IR2: Information Regularization for Information Retrieval

Jianyou Wang*   Kaicheng Wang†   Xiaoyue Wang†   Weili Cao,
Ramamohan Paturi†   Leon Bergen‡

Laboratory for Emerging Intelligence

University of California, San Diego

{ jiw101, kaw036, xiw027, w2cao, rpaturi, lbergen }@ucsd.edu

Abstract

Effective information retrieval (IR) in settings with limited training data, particularly for complex queries, remains a challenging task. This paper introduces **IR2**, **I**nformation **R**egularization for **I**nformation **R**etrieval, a technique for reducing overfitting during synthetic data generation. This approach, representing a novel application of regularization techniques in synthetic data creation for IR, is tested on three recent IR tasks characterized by complex queries: DORIS-MAE, ArguAna, and WhatsThatBook. Experimental results indicate that our regularization techniques not only outperform previous synthetic query generation methods on the tasks considered but also reduce cost by up to 50%. Furthermore, this paper categorizes and explores three regularization methods at different stages of the query synthesis pipeline—input, prompt, and output—each offering varying degrees of performance improvement compared to models where no regularization is applied. This provides a systematic approach for optimizing synthetic data generation in data-limited, complex-query IR scenarios. All code, prompts and synthetic data are available at `https://github.com/Info-Regularization/Information-Regularization`.

Keywords: Information Retrieval, Data Augmentation, Synthetic Query Generation, Regularization, Contrastive Learning, Large Language Models

1. Introduction

Users often submit complex queries to information retrieval (IR) systems, expecting answers that accurately address multiple, interconnected questions. These complex queries may involve ambiguous language or multiple specific criteria, requiring that IR systems decipher and respond to the layered intents accurately to provide relevant results (Wang et al., 2023; Lin et al., 2023). Ensuring that IR systems effectively handle the subtleties of such queries is important, allowing these systems to align with varied and nuanced user demands.

Effectively addressing complex queries in IR depends on a model's exposure to a wide range of user queries during training. However, obtaining diverse real-world training data is often constrained by privacy concerns, availability, and resources. Synthetic data, therefore, becomes crucial, offering a means to expand training datasets, enabling models to learn from a broader spectrum of queries and user intents (Ma et al., 2021; Liang et al., 2020).

Recent research exploits Large Language Models (LLMs) to generate synthetic data pairs, constructing synthetic queries from real passages, often derived from zero-shot or few-shot examples (Bonifacio et al., 2022; Jeronymo et al., 2023; Meng et al., 2022; Peng et al., 2023; Penha et al., 2023).

Addressing the challenges of complex query information retrieval (IR) tasks through LLM-based synthetic data generation presents distinct difficulties. While synthetic data improves model performance across multiple tasks and metrics, generating queries from documents often results in synthetic pairs that exhibit superficial textual patterns, such as synonyms, matching keywords, and similar organizational flow, see the Promptagator Query in Figure 2 by a standard 8-shot synthetic query generation technique (Dai et al., 2022). Though this might be effective for certain IR tasks, models may overfit to more superficial features, preventing them from understanding more conceptual relationships between query and document.

This paper focuses on developing methods to generate and utilize synthetic data to enhance the retrieval capabilities of IR systems. To mitigate overfitting, we introduce IR2 (Information Regularization for Information Retrieval), which creates queries that have conceptual overlap with the original document, but varied phrasing and structure. This yields synthetic queries that have a less explicit relationship with the document, see the Document Regularization Query in Figure 2. We posit that training models with such data can be advantageous by nudging them towards understanding deeper semantic relationships, resulting in improved retrieval performance for complex queries.

In this paper, we make several contributions to

[*] Equal contribution
[†] Equal contribution

LREC-COLING 2024, pages 9261–9284

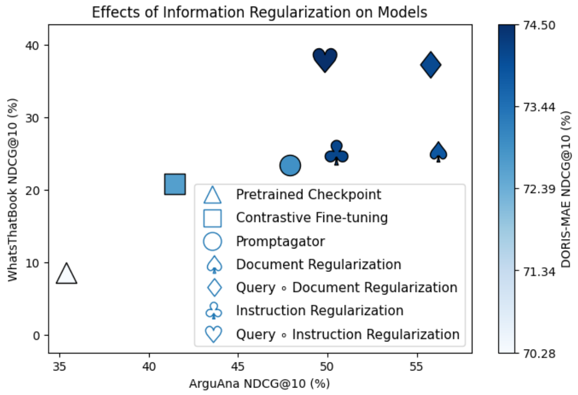


Figure 1: Performance of synthetic data generation methods on complex IR benchmarks. The △, □ (Gao et al., 2021), and ○ (Dai et al., 2022) icons represent baselines. The other four icons ♤ ♡ ♧ ♢ denote IR2 approaches, indicating the performance of models after fine-tuning on information-regularized synthetic datasets. Metrics are chosen based on standard practice for the three benchmarks. Model performances are averaged across all models used in experiments.

synthetic data generation for Information Retrieval. First, we introduce three distinct regularization techniques applied at various stages of the synthetic data generation process. This approach addresses models' tendency to learn superficial and unnatural overlapping features between synthetic queries and documents, which would hamper model performance on complex query IR tasks

We also present a comprehensive analysis of how different pretrained transformer-based embedding models respond to these regularization techniques (Wang et al., 2022c; Liu et al., 2019; Cohan et al., 2020). Our empirical evidence demonstrates consistent performance improvements with our methods compared to non-regularized baselines (Gao et al., 2021; Dai et al., 2022).

Furthermore, we find that these IR2 regularization techniques can be effectively combined, with instruction and output regularization emerging as the most potent pairing in our tests, as shown by ♡ and ♢ in Figure 1, top right corner. This finding paves the way for more effective strategies in data augmentation for IR systems.

2. Related Work

In recent years, Information Retrieval has seen increasing task complexity as well as advancements in models designed to navigate this greater complexity.

IR tasks have typically centered on retrieving relevant passages to answer simple, sentence-level queries, as seen in foundational datasets like MS MARCO (Nguyen et al., 2016) and NQ (Kwiatkowski et al., 2019). The BEIR dataset (Thakur et al., 2021) combines 19 IR tasks, most of which are sentence-level, including specific challenges like TREC-COVID (Voorhees et al., 2020) and SCIFACT (Wadden et al., 2020). Several recent datasets contain complex, paragraph-length queries. DORIS-MAE (Wang et al., 2023) poses the task of retrieving scientific abstracts given complex scientific research questions. In the WhatsThatBook dataset (Lin et al., 2023), tip-of-the-tongue user queries are used to retrieve book descriptions. In ArguAna (Wachsmuth et al., 2018), paragraph-length arguments are used to retrieve counterarguments.

Lexicon-based methods such as TF-IDF (Sparck Jones, 1972) and BM25 (Robertson et al., 2009) retrieve based on keyword matching, and are effective in scenarios with substantial token overlap between queries and relevant passages. However, their limitations became apparent as tasks required deeper semantic comprehension, especially in cases with minimal token overlap.

Transformer-based models, including various cross-encoders and dual-encoders (Wang et al., 2022c,b; Gao et al., 2021; Cohan et al., 2020; Santhanam et al., 2022; Xiong et al., 2020; Formal et al., 2021), have offered more nuanced document and query representations. They have shown strong results on benchmarks like BEIR (Thakur et al., 2021) but have struggled on complex query tasks.

Recently, improvements in LLMs, coupled with prompting techniques such as Chain-of-Thought (Kojima et al., 2022; Wei et al., 2022b) and In-Context-Learning (Chowdhery et al., 2022; Wei et al., 2022a), have influenced the field of Information Retrieval (IR) in several directions. This has included data annotation (Wang et al., 2023; Gilardi et al., 2023; Wang et al., 2022a) as well as retrieval and reranking processes (Sun et al., 2023). Most relevant for the current work, LLMs have been used for synthetic data generation, particularly in data-scarce conditions, outperforming traditional augmentation methods (Izacard et al., 2022). For instance, Ma et al. (2021); Liang et al. (2020) applied LLMs for synthetic question generation in Question Answering (QA).

