CPS-MEBR: Click Feedback-Aware Web Page Summarization for Multi-Embedding-Based Retrieval

Wenbiao Li∗
School of Software & Microelectronics, Peking University
Beijing, China
liwb@stu.pku.edu.cn

Pan Tang∗
Baidu Inc.
Beijing, China
tangpan@baidu.com

Zhengfan Wu
Baidu Inc.
Beijing, China
wuzhengfan@baidu.com

Weixue Lu
Baidu Inc.
Beijing, China
luweixue@baidu.com

Minghua Zhang
Baidu Inc.
Beijing, China
zhangminghua@baidu.com

Zhenlei Tian
Baidu Inc.
Beijing, China
tianzhenlei@baidu.com

Daiting Shi
Baidu Inc.
Beijing, China
shidaiting01@baidu.com

Yu Sun
Baidu Inc.
Beijing, China
sunyu02@baidu.com

Simiu Gu
Baidu Inc.
Beijing, China
gusimiu@baidu.com

Dawei Yin
Baidu Inc.
Beijing, China
yindawei@acm.org

ABSTRACT
Embedding-based retrieval (EBR) is a technique to use embeddings to represent query and document, and then convert the retrieval problem into a nearest neighbor search problem in the embedding space. Some previous works have mainly focused on representing the web page with a single embedding, but in real web search scenarios, it is difficult to represent all the information of a long and complex structured web page as a single embedding. To address this issue, we design a click feedback-aware web page summarization for multi-embedding-based retrieval (CPS-MEBR) framework which is able to generate multiple embeddings for web pages to match different potential queries. Specifically, we use the click data of users in search logs to train a summary model to extract those sentences in web pages that are frequently clicked by users, which are more likely to answer those potential queries. Meanwhile, we introduce sentence-level semantic interaction to design a multi-embedding-based retrieval (MEBR) model, which can generate multiple embeddings to deal with different potential queries by using frequently clicked sentences in web pages. Offline experiments show that it can perform high quality candidate retrieval compared to single-embedding-based retrieval (SEBR) model.

CCS CONCEPTS
• Information systems → Retrieval models and ranking;

KEYWORDS
Multi-Embedding-Based Retrieval; Web Search

ACM Reference Format:
Wenbiao Li, Pan Tang, Zhengfan Wu, Weixue Lu, Minghua Zhang, Zhenlei Tian, Daiting Shi, Yu Sun, Simiu Gu, and Dawei Yin. 2018. CPS-MEBR: Click Feedback-Aware Web Page Summarization for Multi-Embedding-Based Retrieval. In Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION
Search engines (e.g., Google, Baidu, and Bing) are retrieval technologies that retrieve relevant content and feed back to users according to user needs. The first-stage retrieval aims to quickly return a small number of candidate documents from a large-scale corpus. Embedding-based retrieval (EBR) has been shown to be an effective complement to previous sparse retrieval based on term matches and inverted index [11, 36]. It has become the mainstream trend to build EBR system in industry(e.g., Facebook [9], Amazon [23], eBay [32], Baidu [18], Taobao [15], and JD [39]). Currently, building an EBR system mainly faces the following challenges:

• Computational Complexity. Generally, the title is a brief summarization of the page content. As shown in Figure 1, compared
to the title, the page text is much longer, almost two orders of magnitude. Transformer [30] has achieved state-of-the-art results on a wide range of natural language processing (NLP) tasks and has become the most mainstream encoder. However, the computational cost of its core module multi-head self-attention mechanism is proportional to the square of the sequence length, and it is not feasible to directly apply it to long page text in billion-scale industrial scenarios.

- **Redundancy and Noise.** The relevance of a document depends only on that part of the entire web page that matches the search query. As shown in Figure 2, the web page perfectly matches the user’s query of "who proposed the fp5-fp7 plan". However, only a small part of the main content is actually matched, most of which are irrelevant to the query and rarely refer to the user’s search interests. If we calculate the relevance between the query and the title, or the relevance between the query and the content of the entire web page, we may come to the wrong conclusion of a mismatch. Likewise, web pages usually contain many information blocks. In addition to the main content block, it usually has a navigation bar, copyright, privacy notice, sidebar, etc. The information contained in these noise blocks can seriously impair the accuracy of relevance modeling [25, 33, 37].

