we first train a query generation model over supervised data based on a T5 backbone. Then, for a document containing a series of terms $\{t_1, t_2, \ldots, t_n\}$, the query generation model outputs k predicted queries, i.e., $Q = \{q_1, \ldots, q_k\}$. Finally, by training over *pseudo query-to-docid* samples, our model implements the adaptation from general tasks to search tasks. Formally, the training pairs are formed as:

$$\text{pseudo query} : q_i \longrightarrow \text{docid}, i \in \{1, \ldots, k\} \tag{6}$$

Different from the training samples used in the general pre-training stage, the pseudo queries have two distinct features related to search tasks. First, the average length of the generated queries is much shorter, aligning with the typical queries posed by users. Second, the pseudo queries often take the form of a question, commonly starting with phrases like "how about," "what is," and so forth. Training the model with such data aids in enhancing its proficiency in mapping query-like strings to docids.

C. Supervised Fine-Tuning

After general pre-training and search-oriented pre-training, our model already possesses a foundational understanding and

TABLE I
STATISTICS OF DIFFERENT DATASETS

Dataset	#Doc	#Train Q	#Dev Q
MS MARCO Relevant 300K	319,927	367,013	808
MS MARCO Random 300K	321,631	36,670	504
NQ Relevant 320K	231,695	307,373	7,830

reasoning capacity over docids. To tailor the model to the specific data distribution of downstream datasets, we further use supervised data to fine-tune the model. Specifically, the supervised data contains query-docid pairs indicating their relevance. By training the model with these *query-to-docid* samples, it becomes comprehensively equipped with the necessary knowledge for the document retrieval task.

Given that all the training tasks are unified under the "text-to-docid" format, we finalize the three-stage training of WebUltron based on the standard seq-to-seq objective, i.e., maximizing the output sequence likelihood with teacher forcing. Concretely, for the input sequence q, the generation objective can be formalized as:

$$\mathcal{L} = \arg\max_{\theta} \sum_{i} \log p_{\theta}(y_i | y_{<i}, q), \tag{7}$$

where $p_{\theta}(y_i|y_{<i}, q)$ is the generation probability of token y_i based on the given input. The parameters are optimized by the cross-entropy loss and the AdamW optimizer [43].

V. Experimental Settings

A. Datasets

We conduct experiments on two datasets commonly used in document retrieval tasks: MS MARCO Document Ranking [44] and NQ (Natural Questions) [45]. The datasets' specifics are presented in Table I.

MS MARCO [44] is a large collection of 367,013 training queries paired with 3.2 million documents. To assess the model's efficacy across varying levels of supervised fine-tuning data, we construct two different subsets, the *Relevant 300 K* and the *Random 300 K*. The Relevant 300 K subset includes documents that each have a corresponding query. In contrast, the Random 300 K subset comprises 10% of candidate documents, randomly sampled from the entire corpus. For all three sets, we use queries whose relevant document is contained in the corresponding set for training and testing.

NQ [45] is a public natural question dataset. Each piece of data contains a real question alongside a corresponding Wikipedia article serving as its answer. We use URL to eliminate duplicate documents within the corpus. The primary objective of the document retrieval task for this dataset is to fetch the specific Wikipedia page associated with the question.

B. Baseline

For comparison, we select three categories of models as baselines.

1) Sparse Retrieval Methods: These methods score candidate documents based on the weight of query terms appearing in each document. *BM25* [1] uses the tf-idf feature to measure term weights. *DocT5Query* [36] expands the document content with possible queries predicted by a fine-tuned T5 [4], which takes the given document as its input.

2) Dense Retrieval Methods: This category emphasizes the dual encoder framework, where both the query and the document are individually embedded into vectors. The inner product of these vectors is then computed to derive a relevance score. Within this category, we examine two distinct implementations, each based on different foundational encoders: *RepBERT* [7] and *Sentence-T5* [28]. For both models, we first train them through in-batch contrastive learning and then retrieve relevant documents on top of faiss [46]. Since we do not use hard negative samples to optimize WebUltron model, those advanced dense retrieval models with hard negative sampling as baselines are not included in this paper.