In IR tasks, since user-generated queries are frequently difficult or expensive to obtain, much of the work on synthetic data has focused on generating synthetic queries. This has included InPars (Bonifacio et al., 2022; Jeronymo et al., 2023) and Promptagator (Dai et al., 2022), the latter showcasing significant success on the BEIR benchmark. Augtriever (Meng et al., 2022) introduced methods for synthetic query generation using smaller models, optimizing both time and cost. Peng et al. (2023) used soft prompt-tuning to further enhance

Promptagator Query:
As a medical researcher, I am looking for a tool that can help me efficiently explore biomedical literature, especially in the context of the COVID-19 pandemic. I need a tool that can not only retrieve relevant articles but also filter them based on clinically-relevant concepts and their relations. This tool should be able to decrease the proportion of unjudged documents and increase the precision of the search results, ensuring that I am exposed to a larger number of relevant documents. I am also interested in understanding how this concept-based literature exploration compares to traditional text-based retrieval. I would like to see both quantitative and qualitative insights into the characteristics of this approach.

Original Abstract:
The COVID-19 pandemic has driven ever-greater demand for tools which enable efficient exploration of biomedical literature. Although semi-structured information resulting from concept recognition and detection of the defining elements of clinical trials (e.g. PICO criteria) has been commonly used to support literature search, the contributions of this abstraction remain poorly understood, especially in relation to text-based retrieval. In this study, we compare the results retrieved by a standard search engine with those filtered using clinically-relevant concepts and their relations. With analysis based on the annotations from the TREC-COVID shared task, we obtain quantitative as well as qualitative insights into characteristics of relational and concept-based literature exploration. Most importantly, we find that the relational concept selection filters the original retrieved collection in a way that decreases the proportion of unjudged documents and increases the precision, which means that the user is likely to be exposed to a larger number of relevant documents.

Document Regularization Query:
I am a researcher studying the impact of COVID-19 on various sectors. I am particularly interested in the development of tools that can efficiently manage the challenges brought about by the pandemic. I am considering using a method that involves the detection of defining elements of certain trials, which has been commonly used to support related studies. However, I am unsure of the contributions of this method, especially in relation to specific aspects of the pandemic. I am planning to compare the results retrieved by this method with those filtered using a different approach. I am also interested in understanding the implications of the selection filters used in the original method. I am particularly concerned about the decrease in the number of unjudged elements and the increase in other factors. Ultimately, I want to determine if the original method is likely to be applicable to a larger number of pandemic-related studies.

Figure 2: Sample synthetic query from Promptagator and a synthetic query generated with document regularization. (Both queries are generated from the same abstract.) Red indicates overlaps between the Promptagator query and the original abstract. Blue indicates overlaps between the document regularized query and the original abstract. Green indicates overlap with both queries. The document regularized query has less textual overlap with the original query.

the quality of generated queries.

Synthetic data generation also has applications in other fields, including text classification (Li et al., 2023), information extraction (Josifoski et al., 2023), reinforcement learning for language model alignment (Yang et al., 2023), human-computer interaction (Hämäläinen et al., 2023), and computational social science (Veselovsky et al., 2023).

In addition to these LLM developments, there has been an ongoing push to improve sentence embeddings, including SimCSE (Gao et al., 2021), which applies dropout masking as a data augmentation technique, and other variants such as DiffCSE (Chuang et al., 2022), and RankCSE (Liu et al., 2023). By using a contrastive learning objective, these models have substantially increased the utility of unlabeled data for retrieval systems.

3. Methodology

3.1. Synthetic Query Generation

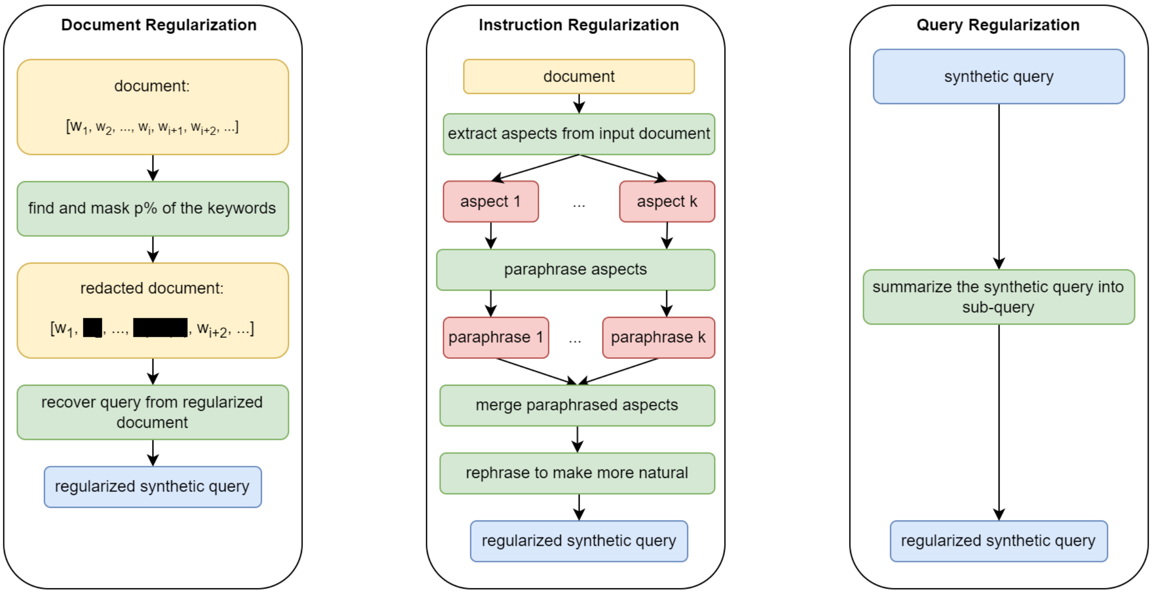
The goal of IR systems is to match user queries, which express the user's information need, with relevant documents that contain the desired information. A typical IR system would benefit from fine-tuning of pairs of query and relevant document if sufficient data is available. However, real user queries are difficult and costly to collect. In addition, finding relevant documents for user queries also requires careful annotations. Therefore, synthetic query generation aims to improve IR system performance when faced with challenges of the scarcity of query data and the lack of supervised pairs of query and relevant document. Since documents are generally more readily available, by generating artificial yet plausible user queries based on existing documents, we can create additional training data for IR systems, helping them better understand and respond to a wider range of potential user queries.

In this research, we concentrate on methods for generating synthetic queries that are relevant to given documents. This involves not just creating queries that a user might realistically pose, but also ensuring that these queries are diverse enough to train IR systems effectively. By using this synthetic data for training, we aim to improve the systems' overall ability to handle complex user queries, enhancing retrieval accuracy and efficiency.

3.2. Baseline Query Generation

Promptagator (Dai et al., 2022) proposes a few-shot method for LLM synthetic query generation. By providing an LLM with a small number of pairs of relevant documents and queries, it will implicitly learn the transformation from document to query. When a new document is provided, the LLM will transform it to a relevant query. We use d_i to denote a document and q_i to denote a relevant query. Promptagator uses 8-shot prompting of the following form:

$$d_1, q_1, \ldots, d_8, q_8, d_9,$$



(a) Document Regularization (b) Instruction Regularization (c) Query Regularization

Figure 3: Illustration of the three information regularization methods.

which includes the 8 example pairs as well as the target document d_9.

However, as shown in Figure 2, biases from the LLM may result in suboptimal queries. In our testing, we observed that queries frequently contained an implausibly large number of details from the documents, including highly similar phrasing and mirroring structure. When these queries are used for training downstream IR systems, they may lead to overfitting, as the IR systems learn simple phrase-matching heuristics between queries and documents.