- **Representation Difficulty.** Most existing industrial works [9, 18] assign only one embedding to the document. Unlike short and topic-focused queries, documents are usually long and can match different potential query needs of users. It is inappropriate to represent the document containing different semantic information as a single embedding. In Figure 2, some users may be interested in the question "Who proposed the fp5-fp7 plan", while some users may be interested in "When will the Horizon plan start?", the answer is located in another sentence of the Wikipedia page.

Academia and industry are committed to addressing these challenges. In order to reduce the redundancy and noise of web page information, previous works have explored to identify and extract the main content of the web page based on web structures [26, 33] or feature fusion by deep networks [40]. However, previous works have mainly focused on noise elimination or restricted to specific websites, few works have discussed how to reduce information redundancy and design a more efficient extraction model for general web pages. To enable Transformer to cope with long sequences, various methods [1, 13, 28, 31, 38] have been proposed to reduce the complexity of full attention. However, when the sequence is shorter than the input limit (512 tokens), these methods suffer performance loss compared to full attention [31]. To cope with the single-embedding problem, some previous works [10, 12, 14, 20] show that multi-embedding representations provide better effectiveness. However, it is computationally complex to directly apply the token-level post-interaction strategy [10, 12] to generate multi-embedding representations for long texts, and directly applying phrase-level fragments to generate multi-embedding representations [14] will introduce a lot of content that is irrelevant to the potential search query. There is still a lack of research on how to develop and deploy multi-embedding-based retrieval (MEBR) systems for general industrial network search.

In this paper, to address the above existing problems, we design a Click feedback-aware web Page Summarization for Multi-Embedding-based Retrieval (CPS-MEBR) framework. Considering that web content is long and complex, and most of it is irrelevant to the potential search intent, we design a summarization model Pageformer to extract some sentences that may match potential queries. Specifically, we comprehensively consider the semantic and structural information of sentences, and supervised training of the Pageformer model using user click data from search logs.
Meanwhile, in order to reduce the complexity of web content modeling, we first score the sentences extracted by Pageformer, and we only select some sentences with high scores to model the MEBR model. To enable different sentences to match different potential queries, instead of performing token-level interactions between sentences on the input side, we use post-interaction with sentence-level semantics, which neither makes representations of different sentences similar nor loses global semantic information.

The main contributions of this paper are:

1. **Pageformer.** Based on the semantic and structural information of sentences, we design a summarization model to extract those sentences that are likely to match the potential query.
2. **Click feedback-aware pre-training task.** We supervise the training of the Pageformer model with this task to further improve the quality of the extracted sentences.
3. **MEBR model.** We use those high-scoring sentences that are likely to match the potential query to build the MEBR model, and introduce sentence-level semantic post-interaction to enrich the multi-embedding representation.
4. **System Design.** Our CPS-MEBR framework can be deployed into production as a complement to existing EBR models.

## 1 RELATED WORK

### 1.1 Extractive Summarization

Extractive summarization systems create a summary by identifying (and subsequently concatenating) the most important sentences in a document. Most models consider extractive summarization as a sentence classification task: creating sentence representations and predicting which sentences should be selected as summaries.

Traditional methods are still strong baselines. Lead-3 simply selects the first three sentences as the summary, which is used as a standard baseline by most recent work [2, 6]. TextRank [21] constructs a weighted graph based on co-similarity between sentences and graph-based ranking models, then sentences with high weights are extracted as summaries.

Representative neural extractive model [22] selects sentences by first encoding the input with hierarchical recurrent neural networks and then sequentially predicting the summaries using a logistic classifier. BERTSum [17] takes BERT [3] as the sentence encoder taking advantages of knowledge from pretraining, and an additional Transformer as the classifier for sentence selection.

But as yet, neither traditional nor neural approach in existing can extract the key content from a long and complex web page very well.

### 1.2 Multi-Embedding-Based Retrieval

EBR is essential for a modern retrieval system. Typical structures of EBR models can be viewed as bi-encoders or Siamese networks [4], which comprises two encoders that conduct semantic modeling. MEBR is developed for full information modeling. Typical MEBR methods can be devided into three levels: token level [12], phrase level [14] and general level [10, 20]. Token level methods compute multiple embeddings after contextualized late interaction over BERT and uses an maximum-similarity function for retrieving relevant documents. Phrase level methods generate embeddings by encoding multiple contiguous text segments of words. Token level or phrase level multi-encoders are still infeasible for billion-scale industrial deployment. General level methods get multiple embeddings through different pooling strategies after encoding, which may not be effective for settings of long input sequences to model different aspects of the input.