3) Generative Retrieval Methods: Several generative models have been explored for model-based IR. DSI [12] uses a text-to-text model to map queries to relevant docids. *DSI-Atomic* assigns each document with a random integer as the identifier. *DSI-Semantic* semantically clusters all documents into a decimal tree and uses the paths as their docids. *DynamicRetriver* [14] includes a BERT encoder and a Docid decoder with a trainable vector for each document. It generates relevant docids by mapping the query representation through the decoder. For our studies, we reproduced the OverDense variant. The models *DSI-QG* [16] and *NCI* [15] build upon the DSI model by incorporating a query generation module. *WebUltron-Atomic*, *WebUltron-URL* and *WebUltron-PQ* are three variants of WebUltron with atomic docids, URL docids and PQ docids respectively.

We assess the recall capabilities of models on the recall@10 metric and evaluate the ranking performance based on p@1 and mrr@10.

C. Implementation Details

In our experiments, BERT model corresponds to the pre-trained 'bert-base-uncased' and T5 model uses 't5-base', both sourced from huggingface transformers.[1] For the dense retrieval models, we set the maximum length of input sequences to 512 and the batch size to 48. For WebUltron-URL and WebUltron-PQ, the max length of URL docids is 100, the hyper-parameter of PQ is $m = 24, k = 256$, and the batch size is 128 and 200 respectively. During the three-stage training, we utilize 10 pieces of passage, 1 key term sequence, 10 pseudo-queries and 1 annotated fine-tune query for each document. The max length of each input sequence is set to 128. All models are trained with the AdamW [43] optimizer. The learning rate is 5e-5 for BERT based models, and 1e-3 for T5 based models. All experiments are carried out on NVIDIA-A100(40 GB). The source code for our experiments can be accessed at https://github.com/smallporridge/WebUltron.

[1] https://huggingface.co/bert-base-uncased/tree/main

TABLE II
OVERALL RESULTS

Model	Scalable	Incremental Documents	MS MARCO						Natural Questions		
			Relevant 300K			Random 300K			Relevant 320K		
			p@1	mrr@10	r@10	p@1	mrr@10	r@10	p@1	mrr@10	r@10
Sparse Retrieval											
BM25	✓	Easy	0.1894	0.2924	0.5507	0.4385	0.5421	0.7381	0.1406	0.2360	0.4793
DocT5Query	✓	Easy	0.2327	0.3425	0.6138	0.4821	0.5795	0.7738	0.1907	0.2955	0.5583
Dense Retrieval											
RepBERT	✓	Easy	0.2525	0.3848	0.6918	0.4087	0.5109	0.7281	0.2263	0.3608	0.6876
Sentence-T5	✓	Easy	0.2723	0.4070	0.7240	0.4226	0.5359	0.7500	0.2251	0.3495	0.6512
DPR	✓	Easy	0.2808	0.4140	0.7310	0.4286	0.5416	0.7552	0.2281	0.3535	0.6564
Generative Retrieval											
DSI-Atomic	✗	Hard	0.3247	0.4429	0.6992	0.4504	0.5640	0.7758	0.2023	0.3216	0.6146
DynamicRetriever	✗	Hard	0.2904	0.4253	**0.7859**	0.4413	0.5518	0.7293	0.2263	0.3608	0.6876
WebUltron-Atomic	✗	Hard	**0.3281**	**0.4686**†	0.7413	**0.4881**†	**0.5942**†	**0.7917**†	**0.2543**†	**0.3859**†	**0.6953**†
DSI-Semantic	✓	Hard	0.2574	0.3392	0.5384	0.2501	0.3221	0.4881	0.1323	0.2377	0.4828
DSI-QG	✓	Hard	0.2782	0.3745	0.6026	0.3427	0.4093	0.5679	0.1909	0.3085	0.5837
NCI	✓	Hard	0.2835	0.3893	0.6385	0.3699	0.4723	0.6016	0.2017	0.3390	0.6027
WebUltron-URL	✓	Easy	0.2896*	0.4044	0.6386	0.3849	0.4679	0.6290	0.2309*	0.3652*	0.6705
WebUltron-PQ	✓	Easy	0.3032*	0.4416*	0.7215	0.4663	0.5639	0.7282	0.2276	0.3523	0.6575

"Scalable" and "Incremental Documents" indicate the model's adaptability to a larger corpus and its ability to manage new documents, respectively. The highest scores are highlighted in bold and the best results of scalable models are underlined. "†" and " * " denotes the result is significantly better than all baselines and scalable baselines in t-test with *p*<0:05.