3.3. Information Regularization

In order to mitigate these potential problems in the query generation process, we introduce IR2 (Information Regularization for Information Retrieval). These methods aim to reduce the number of superficial features shared by documents and synthetic queries, while still maintaining deeper semantic relationships.

The three information regularization methods can be categorized by the stage of the query generation pipeline at which they are applied:

- Document Regularization is applied to the input document d_i of LLM by intentionally withholding parts of the semantic information in d_i.
- Instruction Regularization is applied to the prompt. We design a specialized prompt for the LLM, explicitly instructing it to avoid superficial similarities between the query and document.
- Query Regularization is applied to the synthetic query which was generated by the LLM. It is designed to summarize the synthetic query, transforming it into a less complex sub-query.

Below we will discuss the motivation and intended effects of each type of regularization.

3.3.1. Input Document Regularization

Document Regularization aims to address implausibly high levels of phrase overlaps between documents and synthetic queries. This technique disrupts phrase overlaps by partially masking the document's content before feeding it into the LLM. As shown in Figure 3 (a), the system identifies key semantic words in a document and hides a random $p\%$ of them. The LLM, guided by the style of a single example query, then generates a new query based on this incomplete information.

This approach forces the LLM to generate queries that diverge textually from the source document while still remaining broadly relevant, and it makes the resulting training data more challenging for downstream IR systems. The method can be straightforwardly adapted to different IR datasets, and the $p\%$ parameter offers control over the degree of regularization. Setting $p = 0$ defaults to the original Promptagator approach, with no regularization.

3.3.2. Instruction Regularization

This method involves guiding the LLM through a structured thought process to generate queries from documents, emphasizing the extraction and synthesis of key ideas while avoiding superficial mimicry. The model is first instructed to break the document into a sequence of aspects, each describing an important part of the document. Each aspect is subsequently paraphrased. The LLM is finally instructed to combine the paraphrased aspects into a natural query. The approach is detailed in Figure 3 (b).

Due to the complexity of the prompt, we found that GPT-3.5 was unable to perform this task, while GPT-4 (which is used for our experiments) was able to.

3.3.3. Output Query Regularization

As depicted in Figure 3 (c), this technique simplifies synthetic queries, making them shorter and less detailed yet still conceptually relevant to the source document. The process challenges embedding models to focus on deeper semantic understanding rather than textual similarity.

Query regularization is straightforward, requires a small context window, and can be combined with our other synthetic generation techniques. In our experiments, we investigate the effectiveness of query regularization applied to synthetic queries generated from document regularization and instruction regularization.

4. Experiments

To evaluate the effectiveness of information regularization for synthetic query generation, we choose 3 complex-query based test benchmarks (DORIS-MAE, ArguAna and WhatsThatBook) and 4 different embedding models (E5-Large-v2 (Wang et al., 2022c), SimCSE-Large (Gao et al., 2021), RoBERTa-Large (Liu et al., 2019), SPECTER-v2 (Cohan et al., 2020)). For each benchmark, we generate 8 synthetic datasets:

- Document Regularization ($p = 40\%$)
- Document Regularization ($p = 60\%$)
- Document Regularization ($p = 80\%$)
- Instruction Regularization
- Query Regularization ∘ Document Regularization ($p = 40\%$)
- Query Regularization ∘ Document Regularization ($p = 60\%$)
- Query Regularization ∘ Document Regularization ($p = 80\%$)
- Query Regularization ∘ Instruction Regularization.

The notation ∘ indicates function composition.

In this section, we will discuss the details of benchmark datasets, pretrained models, baseline query generation methods, and regularization techniques. All experiment code is publicly accessible.[1]

For model evaluations, we report the IR metrics recommended by each benchmark's associated paper. Full experimental results with a more complete set of metrics are reported in Appendix B.

4.1. Benchmarks

WhatsThatBook (WTB) (Lin et al., 2023) contains 14,441 queries from users trying to recall specific books, paired with book titles and descriptions. The task is to match each of the 1,445 test set queries with the correct descriptions. We use descriptions from 4,000 unique books, exclusive from the test set, for synthetic query generation.

DORIS-MAE (Wang et al., 2023) includes 100 complex research queries in AI, CV, NLP, and ML, split into 40 for training and 60 for testing. Each query is associated with a candidate pool of approximately 100 research paper abstracts, with a fine-grained ranking system. The candidate pools are drawn from a corpus of approximately 360,000 computer science papers. We use 4,000 abstracts from a 360,000 paper corpus for synthetic query generation.

ArguAna (Wachsmuth et al., 2018) from BEIR (Thakur et al., 2021) consists of 8,674 paragraph-length arguments, and a test set of 1,406 argument-counterargument pairs. The task is to retrieve the correct counterargument for each test set argument. We use 5,700 arguments, not part of the test set, for synthetic counterargument generation. Due to the symmetry between arguments and counterarguments, we can use our synthetic data to train for retrieval of counterarguments given arguments.

4.2. Models

We experiment with four models: E5-Large-v2 (Wang et al., 2022c) (355M), SimCSE-Large (Gao et al., 2021) (355M), RoBERTa-Large (Liu et al., 2019) (355M), and SPECTER-v2 (Cohan et al., 2020) (110M). E5-Large-v2 (denoted as E5), a contrastively trained model, demonstrates performance on par with the state-of-the-art ada-002 (Greene et al., 2022) on BEIR (Thakur et al., 2021). For SimCSE-Large (denoted as SimCSE), which uses self-supervised training with dropout masks, we choose the top-performing checkpoint on the STS benchmark (Agirre et al., 2013). RoBERTa-Large is denoted as RoBERTa. SPECTER-v2 is

[1] https://github.com/Info-Regularization/Information-Regularization

specialized for scientific document representation. Embeddings are derived according to each model's specifications, using pooling strategies, and retrieval and reranking are based on similarity measures (L2 distance for SPECTER-v2; cosine similarity for others).

4.3. Baselines

Three baseline types were used. First, the "Pre-trained" baselines indicate performance of the pre-trained checkpoints on each dataset. Second, a "Contrastive Fine-tuning" approach, similar to SimCSE (Gao et al., 2021), applies random dropout masks to documents, creating self-supervised pairs for fine-tuning all models. This is used as a domain adaptation baseline. Third, we compare to Promptagator (Dai et al., 2022), a few-shot synthetic query generation method. Its synthetic data generation process was replicated, using GPT-4 (gpt-4-0613) instead of FLAN 137B (Wei et al., 2021).

To eliminate possible data confounds, all baselines use the same data as the regularization methods. For each benchmark, the regularization methods use a subset of documents for synthetic query generation. For the Promptagator baseline, the same documents are used for synthetic generation. For the Contrastive Fine-tuning baseline, the embedding models are directly fine-tuned on the same documents. Specifically, we train using 4000 documents from DORIS-MAE's corpus, 5700 arguments from ArguAna, and 4000 book descriptions from WhatsThatBook. We removed documents which were unusually short or long, and randomly sampled from the remaining documents.

4.4. Implementation of Regularization

Input Document Regularization: We use gpt-3.5-turbo-0301 to extract keywords/phrases from a target document, subsequently redacting $p\%$ of them. gpt-4-0613 then generates a query from this redacted document, guided solely by an example query format, without paired documents. (For ArguAna, we do not provide an example query format, since the LLM has sufficient prior knowledge of the argument and counterargument format.)

Prompt Instruction Regularization: Refer to Figure 3 (b) and Section 3.3.2. We use gpt-4-0613 with a prompt designed to elicit the breakdown and paraphrasing of query aspects. All prompts are available in the project GitHub repository.[2]

Output Query Regularization: gpt-3.5-turbo-0301 is used to summarize the original query into a simpler query. This method was applied to both document and instruction regularization outputs in our tests.

For all prompts used for keyword extraction and synthetic query generation, see Appendix A.