### 1.3 EBR in Industry Search

EBR has been regarded as the next generation search technology in industry. For social networks, Facebook developed an EBR system, which took text matching into consideration and achieve hybrid retrieval, for personalize search [9]. For display advertising, Baidu proposed MOBIUS [5] for CPM (Cost Per Mile) maximization in the web ads retrieval phase, reducing the objective distinction between ranking and matching. For recommendation systems in web search, Google [34] adopted transfer learning to learn semantic embeddings to alleviate the cold start problem. For e-commerce search, Amazon [23], Taobao [15] and JD [39] studied the problem of insufficient or low relevance issue in the EBR system and designed more effective encoders with different representation learning techniques. Closest to our scenario in web search, Liu et al. [18] developed and deployed an EBR system equipped with expressive Transformer-based semantic encoders and comprehensive multi-stage training paradigm, whereas only the title of a document with one-single embedding is considered in this work.

Despite the initial success achieved by these works, there is still a lack of investigation on pre-trained page summarization models and more effective document representation methods for real search engine practice.

## 2 CPS-MEBR FRAMEWORK

Our CPS-MEBR framework is mainly divided into two parts: click feedback-aware web page summarization and multi-embedding-based retrieval model.

### 2.1 Click Feedback-Aware Web Page

#### 2.1.1 Pageformer

In this section, we sequentially introduce the model structure of Pageformer and the click feedback-aware pre-training task.

**Pageformer.** Neural network models generally treat extractive summarization as a sentence classification task. An encoder is used to create sentence representations, and a classifier is used to predict which sentences should be selected to form a summarization. Here we design a summarization model Pageformer. The specific structure is shown in Figure 3 and consists of two main parts: (1) In order to solve the problem of semantic modeling of long sequences in web pages, we use a page-level encoder to obtain context-dependent semantic representations; (2) In order to resist the interference of noise content in the web page, we add the structural information of the sentence based on the semantic representation embedding of the sentence. And use a structure-aware sentence classifier to predict the importance of sentences.

**Page-Level Semantic Encoder.** Pretrained language models as a revolutionary technology have been achieving impressive gains in a wide range of NLP tasks. These models extend the idea of
static word embeddings by dynamically learning contextual representations. But most of these Transformer-based methods are limited by the fixed context length in language modeling (up to 512 tokens). For sequences that exceed the length limit, a simple approach is to perform on separated fixed-length short segments without any information flow across them. However, this way the model cannot capture long-term dependency beyond the length of the predefined context. And fixed-length segments are typically created by choosing contiguous chunks of tokens without regard to any other semantic boundaries. The model lacks the necessary contextual information to understand well the boundary markers that lead to performance degradation. This problem is also referred as the well-known context fragmentation.

Unlike other learning tasks, the aim of text summarization is to compress a full document containing thousands of tokens into a shorter version while preserving the main meaning. Therefore, it requires a broad coverage of natural language understanding beyond the meaning of a single word, sentence or even paragraph, and context fragmentation severely compromises the effect.

To address the above bottleneck caused by fixed-length contexts, we construct a page-level encoder to semantically model web page content. Considering that the length of web content is much more than 512 tokens, we use the Reformer [13] model, which is more memory efficient and faster on long sequences, to extract semantic representations of sentences. Specifically, let the web content be $S = (S_1, \ldots, S_N)$, where $S_i$ represents the $i$-th sentence in the web page content, $N$ is the total number of sentences. The input to the model is:

$$[\text{CLS}], S_1, [\text{SEP}], \ldots, [\text{CLS}], S_N, [\text{SEP}]$$

So the semantic representation of the $i$-th sentence is:

$$E_i^s = \text{MeanPooling}(H_{cls}, H_{S_i}, H_{sep}), i = 1, \ldots, N$$

Where $\{H_{cls}, H_{S_i}, H_{sep}\}$ are the encoded hidden representations of sentence $i$ at the corresponding position.