WebUltron's complete training process consists of three steps: docid representation (keyword-based or semantics-based), pre-training data construction (general pre-training and search-oriented pre-training), and the model training (generation task loss function). In terms of computational complexity, generative retrieval differs from index-based retrieval as the computational complexity during model inference is only dependent on the beam size B and docid length L, denoted as $O(B * L)$. This complexity is independent of the corpus's size. Therefore, with a smaller designated beam size and a concise docid length, WebUltron can facilitate swift document retrieval (details in Section VI-C).

VI. Experimental Results

In this section, we conduct extensive experiments to answer the following research questions:

RQ1: How does the generative model WebUltron perform on the document retrieval task compared to index-based methods? In which scenarios is it most apt?

RQ2: How does each stage of the three-stage training workflow contribute to the final retrieval outcomes?

RQ3: Does WebUltron have lower memory overhead and higher inference speed than existing retrieval methods?

RQ4: Is WebUltron feasible on large-scale document corpora, and if so, how does it perform?

A. Overall Performance (RQ1)

The overall results are presented in Table II . From the results, we can infer several key insights to address *RQ1*.

(1) In most scenarios, the generative retrieval models outperform index-based retrieval methods, with paired t-test at $p < 0.05$ level. Among them, WebUltron-Atomic stands out with the best performance on both the MS MARCO and NQ datasets. We postulate that the superior performance of generative models stems from their capacity for end-to-end optimization tailored specifically for the document retrieval task. Notably, on the Random 300 K dataset, most models lag behind DocT5Query, with the exception of WebUltron-Atomic. This disparity could be attributed to the reduced number of document-query pairs available for training these deep models on this dataset, thus limiting their ability. Yet, the search-oriented pre-training stage of WebUltron appears to compensate for this data scarcity. Another notable trend is that generative models demonstrate a more pronounced advantage in the ranking metrics p@1/mrr@10 compared to r@10. This may be attributed to the fact that generative models are tailored for direct docid generation, rather than relying on pairwise comparisons.

(2) Generative retrieval models with atomic docids (including DSI-Atomic, DynamicRetriver and WebUltron-Atomic) outperform those employing semantic docids (WebUltron-PQ/URL). In models with atomic docids, there are vectors individually set for each document to maintain richer semantic knowledge, thus making it easier to distinguish different documents. Furthermore, WebUltron-Atomic learns more information from pseudo queries, surpassing DSI-Atomic and DynamicRetriever in performance. However, their parameters will increase linearly as the number of documents increases, rendering them less practical for large-scale corpora. The semantic URL and PQ docids with strong scalability and generalizability hold the potential to address these challenges.

(3) When comparing the generative models that use semantic docids, WebUltron-URL and WebUltron-PQ yield superior results compared to DSI-Semantic, DSI-QG, and NCI. On the Relevant 300 K dataset, WebUltron-PQ surpasses NCI by 13.4% on mrr@10. On the NQ dataset, WebUltron-URL outperforms NCI by 7.7% on mrr@10. This indicates that our devised docids can embed richer semantic information, thereby enhancing model

TABLE III
ABLATION STUDY OF THE THREE-STAGE TRAINING WORKFLOW

Model	MS MARCO				Natural Questions	
	Relevant 300K		Random 300K		All 320K	
	mrr@10	r@10	mrr@10	r@10	mrr@10	r@10
WebUltron-URL	**0.4044**	**0.6386**	**0.4679**	**0.6290**	**0.3652**	**0.6705**
w/o General Pretrain	0.3856 (-4.6%)	0.6321 (-1.0%)	0.4396 (-6.0%)	0.5933 (-5.7%)	0.3587 (-1.8%)	0.6608 (-1.4%)
w/o Search-oriented	0.3341 (-17.4%)	0.5211 (-18.4%)	0.2198 (-53.0%)	0.3194 (-49.2%)	0.3071 (-15.9%)	0.6147 (-4.5%)
w/o Fine-tune	0.3477 (-14.0%)	0.5693 (-10.9%)	0.4548 (-2.8%)	0.6083 (-3.3%)	0.3504 (-4.1%)	0.6405(-4.5%)
WebUltron-PQ	**0.4416**	**0.7215**	**0.5639**	**0.7282**	**0.3523**	**0.6575**
w/o General Pretrain	0.4099 (-7.2%)	0.6968 (-3.4%)	0.4984 (-11.6%)	0.6582 (-9.8%)	0.3328 (-5.5%)	0.6327 (-3.8%)
w/o Search-oriented	0.3445 (-22.0%)	0.5730 (-20.6%)	0.3656 (-35.2%)	0.5655 (-22.3%)	0.2427 (-31.1%)	0.5220 (-20.6%)
w/o Fine-tune	0.4176 (-5.4%)	0.7203 (-0.2%)	0.5624 (-0.3%)	0.7262 (-0.3%)	0.3522 (-0.0%)	0.6386 (-2.9%)

The bold entities highlighted the best results among different model variants for each docid.