4.5. Training Methods

We fine-tuned each model on the different regularized datasets for one epoch, utilizing the AdamW optimizer (Loshchilov and Hutter, 2017) with an initial learning rate of 1e-5, coupled with a cosine annealing schedule. While various warm-up ratios were tested, omitting the warm-up phase yielded more stable training/validation loss curves and enhanced performance. No additional hyperparameter tuning was done; hyperparameters were set to their default values. To maintain consistent training memory, texts exceeding 512 tokens were truncated to their initial 512 tokens, a sufficient range as most texts fell within this limit. We employed the standard NT-Xent loss with in-batch negatives, setting the temperature $\tau = 0.05$ and a batch size of 80, the maximum capacity for four 40G NVIDIA A100 GPUs. To minimize gradient noise during minibatch processing, we accumulated gradients across 20 minibatches.

The synthetically generated query q_i, the relevant document d_i, and the batch size N are used as inputs to an embedding function f with range in $\mathbb{R}^n$. The NT-Xent Loss (Chen et al., 2020) is then defined as:

$$L = \mathop{\mathbb{E}}_{1 \leq i \leq N} \left[-\log \frac{e^{\langle f(q_i), f(d_i) \rangle / \tau}}{\sum\limits_{1 \leq j \leq N} e^{\langle f(q_i), f(d_j) \rangle / \tau}} \right] \quad (1)$$

All experiments were performed using 20 random seeds, with the averaged results shown in Tables 1, 2 and 3. We use t-tests to compare the effectiveness of our regularization strategies against the 8-shot Promptagator baseline. Because multiple methods are being compared, we use Bonferroni correction (Weisstein, 2004) with a correction factor 8 (the number of IR2 methods). Using the adjusted p-value, colored arrows in the tables indicate statistically significant increases or decreases (adjusted $p < 0.05$) relative to the Promptagator baseline.

5. Results

5.1. Results on WhatsThatBook

Table 1 shows comparisons between the IR2 methods and the Promptagator baseline. Except for $\text{Doc}_{80\%\ \text{reg}}$, RoBERTa and SPECTER-v2 show consistent improvement with all regularization methods. For E5 and SimCSE, there is no clear signal

[2] https://github.com/Info-Regularization/Information-Regularization/tree/main/prompt

	E5-Large-v2				RoBERTa			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	18.41	27.06	15.25	13.07	2.84	5.33	2.17	1.73
Contrastive Fine-tuning	37.25	49.85	33.16	29.87	23.59	35.27	20.69	18.10
Promptagator	41.53	54.66	37.07	33.56	27.23	39.14	23.69	20.78
$\text{Doc}_{40\%\ \text{reg}}$	41.40	53.69 ↓ -0.97	37.01	33.72	28.30 ↑ 1.07	40.89 ↑ 1.75	24.79 ↑ 1.10	21.81 ↑ 1.03
$\text{Query}_{\text{reg}} \circ \text{Doc}_{40\%\ \text{reg}}$	45.50 ↑ 3.97	58.08 ↑ 3.42	40.50 ↑ 3.43	37.01 ↑ 3.45	32.83 ↑ 5.60	46.38 ↑ 7.24	29.05 ↑ 5.36	25.82 ↑ 5.04
$\text{Doc}_{60\%\ \text{reg}}$	44.15 ↑ 2.62	55.69 ↑ 1.03	39.64 ↑ 2.57	36.16 ↑ 2.60	28.67 ↑ 1.44	41.49 ↑ 2.35	25.14 ↑ 1.45	22.06 ↑ 1.28
$\text{Query}_{\text{reg}} \circ \text{Doc}_{60\%\ \text{reg}}$	**46.23** ↑ 4.70	**59.76** ↑ 5.10	**41.58** ↑ 4.51	**38.13** ↑ 4.57	34.30 ↑ 7.07	**47.52** ↑ 8.38	**30.15** ↑ 6.46	**26.83** ↑ 6.05
$\text{Doc}_{80\%\ \text{reg}}$	28.30 ↓ -13.23	39.85 ↓ -14.81	24.05 ↓ -13.02	21.13 ↓ -12.43	21.74 ↓ -5.49	33.16 ↓ -5.98	18.91 ↓ -4.78	16.35 ↓ -4.43
$\text{Query}_{\text{reg}} \circ \text{Doc}_{80\%\ \text{reg}}$	44.80 ↑ 3.27	58.21 ↑ 3.55	40.29 ↑ 3.22	36.94 ↑ 3.38	**33.69** ↑ 6.46	46.23 ↑ 7.09	29.41 ↑ 5.72	26.20 ↑ 5.42
$\text{Instr}_{\text{reg}}$	42.15 ↑ 0.62	53.02 ↓ -1.64	37.33	33.89 ↑ 0.33	28.12 ↑ 0.89	40.80 ↑ 1.66	24.51 ↑ 0.82	21.46 ↑ 0.68
$\text{Query}_{\text{reg}} \circ \text{Instr}_{\text{reg}}$	44.90 ↑ 3.37	56.28 ↑ 1.62	39.29 ↑ 2.22	35.79 ↑ 2.23	33.09 ↑ 5.86	46.09 ↑ 6.95	29.28 ↑ 5.59	26.06 ↑ 5.28
	SimCSE				SPECTER-v2			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	16.96	24.98	14.58	12.67	2.98	5.26	2.43	1.87
Contrastive Fine-tuning	26.78	37.15	23.11	20.42	7.85	12.56	6.36	5.35
Promptagator	30.97	44.26	27.15	23.99	6.56	11.81	5.72	4.88
$\text{Doc}_{40\%\ \text{reg}}$	32.10 ↑ 1.13	43.07 ↓ -1.19	28.30 ↑ 1.15	25.45 ↑ 1.46	9.98 ↑ 3.42	16.12 ↑ 4.31	8.55 ↑ 2.83	7.32 ↑ 2.44
$\text{Query}_{\text{reg}} \circ \text{Doc}_{40\%\ \text{reg}}$	34.63 ↑ 3.66	47.44 ↑ 3.18	30.64 ↑ 3.49	27.51 ↑ 3.52	10.84 ↑ 4.28	16.26 ↑ 4.45	8.81 ↑ 3.09	7.57 ↑ 2.69
$\text{Doc}_{60\%\ \text{reg}}$	33.03 ↑ 2.06	43.96	28.87 ↑ 1.72	25.97 ↑ 1.98	8.64 ↑ 2.08	14.49 ↑ 2.68	7.46 ↑ 1.74	6.26 ↑ 1.38
$\text{Query}_{\text{reg}} \circ \text{Doc}_{60\%\ \text{reg}}$	34.84 ↑ 3.87	47.08 ↑ 2.82	30.77 ↑ 3.62	27.64 ↑ 3.65	7.54 ↑ 0.98	13.29 ↑ 1.48	6.55 ↑ 0.83	5.48 ↑ 0.60
$\text{Doc}_{80\%\ \text{reg}}$	19.75 ↓ -11.22	29.89 ↓ -14.37	16.73 ↓ -10.42	14.33 ↓ -9.66	7.39 ↑ 0.83	13.19 ↑ 1.38	6.46 ↑ 0.74	5.31 ↑ 0.43
$\text{Query}_{\text{reg}} \circ \text{Doc}_{80\%\ \text{reg}}$	33.76 ↑ 2.79	45.63 ↑ 1.37	29.34 ↑ 2.19	26.26 ↑ 2.27	8.14 ↑ 1.58	14.69 ↑ 2.88	7.33 ↑ 1.61	6.13 ↑ 1.25
$\text{Instr}_{\text{reg}}$	33.81 ↑ 2.84	45.79 ↑ 1.53	29.74 ↑ 2.59	26.79 ↑ 2.80	10.49 ↑ 3.93	16.80 ↑ 4.99	9.30 ↑ 3.58	8.07 ↑ 3.19
$\text{Query}_{\text{reg}} \circ \text{Instr}_{\text{reg}}$	**37.34** ↑ 6.37	**50.45** ↑ 6.19	**33.18** ↑ 6.03	**30.02** ↑ 6.03	**11.08** ↑ 4.52	**17.36** ↑ 5.55	**9.59** ↑ 3.87	**8.28** ↑ 3.40

Table 1: Results for WhatsThatBook. The average of 20 random trials is reported for each model/method/metric. Green arrow indicates a statistically significant ($p < 0.05$) increase over Promptagator baseline. Red arrow indicates a significant decrease.

of improvement from document or instruction regularization alone. However, query regularization, combined with either document or instruction regularization, improves performance across all models.