**Structure-Aware Sentence Classifier.** In addition to semantic understanding, many studies have pointed out that the performance of extraction model largely depends on location awareness [8, 29]. For a web page, whether a sentence is important or not is likely to have been revealed by the structure of the web page. Many key attributes of sentences are depicted in HTML tags. `<h1>...6>` and `<b>` indicate the important headline and the specially-emphasized bold text, but the `<footer>` means it can hardly be a candidate key sentence. What’s more, for a rendered web page, users are likely to focus on the content on the top of screen layout that has smaller Y-axis coordinate. For these reasons, it is necessary to model the web page structure. But so far, there is still a lack of formal work related to pretrained language models that characterize the structural properties of web pages.

Therefore, in order to take advantage of the structural information in web pages, we design three structural embedding features for a given sentence in a web page. First, we use a vanilla position embedding as the backbone to indicate sentence identity. Second, the Y-axis coordinate is introduced to represent the cost of consuming the current web content for the user. We map the Y pixel coordinate corresponding to the sentence in the rendered page to a value $y$. Third, HTML tags are introduced to indicate the importance of sentence. $\Omega$ is a set of HTML tags from the node where the current sentence is located to the root node. The final structural embedding of the sentence is:

$$E_i^s = e_i + y_i V + \sum_{\delta \in \Omega} \delta W_\delta$$

where $e_i$ is the vanilla position embedding of the $i$-th sentence, $V$ is a randomly initialized vector, and $W_\delta$ is a randomly initialized embedding lookup table.
The final sentence representation that fuses structural information and semantic information is:

$$E_i = E_i^{se} + E_i^{st}, i = 1, \cdots, N$$

(4)

So the input to the structure-aware sentence classifier is $\{E_1, \cdots, E_N\}$. As shown in Figure 3, we will eventually get the key sentences in the web page.

3.1.2 Click feedback-aware pre-training task. Different from traditional summarization tasks, it is more important to extract scattered sentences that users are more likely to be interested in. To this end, after standard pre-training with a masked language modeling (MLM) task, we further perform a click feedback-aware pre-training task for the end-to-end Pageformer. Specifically, we use large-scale data automatically mined by users’ click behaviors in search logs for training, so the distribution of users’ interest in web pages is implicit in the model.

Query-guided data mining. The key to model pre-training is to build a billion-scale feedback-aware summarization data. In the search log, queries and documents are associated with click signals, i.e. whether the document was clicked by the user. We refer to these queries as click queries. For a given candidate sentence, its corresponding weight is:

$$\omega = \left( \sum_{\theta \in \Theta} \sum_{i=1}^{n_\theta} \rho_\theta n_\theta \right) - \rho_1 l - \rho_2 p - \rho_3 y$$

(5)

where $\Theta$ is the set of all co-occurring entities in the candidate sentence and the click query, $n_\theta$ is the occurrence times of entity $\theta$, $l$ and $p$ are the length and position of the candidate sentence, respectively. $\{\rho_0, \rho_1, \rho_2, \rho_3\}$ is a set of penalty factors. $\rho_1$ mitigates the impact of very long sentences. $\rho_2$ and $\rho_3$ are used to punish users for content that is difficult to consume.

Then, we normalize the weights of the sentences inside the web page. Here we are more concerned about which sentences in the web page are more important and which are not. In order to avoid extracting sentences in meaningless boundary blocks, we first remove sentences with extremely low scores that are irrelevant to the click query, and then pick them in descending order of sentence weights.

Supervised pre-training. Our Pageformer follows the extractive summarization task paradigm, and performs end-to-end pre-training on click feedback-aware data to learn the knowledge of posterior click behavior, and further can implicitly learn the distribution of user attention in web pages. In inference, Pageformer can extract key sentences that users are more likely to be interested in.

3.2 Multi-Embedding-Based Retrieval Model

Considering that it is difficult for a single embedding representation to cover fine-grained semantic information in a document, we use a multi-embedding-based representation to characterize the document. Specifically, we represent all selected key sentences in a document as embeddings and use them as vector indexes for the document.

3.2.1 Preliminaries. Formally, given a query $q \in Q$ and a document $d \in D$, the text pair $(q, d)$ refers to the query and its candidate document. A typical semantic retrieval model adopts a dual-encoder architecture, where the query and document are represented as dense vectors, respectively, and the relevance score $f(q, d)$ between them is measured by the similarity between their embeddings:

$$f(q, d) = \sin(E_Q(q), E_D(d))$$

(6)

where $f$ is a scoring function that measures the similarity between query-document pairs, sim is a similarity metric, e.g., inner-product, cosine similarity, and $E_Q$ and $E_D$ are dense encoders for queries and documents, respectively.