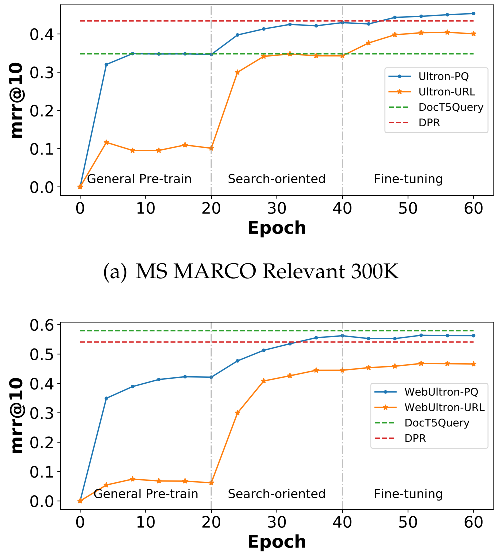
generalizability. On the MS MARCO dataset, WebUltron-URL trails behind WebUltron-PQ. While WebUltron-URL employs URLs to identify documents and primarily captures semantic knowledge linked to these keywords, it might also pick up non-semantic noise present in URLs, such as numbers and symbols. In contrast, PQ docids are derived from the representation of the entire document, making them more closely aligned with the document's content.

In summary, these results indicate that *the end-to-end generative model with semantic document identifiers is a promising approach for document retrieval tasks.*

B. Study of Training Workflow (RQ2)

In this paper, we design a three-stage training process to enhance the WebUltron model. In order to verify the effects of each training stage on the final results (*RQ2*), we conduct an ablation study to remove one training stage at one time and observe its impacts on document retrieval. The results are shown in Table III. We find that the removal of any training stage leads to a decline in performance across all evaluation metrics. Specifically, omitting the search-oriented pre-training results in the most pronounced reduction, particularly evident on the Random 300 K dataset which has fewer supervised document-query pairs. This underscores the pivotal role of pseudo queries in enhancing the model performance on search tasks. Meanwhile, upon eliminating supervised fine-tuning, there is a notable decline in document retrieval performance, confirming the necessity of the supervised stage and its importance in facilitating more potent linkages between queries and docids.

To provide a more detailed understanding of the impact of each training stage, we depict a curve showing mrr@10 in relation to the number of training epochs and stages, as presented in Fig. 4. We can see that with the progression through each stage, the model incrementally acquires knowledge, enhancing its capability for the document retrieval task.



(a) MS MARCO Relevant 300K

(b) MS MARCO Random 300K

Fig. 4. mrr@10 with different training epochs and stages.

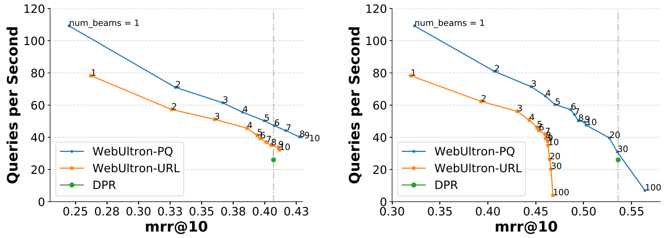
TABLE IV
EXPERIMENTS ABOUT THE MEMORY, MODEL PARAMETERS, AND QUERY LATENCY OF DIFFERENT MODELS

Model	Corpus	Memory	Params	Latency
Brute-force Dual	300K	0.98GB	220M	38.57ms
	3.2M	9.87GB	220M	489.25ms
WebUltron-Atomic	300K	0	495M	20.31ms
	3.2M	0	2718M	-
WebUltron-URL	300K	0.05GB	248M	13.75ms
	3.2M	0.41GB	248M	15.70ms
WebUltron-PQ	300K	0.07GB	257M	8.90ms
	3.2M	0.62GB	257M	9.41ms

C. Study of Memory and Efficiency (RQ3)

Given that document retrieval is an essential component in practical search applications, it is imperative to focus on minimizing memory overhead and maximizing efficiency. To this end, we carry out experiments to compare the memory cost, parameter count and inference latency between our WebUltron model and all baseline models across corpora of varying sizes. The results are displayed in Table IV.