Not all models respond to fine-tuning on synthetic datasets in the same way. SPECTER-v2 and RoBERTa start from similar performance levels, but RoBERTa reaches considerably higher performance after fine-tuning. This suggests that the effectiveness of synthetic data augmentation is model-dependent. The SPECTER-v2 checkpoint had been previously fine-tuned for scientific document understanding, possibly interfering with its performance on the WhatsThatBook dataset.

Document regularization with an 80% masking ratio leads to synthetic queries that are either too vague or contain misleading and false information. Consequently, for E5, RoBERTa, and SimCSE, we observed that document regularization with an 80% mask resulted in significantly lower performances compared to Promptagator. Results are stronger with a masking level between 40% and 60%.

5.2. Results on DORIS-MAE

Table 2 shows that instruction regularization, alone or combined with query regularization, consistently enhances performance across various models and metrics, surpassing the Promptagator's synthetic dataset, with minor exceptions within a 1% margin. The strongest results across all models and data synthesis methods are achieved by query regularization combined with instruction regularization.

Document regularization has variable performance depending on the mask percentage ($p = 40\%, 60\%, 80\%$) and model. It has only a small effect on the performance of E5-Large-v2 compared to the Promptagator baseline, and underperforms with SimCSE. However, it substantially improves performance when applied to RoBERTa and SPECTER-v2.

For E5, SimCSE, and RoBERTa, integrating query regularization with document or instruction regularization improves performance across all metrics. Query regularization decreases the textual similarity between queries and their target documents, suggesting that models are learning deeper semantic relationships between queries and documents.

5.3. Results on ArguAna

In Table 3, we observe that the regularized synthetic datasets consistently outperform baseline approaches. While instruction regularization continues to outperform baselines, all models achieve the strongest performance when fine-tuning on document regularized dataset. Similar to our observations in DORIS-MAE and WTB, query regularization, when combined with another synthetic query method, remains consistently strong.

	E5-Large-v2				RoBERTa			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	14.67	42.15	71.98	14.34	11.96	34.77	66.86	8.56
Contrastive Fine-tuning	15.28	39.87	72.51	13.43	14.75	43.32	73.55	**20.79**
Promptagator	16.18	45.59	73.95	14.83	13.03	42.88	72.23	16.82
Doc$_{40\%\ reg}$	14.90 ↓ -1.28	46.55 ↑ 0.96	74.28 ↑ 0.33	14.87	12.54	44.89 ↑ 2.01	72.94 ↑ 0.71	18.16 ↑ 1.34
Query$_{reg}$ ∘ Doc$_{40\%\ reg}$	15.85	45.67	73.96	16.10 ↑ 1.27	14.66 ↑ 1.63	47.41 ↑ 4.53	74.04 ↑ 1.81	18.81 ↑ 1.99
Doc$_{60\%\ reg}$	15.23 ↓ -0.95	46.57 ↑ 0.98	74.98 ↑ 1.03	15.14	14.11 ↑ 1.08	43.99 ↑ 1.11	72.94 ↑ 0.71	16.84
Query$_{reg}$ ∘ Doc$_{60\%\ reg}$	15.31 ↓ -0.87	48.16 ↑ 2.57	74.90 ↑ 0.95	15.43 ↑ 0.60	14.75 ↑ 1.72	47.25 ↑ 4.37	74.61 ↑ 2.38	20.48 ↑ 3.66
Doc$_{80\%\ reg}$	14.68 ↓ -1.50	43.89 ↓ -1.70	74.13 ↑ 0.18	13.51 ↓ -1.32	14.47 ↑ 1.44	42.47	72.61 ↑ 0.38	15.10 ↓ -1.72
Query$_{reg}$ ∘ Doc$_{80\%\ reg}$	15.63 ↓ -0.55	46.75 ↑ 1.16	75.08 ↑ 1.13	16.62 ↑ 1.79	**15.84** ↑ 2.81	46.96 ↑ 4.08	**74.71** ↑ 2.48	20.49 ↑ 3.67
Instr$_{reg}$	**16.38**	46.73 ↑ 1.14	74.42 ↑ 0.47	17.06 ↑ 2.23	13.80 ↑ 0.77	45.78 ↑ 2.90	73.53 ↑ 1.30	18.24 ↑ 1.42
Query$_{reg}$ ∘ Instr$_{reg}$	15.76 ↓ -0.42	**48.46** ↑ 2.87	**76.02** ↑ 2.07	**20.86** ↑ 6.03	15.20 ↑ 2.17	**47.44** ↑ 4.56	74.08 ↑ 1.85	19.31 ↑ 2.49
	SimCSE				SPECTER-v2			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	14.38	41.83	70.81	**24.83**	13.59	41.92	71.46	21.74
Contrastive Fine-tuning	15.60	44.99	73.18	22.41	14.56	**45.64**	71.46	20.49
Promptagator	16.35	46.69	73.97	16.54	14.13	39.22	71.55	22.37
Doc$_{40\%\ reg}$	**17.70** ↑ 1.35	45.30 ↓ -1.39	74.49 ↑ 0.52	16.49	15.39 ↑ 1.26	43.39 ↑ 4.17	72.11 ↑ 0.56	21.66 ↓ -0.71
Query$_{reg}$ ∘ Doc$_{40\%\ reg}$	16.06	45.46 ↓ -1.23	73.23 ↓ -0.74	16.56	14.19	42.66 ↑ 3.44	71.49	21.40 ↓ -0.97
Doc$_{60\%\ reg}$	17.08 ↑ 0.73	45.33 ↓ -1.36	74.70 ↑ 0.73	17.75 ↑ 1.21	15.41 ↑ 1.28	44.19 ↑ 4.97	72.81 ↑ 1.26	21.35 ↓ -1.02
Query$_{reg}$ ∘ Doc$_{60\%\ reg}$	17.04 ↑ 0.69	46.68	74.65 ↑ 0.68	18.32 ↑ 1.78	14.95 ↑ 0.82	43.30 ↑ 4.08	72.19 ↑ 0.64	21.60 ↓ -0.77
Doc$_{80\%\ reg}$	15.01 ↓ -1.34	42.53 ↓ -4.16	73.69 ↓ -0.28	18.68 ↑ 2.14	**17.02** ↑ 2.89	44.97 ↑ 5.75	**73.94** ↑ 2.39	**22.78** ↑ 0.41
Query$_{reg}$ ∘ Doc$_{80\%\ reg}$	15.92	45.70 ↓ -0.99	73.86	18.54 ↑ 2.00	14.70 ↑ 0.57	44.66 ↑ 5.44	73.07 ↑ 1.52	22.65
Instr$_{reg}$	16.50	46.68	**75.60** ↑ 1.63	19.71 ↑ 3.17	14.73 ↑ 0.60	41.50 ↑ 2.28	73.06 ↑ 1.51	21.05 ↓ -1.32
Query$_{reg}$ ∘ Instr$_{reg}$	16.06	**47.61** ↑ 0.92	75.43 ↑ 1.46	22.14 ↑ 5.60	14.50 ↑ 0.37	41.44 ↑ 2.22	72.46 ↑ 0.91	22.26

Table 2: Results for DORIS-MAE. See Table 1 caption for reporting conventions.