Semantic retrieval models use a conventional contrastive-learning loss to train query and document encoders to distinguish the relevant query-document pair from irrelevant query-document pairs, thereby generating an efficient representation space for queries and documents. Given a query $q$, its corresponding relevant document is $d^+$, and the set of irrelevant documents is $D^-$. The loss function is as follows:

$$L = -\log \frac{e^{f(q, d^+)/\tau}}{\sum_{d \in D^-} e^{f(q, d)/\tau}}$$

(7)

where $\tau$ is the temperature of the softmax operation [7]. Using a higher value for $\tau$ produces a softer probability distribution over classes.

3.2.2 MEBR-TSC Model. Semantic retrieval models usually encode partial information (e.g., title, abstract) of a web document in search into a single embedding, which is called as single-embedding-based retrieval (SEBR) model. When the title or abstract completely contains the whole document information, this method can effectively model the relevance between the query and the document. But this is not always the case. In many cases, the title or abstract cannot cover the entire document information, especially when the content of the web page is relatively long, it often contains content information that can answer multiple queries. In the previous subsection, we obtained key sentences $\{S_1, \cdots, S_k\}$ which have different information in web documents through Pageformer. Let each token of the query be encoded as $[H^{T}_{cls}, H^{S1}_{cls}, \cdots, H^{T}_{cls}]$, and the semantic vectors corresponding to the [CLS] token after the title and key sentences are encoded as $H^{T}_{cls}$ and $[H^{S1}_{cls}, \cdots, H^{S_k}_{cls}]$.

We first calculate the content representation $C$ of the web page:

$$C = \sum_{i=1}^{k} \alpha_i H^{S_i}_{cls}$$

(8)

where $\alpha_1, \cdots, \alpha_k$ = Softmax($v \cdot H^{T}_{cls}, \cdots, v \cdot H^{S_k}_{cls}$). The $v$ is a randomly initialized vector, and learnt during training phase.

Then we use the title and key sentences to build multiple representations. To preserve the semantic information of the context, we introduce sentence-level semantic interactions, inject $C$ into each representation, and control the ratio between them through a weight coefficient $\lambda$:

$$R_0 = \lambda H^{T}_{cls} + (1 - \lambda)C$$

(9)

$$R_i = \lambda H^{S_i}_{cls} + (1 - \lambda)C, i = 1, \cdots, k$$

(10)

where multiple representations of documents are $\{R_0, R_1, \cdots, R_k\}$.

Poly attention. Humeau et al. [10] propose a poly-encoder structure that enables more global features. Liu et al. [19] make
targeted improvements which works slight differently during training and prediction phases. Following on from solid work of Liu et al. [19], we learn a set of \( m \) context codes, i.e., \( c_1, c_2, \ldots, c_m \), where each \( c_i \) extracts a global representation \( g_i \) by attending over all the outputs \( \{ H_{c_{cls}}^q, H_1^q, \ldots, H_n^q \} \) of the query encoder:

\[
g_i = \beta_{c_{i0}}^H H_{c_{cls}}^q + \sum_{j=1}^m \beta_{c_{ij}}^H H_j^q \tag{11}\]

where \( (\beta_{c_{i0}}, \beta_{c_{i1}}, \ldots, \beta_{c_{in}}) = \text{Softmax}(\langle c_i \cdot H_{c_{cls}}^q, c_i \cdot H_1^q, \ldots, c_i \cdot H_n^q \rangle) \), the \( m \) context codes are randomly initialized, and learnt during training phase.

For the calculation of the relevance score, we define the calculation formula:

\[
\text{MaxP}(R_Q: \Omega_Q, R_D: \Omega_D) \tag{12}
\]

Where \( \Omega_Q \) is the query set, \( \Omega_D \) is the document set, each representation in the \( \Omega_Q \) set will calculate a correlation score with each representation in the \( \Omega_D \) set, and finally take the maximum value of all the scores.