Upon examining Table IV, it's evident that WebUltron, especially WebUltron-URL and WebUltron-PQ, offers a marked decrease in memory usage, parameter count, and inference latency when compared to the brute-force dual encoder. Specifically, WebUltron spends 90% less memory than the dual encoder. With regard to the parameter count, the dual encoder, WebUltron-URL and WebUltron-PQ, mainly rely on a pre-trained language model



(a) MS MARCO Relevant 300K (b) MS MARCO Random 300K

Fig. 5. Query latency and mrr@10 with different beams.

TABLE V
RESULTS ON THE LARGE-SCALE MS MARCO DATASET WITH 3.2 M DOCUMENTS

Model	MS MARCO 3.2M			
	Params	p@1	mrr@10	r@10
DSI-Atomic	×10	-	-	-
DynamicRetriever	×10	-	-	-
DSI-Semantic	×1	0.0481	0.0851	0.1217
DSI-QG	×1	0.0621	0.1067	0.1893
NCI	×1	0.0682	0.1147	0.2023
WebUltron-URL	×1	0.1082	0.1690	0.3172
WebUltron-PQ	×1	0.1246	0.2031	0.3975

Params $\times N$ refers to the ratio of the number of parameters compared to the 300k dataset.

with a fixed number of parameters. Conversely, WebUltron-Atomic employs a trainable vector for each document within the model, which means its parameter scale adjusts as the number of documents grows. Most notably, WebUltron outpaces dual encoders in efficiency, with latency decreasing from 489.25 ms to 15.70 ms. The brute-force dual method involves a traversal of candidate documents, meaning its latency is directly influenced by the size of the corpus. While dual encoder approaches can be sped up using approximate search, this could come at the expense of accuracy. With WebUltron, the model directly generates relevant docids through constrained beam search. Consequently, its speed is related to the layer and width of the prefix tree. The curves in Fig. 5 also demonstrate that WebUltron can achieve a better balance between effectiveness and efficiency.

D. Exploration of Scaling up (RQ4)

Generative models with semantic docids–including DSI-Semantic, DSI-QG, NCI, WebUltron-URL, and WebUltron-PQ–show promise for scalability to expansive document collections. To evaluate their performance in such scenarios, we carried out experiments on the MS MARCO dataset, which consists of 3.2 million documents. The experimental results are summarized in Table V .

Our results show that WebUltron-URL and WebUltron-PQ surpass other baseline models, underscoring the effectiveness of our tailored semantic docids and the three-stage training approach. Nevertheless, regardless of the model employed, there remains a substantial margin for improvement in terms of accuracy. To achieve better results on large-scale corpora, several potential solutions can be explored. One approach is to scale up the model, allowing it to encapsulate more nuanced information and thereby improving its ability to differentiate among various documents. Another potential solution is to enrich the training dataset by assembling a broader and more varied corpus, which would amplify the model's proficiency in recognizing diverse textual nuances. Moreover, as part of our future work, we plan to investigate substituting the t5-base model with larger pre-trained language models to possibly further elevate performance.

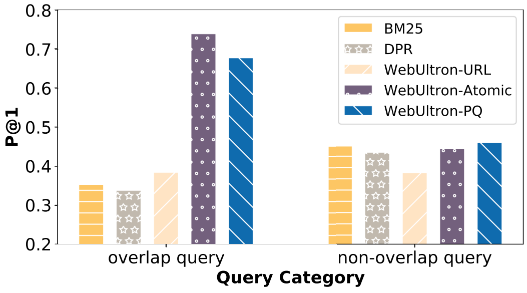


Fig. 6. Model performance on different query subsets.

E. Performance on Different Queries (RQ1)

As listed in Table I, there are approximately 300 k documents paired with relevant queries. Consequently, only a subset of the test queries align with the documents utilized during fine-tuning. Based on this distinction, we categorize all test queries from the Random 300 K dataset into 'overlap query' and 'non-overlap query'. We evaluate the performance of WebUltron and several baselines on these two distinct query sets. The comparison results are shown in Fig. 6.

Our analysis reveals that BM25 and DPR exhibit performance across both query sets. In contrast, the WebUltron model tends to exhibit marked improvements when dealing with overlap queries. Although WebUltron-URL lags slightly behind DPR in performance across the entire query set, it surpasses DPR when evaluated on the overlap query set. As for WebUltron-Atomic and WebUltron-PQ, the former leverages atomic docids to capture the richest document-level feature, and as a result, achieves the highest performance on overlap queries. WebUltron-PQ, on the other hand, strikes a balance between memory capacity and generalizability, thus demonstrating satisfactory results across both subsets.