	E5-Large-v2				RoBERTa			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	59.37	88.16	47.75	38.96	30.51	49.68	23.56	18.49
Contrastive Fine-tuning	70.69	94.77	56.15	47.13	59.12	84.67	47.57	39.65
Promptagator	69.41	94.11	55.14	46.67	61.51	88.19	49.80	41.85
Doc$_{40\%\ reg}$	**78.72** ↑ 9.31	**96.46** ↑ 2.35	64.24 ↑ 9.10	56.45 ↑ 9.78	67.38 ↑ 5.87	91.03 ↑ 2.84	54.54 ↑ 4.74	46.43 ↑ 4.58
Query$_{reg}$ ∘ Doc$_{40\%\ reg}$	77.76 ↑ 8.35	96.21 ↑ 2.10	63.00 ↑ 7.86	55.24 ↑ 8.57	67.29 ↑ 5.78	**91.08** ↑ 2.89	54.04 ↑ 4.24	45.85 ↑ 4.00
Doc$_{60\%\ reg}$	78.16 ↑ 8.75	96.41 ↑ 2.30	**64.41** ↑ 9.27	**56.57** ↑ 9.90	67.51 ↑ 6.00	90.77 ↑ 2.58	**54.85** ↑ 5.05	**46.85** ↑ 5.00
Query$_{reg}$ ∘ Doc$_{60\%\ reg}$	78.34 ↑ 8.93	96.37 ↑ 2.26	64.10 ↑ 8.96	56.24 ↑ 9.57	67.71 ↑ 6.20	90.99 ↑ 2.80	54.41 ↑ 4.61	46.34 ↑ 4.49
Doc$_{80\%\ reg}$	77.37 ↑ 7.96	95.57 ↑ 1.46	63.14 ↑ 8.00	55.16 ↑ 8.49	**67.82** ↑ 6.31	90.18 ↑ 1.99	54.62 ↑ 4.82	46.79 ↑ 4.94
Query$_{reg}$ ∘ Doc$_{80\%\ reg}$	78.44 ↑ 9.03	96.25 ↑ 2.14	64.36 ↑ 9.22	56.49 ↑ 9.82	67.38 ↑ 5.87	90.70 ↑ 2.51	54.53 ↑ 4.73	46.59 ↑ 4.74
Instr$_{reg}$	72.06 ↑ 2.65	95.70 ↑ 1.59	57.76 ↑ 2.62	49.17 ↑ 2.50	64.47 ↑ 2.96	90.09 ↑ 1.90	51.49 ↑ 1.69	43.20 ↑ 1.35
Query$_{reg}$ ∘ Instr$_{reg}$	71.59 ↑ 2.18	95.18 ↑ 1.07	56.89 ↑ 1.75	48.37 ↑ 1.70	65.14 ↑ 3.63	90.26 ↑ 2.07	52.25 ↑ 2.45	43.89 ↑ 2.04
	SimCSE				SPECTER-v2			
Method	R@5	R@20	NDCG@10	MRR@10	R@5	R@20	NDCG@10	MRR@10
Pretrained	49.03	81.12	39.23	30.35	38.91	71.79	30.91	23.23
Contrastive Fine-tuning	36.70	64.79	28.85	22.12	41.90	75.70	33.20	25.26
Promptagator	69.49	92.54	56.38	47.97	38.64	75.15	30.24	21.77
Doc$_{40\%\ reg}$	74.92 ↑ 5.43	93.36 ↑ 0.82	60.24 ↑ 3.86	52.42 ↑ 4.45	56.09 ↑ 17.45	86.32 ↑ 11.17	45.00 ↑ 14.76	36.30 ↑ 14.53
Query$_{reg}$ ∘ Doc$_{40\%\ reg}$	**76.07** ↑ 6.58	**93.97** ↑ 1.43	**61.38** ↑ 5.00	**53.57** ↑ 5.60	51.28 ↑ 12.64	83.33 ↑ 8.18	40.59 ↑ 10.35	31.81 ↑ 10.04
Doc$_{60\%\ reg}$	73.20 ↑ 3.71	92.10	58.46 ↑ 2.08	50.58 ↑ 2.61	58.87 ↑ 20.23	86.93 ↑ 11.78	46.55 ↑ 16.31	37.88 ↑ 16.11
Query$_{reg}$ ∘ Doc$_{60\%\ reg}$	75.75 ↑ 6.26	93.19 ↑ 0.65	60.94 ↑ 4.56	53.24 ↑ 5.27	54.22 ↑ 15.58	84.97 ↑ 9.82	42.90 ↑ 12.66	34.21 ↑ 12.44
Doc$_{80\%\ reg}$	70.38	89.67 ↓ -2.87	55.76	47.97	**61.02** ↑ 22.38	**87.48** ↑ 12.33	**48.68** ↑ 18.44	**40.24** ↑ 18.47
Query$_{reg}$ ∘ Doc$_{80\%\ reg}$	74.10 ↑ 4.61	92.59	59.79 ↑ 3.41	52.18 ↑ 4.21	57.55 ↑ 18.91	86.65 ↑ 11.50	46.01 ↑ 15.77	37.44 ↑ 15.67
Instr$_{reg}$	70.97	93.91 ↑ 1.37	57.41	48.80	43.26 ↑ 4.62	80.16 ↑ 5.01	34.80 ↑ 4.56	26.07 ↑ 4.30
Query$_{reg}$ ∘ Instr$_{reg}$	71.46 ↑ 1.97	93.78 ↑ 1.24	58.16 ↑ 1.78	49.78 ↑ 1.81	38.57	77.12 ↑ 1.97	31.72 ↑ 1.48	23.15 ↑ 1.38

Table 3: Results for ArguAna. See Table 1 caption for reporting conventions.

5.4. Analysis and Discussion

Our experiments demonstrate substantial improvements from regularizing synthetic queries in IR tasks. Rather than simply altering task difficulty, these methods refine the model's ability to discern and process relevant information, an advancement over existing strategies. Specifically, integrating query and instruction regularization emerges as a robust method, enhancing the model's performance consistently.

Models trained with the proposed regularized synthetic data generation strategies not only outperform pretrained counterparts but also exhibit considerable gains over existing synthetic data methods like Promptagator. This is shown across various metrics and models.

The practical implications of these methods extend beyond improved model performance. They point toward a greater adaptability in handling complex information retrieval tasks, a critical component given the variability of real-world applications. Moreover, the insights gained from this research contribute to the broader discourse on the role of synthetic data in model training, especially in scenarios where data scarcity is a challenge.

Though IR2 improves over prior methods, there are several caveats. The document regularization method, which masks part of the input document and generates a query from this masked document, sometimes results in hallucinations in the queries. We hypothesize that performance differences across datasets are due to varying task sensitivity to these hallucinations. Performance was generally weaker with a mask ratio above 60%.

As shown in Table 1, document regularization had weaker performance on the WhatsThatBook dataset. This task requires a match between fine-grained details in the query and document. In contrast, for the ArguAna dataset in Table 3, where the task involves pairing arguments with counter-arguments, the method had stronger performance. Thematic similarity is sufficient for this task.

5.5. Cost Analysis

Instruction Regularization, which costs $100-200 per dataset, is the most expensive of the three regularization methods. Due to longer prompts with the inclusion of 8 example pairs, the baseline method Promptagator costs $400-$600 per dataset.

We attempted to generate synthetic queries by hand to estimate human labor costs, and required at least 5 minutes per query. Assuming a wage of $15 per hour, human generation costs at least $5,000 per dataset. This suggests substantial savings by using GPT-4.

6. Acknowledgement

We would like to thank Mahidhar Tatineni and the San Diego Supercomputer Center for providing compute support.

7. Limitations

The effectiveness of our regularization strategies does show variability across different models, signaling that these methods may be more suited to certain architectures or training setups. Future work could delve deeper into this aspect, aiming to identify the underpinning factors that contribute to these discrepancies.