The relevance score for the training phase is:

\[
f(q,d) = \text{MaxP}(R_Q: \{g_1, \ldots, g_m\}, R_D: \{R_0, R_1, \ldots, R_k\}) \tag{13}
\]

The relevance score for the inference phase is:

\[
g_{\text{mean}} = \frac{1}{m} \sum_{i=1}^m g_i \tag{14}
\]

\[
f(q,d) = \text{MaxP}(R_Q: \{g_{\text{mean}}\}, R_D: \{R_0, R_1, \ldots, R_k\}) \tag{15}
\]

3.2.3 Negative sampling strategy. In the first-stage retrieval, separating relevant documents from the entire corpus is a big challenge, where the size of the corpus can reach tens of billions or more. We need to sample relevant and irrelevant documents during training phase. For the mining of positive and negative examples, we mainly consider two data sources commonly used in practice:

- Search log data. Queries and documents record a click signal, i.e., whether the user clicked on the document. For each query, we used those clicked results as positive examples and those that were public but not clicked as negative examples, since clicks can roughly represent satisfaction with user intent.

- Manually labelled data. This is collected through a fine-grained level (i.e., 0, 1, 2, 3, 4) assigned by manual evaluation. For each query, positive and negative examples are defined in pairs. For a given document considered positive, other lower level documents under the same query can be considered negative.

The goal of online retrieval is to distinguish between a small number of relevant documents and a large number of irrelevant documents. Inspired by previous work [16, 24, 35], instead of labeling a large number of irrelevant documents, we use **in-batch negative mining**. Specifically, we refer to the non-click (or lower-level) documents under a query as strong negative examples, and the corresponding positive and strong negative examples of other

---

**Figure 4:** Model architecture and relevance computation during training and inference phases. The yellow arrows indicate separate encoding within the batch and obtain the semantic vector corresponding to the [CLS] token.
queries in the same mini-batch are called random negative examples of this query. So each query has $2b - 1$ ($b$ is the size of the mini-batch) negatives.

4 EXPERIMENTAL SETUP

4.1 Datasets

We use the search log data and manually labelled data described with respect to query. \(\bar{\text{D}}_q\) is the set of ground truth and \(\hat{\text{D}}_q\) is the top-k retrieval with respect to query.

We use Recall@5 and Recall@10 on the Manual and Content Hit dataset, which contains an average of 10 relevant (label \(\geq 2\)) documents per query. Similarly, we use Recall@1 and Recall@3 on the Hardcase dataset, with an average of 3 relevant documents (label \(\geq 2\)) per query.

Taking into account the difference between datasets, we use the average of Recall@k to measure the comprehensive performance of the model.

4.3 Model Implementation

We use SEBR-T, SEBR-P, SEBR-TC, SEBR-TP, SEBR-P and SEBR-TP as baseline models.

4.4 Implementation Details

We use the PaddlePaddle framework for experiments. For Pageformer, the training epochs are 10, the batch size is 1, the learning rate is 5e-5, the optimizer is AdamW, the weight decay is 0.01, and the warm-up ratio is 0.1.

For the page-level semantic encoder, we use a 12-layer Reformer [13] structure with a maximum length set to 8192. For the structure-aware sentence classifier, we use a 3-layer Transformer structure with a maximum length set to 256. The penalty factors \(\rho_0\), \(\rho_1\), \(\rho_2\), and \(\rho_3\) in Eq. (5) are set to 0.2, 0.0025, 0.002, and 1.2, respectively.

For MEBR-TSC model, the query and document encoders use a 6-layer Transformer structure, the training epochs are 5, the context codes \(m\) are 16, the weight coefficient \(\lambda\) is 0.8, the batch size is 500, the learning rate is 2e-5 with linear schedule, the optimizer is AdamW, the weight decay is 0.01, and the warm-up ratio is 0.1.

Other unmentioned details are set as the same as vanilla ERNIE model. Additionally, we apply mix-precision and re-compute methods that allow larger batchsize during training with a fixed GPU memory.

5 RESULTS AND ANALYSIS

5.1 Overall Results

Table 1 summarizes main experimental results of various models on three datasets mentioned above. We find that using only title (SEBR-T) can achieve good results. Especially, compared to SEBR-P and SEBR-C, SEBR-T on Manual datasets performs better than others, for title is highly relevant to content on this dataset.