F. Case Study

To provide an intuitive understanding of how the generative model WebUltron works, we visualized the inference process for WebUltron-URL, which uses URLs as docids, as illustrated in Fig. 7.

First, all URL docids are structured as a prefix tree. Given a query, its contextualized representation vector is obtained through the encoder. This vector is then fed into the decoder, and a constrained beam search is carried out based on the prefix tree. At each step, the token with the highest probability is chosen, corresponding to the darkest node in Fig. 7. The process continues until the URL 'https://en.wikipedia.org/wiki/

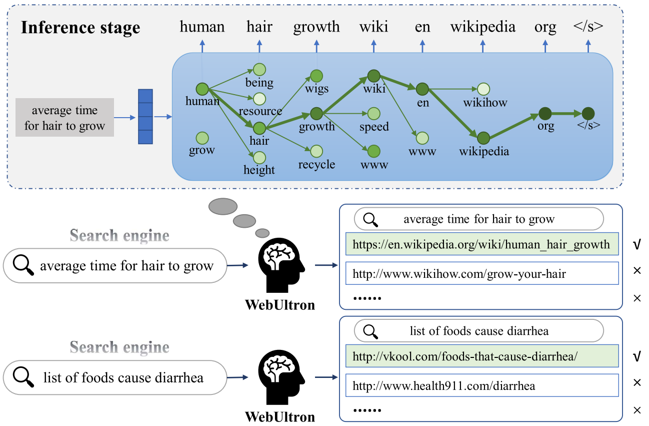

Inference stage
human
hair
growth
wiki
en
wikipedia
org
</s>
average time for hair to grow
human
being
resource
hair
grow
height
wigs
growth
recycle
wiki
speed
www
en
wikihow
www
wikipedia
org
</s>
Search engine
average time for hair to grow
WebUltron
average time for hair to grow
https://en.wikipedia.org/wiki/human_hair_growth
http://www.wikihow.com/grow-your-hair
......
√
×
×
Search engine
list of foods cause diarrhea
WebUltron
list of foods cause diarrhea
http://vkool.com/foods-that-cause-diarrhea/
http://www.health911.com/diarrhea
......
√
×
×

Fig. 7. Case study of the inference of WebUltron-URL.

human_hair_growth' is fully formed upon reaching the leaf node. We can see that these generated URLs contain the annotated relevant document and are indeed semantically relevant with the entered query. This verifies WebUltron's capability to associate semantic knowledge with specific docids.

G. Discussions

As the field increasingly shifts its attention to model-based IR, we would like to share some of our reflections and experiences through the following discussion.

1) What are the Major Challenges in the Implementation of Model-Based IR?: In the implementation of model-based information retrieval, there are significant challenges that need to be addressed. First and foremost is the issue of scalability. The model needs to encode all docid information, which places considerable demands on the design of docids. To address this, incorporating semantics into docids can enhance generalization, thereby reducing the need for a larger model. The lack of sufficient training data is another critical challenge that limits model performance. It has been observed that relying solely on supervised data for model training does not yield satisfactory results. To circumvent this challenge, a three-stage model training framework has been designed to leverage available data effectively.

2) What Valuable Insights Can Serve as Inspiration for Other IR Researchers?: Several valuable insights have emerged from our extensive experiments. First, the generative paradigm exhibits superior ranking performance and higher efficiency in document retrieval tasks compared to brute-force method of searching the entire index. However, it is important to note that this approach may result in lower recall rates, which warrants further exploration. Second, incorporating semantic-rich docid demonstrates high generalizability and scalability, positioning it as viable solution for large-scale scenarios. Lastly, while the atomic method might falter in terms of scalability, it excels in capturing document-level features and holds potential for close-domain scenarios.

3) What are the Limitations of WebUltron?: Despite the achievements made with WebUltron in the model-centric paradigm, several limitations still exist that warrant attention. First, scaling the model to handle web-sized data demands increased model capacity, which presents a challenge when dealing with expansive corpora. The intricate relationship between model capacity and corpus size merits deeper exploration. Second, the incorporation of new incoming documents into the model-based indexer remains unexplored. It's imperative to devise strategies that circumvent the need to retrain the model from scratch whenever new documents are introduced. Future research should focus on addressing these challenges to enhance the capabilities of WebUltron in the field of model-based IR.