Expanding on the current research, there is an opportunity to explore information regularization in a more granular context, possibly by introducing more sophisticated measures than the existing $p\%$ parameter. Understanding the synergy between different regularization strategies could also unveil new insights into optimal model training practices.

Our experiments relied on models which were accessed through API. We use stable checkpoints for these models, gpt-4-0613 and gpt-3.5-turbo-0301, which will allow for reproducibility as long as these checkpoints are maintained.

8. Bibliographical References

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, Seattle, United States. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade v2: Sparse lexical and expansion model for information retrieval. *arXiv preprint arXiv:2109.10086*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd-workers for text-annotation tasks. *arXiv preprint arXiv:2303.15056*.

Ryan Greene, Ted Sanders, Lilian Weng, and Arvind Neelakantan. 2022. New and improved embedding model. `https://openai.com/blog/new-and-improved-embedding-model/`. Accessed: 2023-06-03.

Perttu Hämäläinen, Mikke Tavast, and Anton Kunnari. 2023. Evaluating large language models in generating synthetic hci research data: a case study. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA. Association for Computing Machinery.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *arXiv preprint arXiv:2301.01820*.

Martin Josifoski, Marija Sakota, Maxime Peyrard, and Robert West. 2023. Exploiting asymmetry for synthetic training data generation: Synthie and the case of information extraction. *arXiv preprint arXiv:2303.04132*.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. *arXiv preprint arXiv:2310.07849*.

Davis Liang, Peng Xu, Siamak Shakeri, Cicero Nogueira dos Santos, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Embedding-based zero-shot retrieval through query generation. *arXiv preprint arXiv:2009.10270*.

Kevin Lin, Kyle Lo, Joseph E Gonzalez, and Dan Klein. 2023. Decomposing complex queries for tip-of-the-tongue retrieval. *arXiv whapreprint arXiv:2305.15053*.

Jiduan Liu, Jiahao Liu, Qifan Wang, Jingang Wang, Wei Wu, Yunsen Xian, Dongyan Zhao, Kai Chen, and Rui Yan. 2023. RankCSE: Unsupervised sentence representations learning via learning to rank. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13785–13802, Toronto, Canada. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. 2021. Zero-shot neural passage retrieval via domain-targeted synthetic question generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1075–1088, Online. Association for Computational Linguistics.

Rui Meng, Ye Liu, Semih Yavuz, Divyansh Agarwal, Lifu Tu, Ning Yu, Jianguo Zhang, Meghana Bhat, and Yingbo Zhou. 2022. Augtriever: Unsupervised dense retrieval by scalable data augmentation. *arXiv preprint arXiv:2212.08841*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *choice*, 2640:660.

Zhiyuan Peng, Xuyang Wu, and Yi Fang. 2023. Soft prompt tuning for augmenting dense retrieval with large language models. *arXiv preprint arXiv:2307.08303*.

Gustavo Penha, Enrico Palumbo, Maryam Aziz, Alice Wang, and Hugues Bouchard. 2023. Improving content retrievability in search with controllable query generation. In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 3182–3192, New York, NY, USA. Association for Computing Machinery.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agent. *arXiv preprint arXiv:2304.09542*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Veniamin Veselovsky, Manoel Horta Ribeiro, Akhil Arora, Martin Josifoski, Ashton Anderson, and Robert West. 2023. Generating faithful synthetic data with large language models: A case study in computational social science. *arXiv preprint arXiv:2305.15041*.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251, Melbourne, Australia. Association for Computational Linguistics.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

Jianyou Wang, Kaicheng Wang, Xiaoyue Wang, Prudhviraj Naidu, Leon Bergen, and Ramamohan Paturi. 2023. Doris-mae: Scientific document retrieval using multi-level aspect-based queries. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022a. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360, Seattle, United States. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022b. Simlm: Pre-training with representation bottleneck for dense passage retrieval. *arXiv preprint arXiv:2207.02578*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022c. Text embeddings by weakly-supervised contrastive pre-training.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Eric W Weisstein. 2004. Bonferroni correction. *https://mathworld. wolfram. com/*.

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*.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2023. Rlcd: Reinforcement learning from contrast distillation for language model alignment. *arXiv preprint arXiv:2307.12950*.

9. Language Resource References

Eneko Agirre and Daniel Cer and Mona Diab and Aitor Gonzalez-Agirre and Weiwei Guo. 2013. *Semantic Textual Similarity (STS) 2013 Machine Translation*. Linguistic Data Consortium, ISLRN 857-492-590-583-5. PID https://doi.org/10.35111/cy4d-7c39.

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

Lin, Kevin and Lo, Kyle and Gonzalez, Joseph E and Klein, Dan. 2023. *Decomposing Complex Queries for Tip-of-the-tongue Retrieval*. arXiv preprint arXiv:2305.15053. PID https://huggingface.co/datasets/nlpkevinl/whatsthatbook.

Voorhees, Ellen and Alam, Tasmeer and Bedrick, Steven and Demner-Fushman, Dina and Hersh, William R. and Lo, Kyle and Roberts, Kirk and Soboroff, Ian and Wang, Lucy Lu. 2020. *TREC-COVID: constructing a pandemic information retrieval test collection*. ACM SIGIR Forum. PID https://doi.org/10.1145/3451964.3451965.

Henning Wachsmuth and Shahbaz Syed and Benno Stein. 2018. *ArguAna Counterargs*. Zenodo. PID https://doi.org/10.5281/zenodo.3973258.

Wang, Jianyou and Wang, Kaicheng and Wang, Xiaoyue and Naidu, Prudhviraj and Bergen, Leon and Paturi, Ramamohan. 2023. *DORIS-MAE-v1*. Zenodo. PID https://doi.org/10.5281/zenodo.8299749.

A. Prompt Design

A.1. DORIS-MAE Prompt

Note, Query Regularization for DORIS-MAE will generate 4 shorter synthetic queries per document, we randomly choose one for later training.

- Figure 4 for Instruction Regularization prompt
- Figure 5 for Query Regularization on Instruction Regularization prompt
- Figure 6 for Document Regularization 40%, 60% prompt
- Figure 7 for Document Regularization 80% prompt
- Figure 8 for Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%
- Figure 9 for keywords extraction prompt
- Figure 10 for Promptagator prompt example

A.2. ArguAna Prompt

- Figure 11 for Instruction Regularization prompt
- Figure 12 for Query Regularization on Instruction Regularization prompt
- Figure 13 for Document Regularization 40%, 60%, and 80% prompt
- Figure 14 for Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%
- Figure 15 for keywords extraction prompt
- Figure 16 for Promptagator prompt example

A.3. WhatsThatBook Prompt

- Figure 17 for Instruction Regularization prompt
- Figure 18 for Query Regularization on Instruction Regularization prompt
- Figure 19 for Document Regularization 40% prompt
- Figure 20 for Document Regularization 60%, 80% prompt
- Figure 21 for Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%
- Figure 22 for keywords extraction prompt
- Figure 23 for Promptagator prompt example

B. Experiment Details

We provide the experimental results on the three datasets in Table 4, 5, 6 with a more complete set of metrics. The results are consistent with those reported in Section 5.

I want you to transform computer science abstracts into queries in a particular manner. Here is one example of the transformation:
{Example_abstract}
->
Example Query: **{Example_query}**

I am going to present you with a new abstract. You should transform it into a new query, while satisfying the following requirements:
1. The query should, broadly speaking, be addressed by the abstract.
2. The query should not contain the distinctive keywords contained in an abstract. Put another way, simple keyword matching *should not* be sufficient to retrieve the abstract given the query. When you encounter distinctive keywords, think of alternative paraphrases. This is very important.
3. The query should describe a natural set of questions or goals that a scientific researcher could have.