We observe that SEBR-C steadily makes a relative improvement of Avg.R(+1.11%) over SEBR-P. Since the user is only interested in a small part of the information in the web page, most of which are irrelevant to the query. Considering the combination of title and content, SEBR-TP and SEBR-TC make significantly improvement of Avg.R(+2.65%) and Avg.R(+2.35%) compared to SEBR-P and SEBR-C respectively. Likewise, SEBR-TC performs better than SEBR-TP with a gain of 0.81% on Avg.R. These show that Pageformer can better discover key sentences for queries, which is very helpful in the follow-up modeling of relevance.

Our proposed MEBR model (MEBR-TSC) outperforms the optimal SEBR model (SEBR-TC) comprehensively with an average recall improvement of 0.9. Specifically, on the Manual dataset, our
The effect of different number of sentences on the results.

| Model     | Manual | Hardcase | Content Hit |
|-----------|--------|----------|-------------|
|           | Recall@5 | Recall@10 | Recall@1 | Recall@3 | Recall@5 | Recall@10 | Avg.R |
| SEBR-T    | 71.74   | 79.22    | 47.39     | 50.00    | 67.23    | 69.64     | 64.20 |
| SEBR-P    | 68.25   | 74.56    | 48.29     | 49.55    | 68.05    | 69.84     | 63.09 |
| SEBR-TP   | 72.70   | 79.58    | 49.45     | 51.50    | 69.79    | 71.40     | 65.74 |
| SEBR-C    | 70.35   | 76.42    | 48.34     | 49.05    | 68.11    | 70.25     | 64.20 |
| SEBR-TC   | 73.15   | 79.58    | 51.08     | 53.05    | 70.23    | 72.13     | 66.55 |
| MEBR-TC   | 71.42   | 78.66    | 52.13     | 54.90    | 70.04    | 72.39     | 66.59 |
| MEBR-TSC  | 73.72   | 80.11    | 52.66     | 54.82    | 70.63    | 72.74     | 67.45 |

5.2 Analysis of the number of sentences

Table 2: The effect of different number of sentences on the results.

| k | Manual | Hardcase | Content Hit |
|---|--------|----------|-------------|
|   | R@5    | R@10     | R@1 | R@3 | R@5 | R@10 | Avg.R |
| 5 | 73.59   | 80.17    | 51.18 | 54.50 | 70.21 | 71.86 | 66.92 |
| 10| 73.72   | 80.11    | 52.60 | 54.82 | 70.63 | 72.74 | 67.45 |
| 15| 73.72   | 80.41    | 50.29 | 55.06 | 71.23 | 73.06 | 67.30 |
| 20| 73.79   | 80.46    | 48.82 | 55.06 | 70.95 | 73.23 | 67.05 |

Since key sentences are sorted in descending order of scores, the further back the sentences are, the lower the quality is and the overall improvement that can be brought is very limited. For Recall@1 of the Hardcase dataset, since no certain representation has a clear advantage over others, when the number of sentences increases, it introduces more noisy or unimportant information and leads to lower results. As the number of sentences increases, the index volume will increase dramatically. Here, we choose the top 10 sentences with the best recall performance on average.

5.3 Analysis of Weight Coefficient $\lambda$

The $\lambda$ controls the proportion of content information introduced by each representation, which needs to be determined by grid search. When $\lambda$ is large, the introduced content information is insufficient, and the correlation between different representations is weak. When $\lambda$ is small, too much content information is introduced, and the diversity between representations of sentences is reduced. The experimental results show that when $\lambda = 0.8$, the average recall result is optimal.

5.4 Case Study

As shown in Table 4, there are two examples retrieved by MEBR-TSC. In order to highlight the key points, we only list the title of document and the sentence that triggers the retrieval.

Case 1: This is a common example where the query is irrelevant to the title. We are not able to retrieve the correct document without content, for semantic of the title is far from the query. Our MEBR-TSC returns the fourth sentence, which is relevant and explainable. It indicates that our model can dive into the content and search right sentence when the content contains different semantics.

Case 2: Take the second sample as an example where the query is semantically similar to the title and partially match the user’s intent. Our MEBR-TSC find the eighth paragraph whose key term *obese people, loss weight* and *joints* exactly match the intent of user. This shows that MEBR-TSC can capture more fine-grained semantic information and increases the relevance between queries and documents.