VII. Conclusion

In this work, we explore a novel model-centric paradigm for document retrieval. The model WebUltron breaks away from the conventional index-based methods by encoding the knowledge of docids into an end-to-end model. Under the T5 backbone, we devise two types of semantic document identifiers, and a three-stage training strategy to optimize the model and adapt it to search scenarios. Experiments on two public datasets indicate the superiority of our model-based indexer on retrieval performance and efficiency over existing baselines.

References

[1] S. E. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Found. Trends Inf. Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[2] B. Mitra and N. Craswell, "An introduction to neural information retrieval," *Found. Trends Inf. Retrieval*, vol. 13, no. 1, pp. 1–126, 2018.

[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[4] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020.

[5] T. B. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst. 33: Annu. Conf. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[6] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv: 1907.11692*.

[7] J. Zhan, J. Mao, Y. Liu, M. Zhang, and S. Ma, "Repbert: Contextualized text embeddings for first-stage retrieval," 2020, *arXiv: 2006.15498*.

[8] J. Ni et al., "Large dual encoders are generalizable retrievers," 2021, *arXiv:2112.07899*.

[9] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 6769–6781.

[10] K. Lee, M. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," in *Proc. 57th Conf. Assoc. Comput. Linguistics*, 2019, pp. 6086–6096. [Online]. Available: https://doi.org/10.18653/v1/p19--1612

[11] S. Kuzi, M. Zhang, C. Li, M. Bendersky, and M. Najork, "Leveraging semantic and lexical matching to improve the recall of document retrieval systems: A hybrid approach," 2020, *arXiv: 2010.01195*.

[12] Y. Tay et al., "Transformer memory as a differentiable search index," 2022, *arXiv:2202.06991*.

[13] M. Bevilacqua, G. Ottaviano, P. S. H. Lewis, S. Yih, S. Riedel, and F. Petroni, "Autoregressive search engines: Generating substrings as document identifiers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 31668–31683.

[14] Y. Zhou, J. Yao, Z. Dou, L. Wu, and J. Wen, "DynamicRetriever: A pre-training model-based IR system with neither sparse nor dense index," 2022, *arXiv:2203.00537*.

[15] Y. Wang et al., "A neural corpus indexer for document retrieval," 2022, *arXiv:2206.02743*.

[16] S. Zhuang et al., "Bridging the gap between indexing and retrieval for differentiable search index with query generation," 2022, *arXiv:2206.10128*.

[17] T. Ge, K. He, Q. Ke, and J. Sun, “Optimized product quantization,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 744–755, Apr. 2014.

[18] H. Jégou, M. Douze, and C. Schmid, “Product quantization for nearest neighbor search,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, Jan. 2011.

[19] J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, and S. Ma, “Jointly optimizing query encoder and product quantization to improve retrieval performance,” in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 2487–2496.

[20] R. Blanco and C. Lioma, “Graph-based term weighting for information retrieval,” *Inf. Retrieval*, vol. 15, no. 1, pp. 54–92, 2012.

[21] F. Rousseau and M. Vazirgiannis, “Graph-of-word and TW-IDF: New approach to ad hoc IR,” in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 59–68.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *Proc. 1st Int. Conf. Learn. Representations*, Scottsdale, Arizona, USA, 2013, pp. 1–12.

[23] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.

[24] G. Zheng and J. Callan, “Learning to reweight terms with distributed representations,” in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, Santiago, Chile, Aug. 9–13, 2015, pp. 575–584.

[25] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, “A deep relevance matching model for ad-hoc retrieval,” in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 55–64.

[26] M. Dehghani, H. Zamani, A. Severyn, J. Kamps, and W. B. Croft, “Neural ranking models with weak supervision,” in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 65–74.

[27] L. Gao, Z. Dai, T. Chen, Z. Fan, B. V. Durme, and J. Callan, “Complement lexical retrieval model with semantic residual embeddings,” in *Proc. Eur. Conf. Inf. Retrieval*, 2021, pp. 146–160.

[28] J. Ni et al., “Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models,” 2021, *arXiv:2108.08877*.

[29] J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, and S. Ma, “Optimizing dense retrieval model training with hard negatives,” in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 1503–1512.

[30] L. Xiong et al., “Approximate nearest neighbor negative contrastive learning for dense text retrieval,” in *Proc. 9th Int. Conf. Learn. Representations*, 2021, pp. 1–16.

[31] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, “REALM: Retrieval-augmented language model pre-training,” 2002, *arXiv: 2002.08909*.

[32] Y. Luan, J. Eisenstein, K. Toutanova, and M. Collins, “Sparse, dense, and attentional representations for text retrieval,” *Trans. Assoc. Comput. Linguistics*, vol. 9, pp. 329–345, 2021.

[33] O. Khattab and M. Zaharia, “ColBERT: Efficient and effective passage search via contextualized late interaction over BERT,” in *Proc. 43 rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 39–48.

[34] W. U. Ahmad, K. Chang, and H. Wang, “Context attentive document ranking and query suggestion,” in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 385–394.

[35] R. Nogueira, W. Yang, J. Lin, and K. Cho, “Document expansion by query prediction,” 1904, *arXiv: 1904.08375*.

[36] R. Nogueira, J. Lin, and A. I. Epistemic, “From doc2query to docTTTTTquery,” Tech. Rep., vol. 6, 2019.

[37] R. Nogueira, Z. Jiang, R. Pradeep, and J. Lin, “Document ranking with a pretrained sequence-to-sequence model,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 708–718.

[38] N. D. Cao, G. Izacard, S. Riedel, and F. Petroni, “Autoregressive entity retrieval,” in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–20.

[39] D. Metzler, Y. Tay, D. Bahri, and M. Najork, “Rethinking search: Making domain experts out of dilettantes,” *SIGIR Forum*, vol. 55, no. 1, pp. 13:1–13:27, 2021.

[40] J. Chen, R. Zhang, J. Guo, Y. Fan, and X. Cheng, “GERE: Generative evidence retrieval for fact verification,” 2022, *arXiv:2204.05511*.

[41] A. Vaswani et al., “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst. 30: Annu. Conf. Neural Inf. Process. Syst.*2017, pp. 5998–6008.

[42] J. P. Callan, “Passage-level evidence in document retrieval,” in *Proc. 17th Annu. Int. ACM-SIGIR Conf. Res. Develop. Inf. Retrieval*, 1994, pp. 302–310.

[43] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *Proc. 7th Int. Conf. Learn. Representations*, 2019, pp. 1–18.

[44] T. Nguyen et al., “MS MARCO: A human generated machine reading comprehension dataset,” in *Proc. Workshop Cogn. Comput.: Integrating Neural Symbolic Approaches 30th Annu. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1–10.

[45] T. Kwiatkowski et al., “Natural questions: A benchmark for question answering research,” *Trans. Assoc. Comput. Linguistics*, vol. 7, pp. 452–466, 2019.

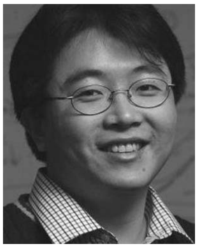
[46] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with GPUs,” *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021.

Yujia Zhou received the BE degree in computer science and technology from School of Information, Renmin University of China, in 2019. He is currently working toward the PhD degree with the School of Information, Renmin University of China. He won the best student paper award in CCIR 2018. His research interests include information retrieval, personalized search, deep learning and data mining.

Jing Yao received the BE degree in computer science and technology from School of Information, Renmin University of China, in 2019, and the MS degree in computer application technology from School of Information, Renmin University of China, in 2022. She has been invited as a reviewer of international conferences SIGIR, WSDM. She is working with Microsoft Research Asia as a research now. Her research interests include information retrieval, personalized search, explainable search/recommendation.

Ledell Wu received the BS and MS degree from Peking University and University of Toronto, respectively. She is currently a research scientist manager with the Beijing Academy of Artificial Intelligence (BAAI). She worked as a research engineer with Facebook AI Research from 2013-2021. She worked on a couple of research projects that also have boarder impact with Facebook, including general purpose embedding system, large-scale graph embedding system and dense passage retrieval system.

Zhicheng Dou (Member, IEEE) received the BS and PhD degrees in computer science and technology from Nankai University, in 2003 and 2008, respectively. He is an associate professor in the School of Information, Renmin University of China. He worked with Microsoft Research as a researcher from 2008 to 2014. His research interests include information retrieval, data mining, and Big Data analytics.



Ji-Rong Wen (Senior Member, IEEE) received the BS and MS degrees from the Renmin University of China, and the PhD degree from the Chinese Academy of Science, in 1999. He is a professor with the Renmin University of China. He was a senior researcher and research manager with Microsoft Research from 2000 to 2014. His main research interests include web data management, information retrieval (especially web IR), and data mining.