6 CONCLUSION

In this paper, we design a click feedback-aware web Page Summarization for multi-embedding-based retrieval (CPS-MEBR) framework. The framework utilizes the page structure information and

Table 1: Experimental results on different datasets.

| Model     | Manual | Hardcase | Content Hit |
|-----------|--------|----------|-------------|
|           | Recall@5 | Recall@10 | Recall@1 | Recall@3 | Recall@5 | Recall@10 | Avg.R |
| SEBR-T    | 71.74   | 79.22    | 47.39     | 50.00    | 67.23    | 69.64     | 64.20 |
| SEBR-P    | 68.25   | 74.56    | 48.29     | 49.55    | 68.05    | 69.84     | 63.09 |
| SEBR-TP   | 72.70   | 79.58    | 49.45     | 51.50    | 69.79    | 71.40     | 65.74 |
| SEBR-C    | 70.35   | 76.42    | 48.34     | 49.05    | 68.11    | 70.25     | 64.20 |
| SEBR-TC   | 73.15   | 79.58    | 51.08     | 53.05    | 70.23    | 72.13     | 66.55 |
| MEBR-TC   | 71.42   | 78.66    | 52.13     | 54.90    | 70.04    | 72.39     | 66.59 |
| MEBR-TSC  | 73.72   | 80.11    | 52.66     | 54.82    | 70.63    | 72.74     | 67.45 |
the click-feedback-aware pre-training task to enhance the extraction effect of the Pageformer on key sentences in web pages. And use the MEBR-TSC model to generate multiple embeddings for the extracted key sentences to model the document. Experimental results show that our method can perform high quality candidate retrieval and better satisfy the user’s search intent.

REFERENCES
[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150 (2020).
[2] Jianpeng Cheng and Mirella Lapata. 2016. Neural Summarization by Extracting Sentences and Words. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 484–494.
[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).
[4] Xingjing Dong and Jianbing Shen. 2018. Triplet loss in siamese network for object tracking. In ECCV. 459–474.
[5] Miao Fan, Jiahe Wang, Guorong Qiao, and Xuedong He. 2019. MOBIUS: towards the next generation of query-ad matching in.baidu’s sponsored search. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Association for Computing Machinery, New York, NY, USA, 4335–3375.
[6] Yiding Liu, Weixue Lu, Linpeng Ding, Shuang Liu, Shuang Liu, and Hao Wen. 2021. Pre-trained Language Model for Web-scale Retrieval in Baidu Search. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Association for Computing Machinery, New York, NY, USA, 3586–3595.
[7] Yiding Liu, Weixue Lu, Linpeng Ding, Shuang Liu, Shuang Liu, and Hao Wen. 2021. Pre-trained Language Model for Web-scale Retrieval in Baidu Search. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Association for Computing Machinery, New York, NY, USA, 3586–3595.
[8] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Thirty-first AAAI conference on artificial intelligence.
[9] Priyanka Nigam, Triveni Song, Vijai Mohan, Vihan Lakshman, Weitian (Allen) Ding, Askti Shangavi, Choon Hui Teo, Hao Gu, and Bing Yin. 2019. Semantic Product Search. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Association for Computing Machinery, New York, NY, USA, 2876–2885.
[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In NeurIPS. 5998–6008.
[11] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized latent interaction over bert. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 39–48.
[12] Nikita Kitasev, Lukasz Kainer, and Anselm Levskaya. 2019. Reformer: The Efficient Transformer. In International Conference on Learning Representations.
[35] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval. 55–64.

[36] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020).

[37] Lan Yi, Bing Liu, and Xiaoli Li. 2003. Eliminating noisy information in web pages for data mining. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. 296–305.

[38] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. Advances in Neural Information Processing Systems 33 (2020), 17283–17297.

[39] Han Zhang, Songlin Wang, Kang Zhang, Zhiling Tang, Yanjun Jiang, Yun Xian, Weipeng Yan, and Wen-Yun Yang. 2020. Towards Personalized and Semantic Retrieval: An End-to-End Solution for E-Commerce Search via Embedding Learning. Association for Computing Machinery, New York, NY, USA, 2407–2416.

[40] Mingzhu Zhang, Zhongguo Yang, Sikandar Ali, and Weilong Ding. 2021. Web Page Information Extraction Service Based on Graph Convolutional Neural Network and Multimodal Data Fusion. In 2021 IEEE International Conference on Web Services (ICWS). IEEE, 681–687.