Abstract:
{Abstract for new query}

Before giving the final query, first think step by step, to make sure that you are satisfying the following constraints.
1. Identify the important questions that are addressed by the abstract.
2. Paraphrase each important question. In order to decide whether a key phrase has been sufficiently reworked, think about whether a naive search engine would be able to find the abstract given the phrase. Highly specialized terms -- terms that are introduced by the abstract which are likely to be distinctive to only a few papers -- should not be included. However, you *should not* paraphrase common technical terms which occur in many papers.
3. Consolidate the questions into a natural query -- not all questions need to be included, and you can combine/synthesize questions to make things more natural.
4. Write the query.
Do steps 1-4 above incrementally

Figure 4: DORIS-MAE Instruction Regularization Prompt

Given a complex query, extract 4 disjoint separate problem statements from it.
Query:
{query}

Figure 5: DORIS-MAE Query Regularization on Instruction Regularization Prompt

Here is a redacted scientific abstract: "**{Redacted_abstract}**".

Here is the example query based on other abstract: "**{Example_query}**".

Follow the style of the example query, write a new **query** based on the provided abstract.

Only output the new query (strictly more than 125 words). The query should not contain "_". Note that the query should mimic the style of the example query and base on the redacted query.

Figure 6: DORIS-MAE Document Regularization 40%, 60% Prompt

Here is a heavily redacted scientific abstract: "**{Redacted_abstract}**".

Here is the example query based on other abstract: "**{Example_query}**".

Follow the style of the example query, write a new **query** based on the provided abstract.

Only output the new query (strictly more than 125 words). The query should not contain "_". Note that the query should mimic the style of the example query and base on the redacted query.

Figure 7: DORIS-MAE Document Regularization 80% Prompt

Given a complex query, extract 3 disjoint separate problem statements from it.
Query: **{query}**

Figure 8: DORIS-MAE Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%

Here is a paragraph: **{Abstract}**. Give me **all** the important words that represent salient ideas in the paragraph. Output the words in a python list format.

Figure 9: DORIS-MAE Keywords Extraction Prompt

Passage_0: **{Abstract_0}**

Query_0: **{Query_0}**

Passage_1: **{Abstract_1}**

Query_1: **{Query_1}**

Passage_2: **{Abstract_2}**

Query_2: **{Query_2}**

Passage_3: **{Abstract_3}**

Query_3: **{Query_3}**

Passage_4: **{Abstract_4}**

Query_4: **{Query_4}**

Passage_5: **{Abstract_5}**

Query_5: **{Query_5}**

Passage_6: **{Abstract_6}**

Query_6: **{Query_6}**

Passage_7: **{Abstract_7}**

Query_7: **{Query_7}**

Passage_8: **{Abstract for new query}**

Write Query_8

Figure 10: DORIS-MAE Promptagator Prompt

I want you to transform the argument to counterargument in a particular manner. Here is one example of the transformation:

{Example_argument} ->

{Example_counterargument}

As a debater, I am going to present you with a new argument. You should transform the argument to a counterargument while satisfying the following requirements:
1. The counterargument takes on the aspects of the topic invoked by the argument, while adding a new perspective to its conclusion and/or premises, conveying the opposite stance.
2. The counterargument should not refute the argument in any sense. Instead, the counterargument should be an independent argument that conveys another point of view. Think of this as if you are the first speaker in a debate and you should primarily focus on introducing and stating your point of view. You should present your stand and arguments on the given topic. You should not refute the opponent's ideas as these arguments have not yet been presented.

Argument: **{Argument for new counterargument}**

Figure 11: ArguAna Instruction Regularization Prompt

Given an argument, summarize it down to a shorter argument with at most 50 words.
Argument: **{argument}**

Figure 12: ArguAna Query Regularization on Instruction Regularization Prompt

Given a redacted argument that has multiple positions it supports, determine one of the positions that it supports, and generate a one paragraph counterargument that has a new perspective on it. The counterargument should be an independent argument. The counter argument *Must Not* have the word argument in it. Output the counter argument after the key word "Counter:"

{redacted_argument}

Figure 13: ArguAna Document Regularization 40%, 60%, and 80% Prompt

Given an argument, summarize it down to a shorter argument with at most 50 words.
Argument: **{argument}**

Figure 14: ArguAna Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%

Here is an argument: **{argument}**
Give me **all** the important words that represent salient ideas in the argument. Output the words in a python list format.

Figure 15: ArguAna Keywords Extraction Prompt

Argument_0:**{argumenet_0}**

Counterargument_0:**{counterargumenet_0}**

Argument_1:**{argumenet_1}**

Counterargument_1:**{counterargumenet_1}**

Argument_2:**{argumenet_2}**

Counterargument_2:**{counterargumenet_2}**

Argument_3:**{argumenet_3}**

Counterargument_3:**{counterargumenet_3}**

Argument_4:**{argumenet_4}**

Counterargument_4:**{counterargumenet_4}**

Argument_5:**{argumenet_5}**

Counterargument_5:**{counterargumenet_5}**

Argument_6:**{argumenet_6}**

Counterargument_6:**{counterargumenet_6}**

Argument_7:**{argumenet_7}**

Counterargument_7:**{counterargumenet_7}**

Argument_8:**{argumenet for new counterargument}**

Write Counterargument_8

Figure 16: ArguAna Promptagator Prompt

I want you to transform the description of a book into a tip of the tongue query in a particular manner. Here is one example of the transformation:

Example Description:
{Exmaple_description}

Example Query:
{Example_query}

I am going to present you with a new book description. You should transform it into a new query, while satisfying the following requirements:
1. The query should, broadly speaking, be addressed by the description.
2. The query should not contain the distinctive keywords contained in a description. Put another way, simple keyword matching *should not* be sufficient to retrieve the description given the query. When you encounter distinctive keywords, think of alternative paraphrases. This is very important.
3. The query should be in tip-of-the-tongue format. Think of a situation where a user wants to find a book that they have previously read. The user may be uncertain about identifying details and may rely on creative strategies for describing the information they want to retrieve. These strategies include text that describes content elements (e.g., book characters or events), information beyond the document text (e.g., descriptions of book covers), or personal context (e.g., when they read a book).
4. The query's length should try to match the new book description length.

New book description:
{Description for new query}

Before giving the final query, first think step by step, to make sure that you are satisfying the following constraints.
1. Identify the important features in the description.
2. Paraphrase each important feature. In order to decide whether a key phrase has been sufficiently reworked, think about whether a naive search engine would be able to find the abstract given the phrase. Highly specific terms should not be included.
3. Consolidate the features into a natural query -- not all features need to be included, and you can combine/synthesize features to make things more natural.
4. Ensure your query matches the length of the new book description. Write *only* the query.
Do steps 1-4 above incrementally

Figure 17: WhatsThatBook Instruction Regularization Prompt

Given a query, summarize it down to a shorter query with at most 50 words.
Query: **{query}**

Figure 18: WhatsThatBook Query Regularization on Instruction Regularization Prompt

Here is a redacted book description: "**{redacted_description}**"

Here is an example tip-of-tongue query based on an example book description: "**{Example_query}**".

Follow the style of the example tip-of-tongue query, write a new tip-of-tongue **query** based on the provided book description.

Only output the new query (strictly more than 125 words) in one paragraph. The query should not contain "_". Note that the query should mimic the style of the example query and entirely base on the redacted book description.

Figure 19: WhatsThatBook Document Regularization 40% Prompt

Here is a heavily redacted book description: "**{redacted_description}**".

Here is an example tip-of-tongue query based on an example book description: "**{Example_query}**".

Follow the style of the example tip-of-tongue query, write a new tip-of-tongue **query** based on the provided book description.

Only output the new query (strictly more than 125 words) in one paragraph. The query should not contain "_". Note that the query should mimic the style of the example query and entirely base on the redacted book description.

Figure 20: WhatsThatBook Document regularization 60%, 80% Prompt

Given a query, summarize it down to a shorter query with at most 50 words.
Query: **{query}**

Figure 21: WhatsThatBook Query Regularization on synthetic queries from Doc Reg 40%, 60%, and 80%

Here is a description of a book: **{description}**
Give me **all** the important words that represent salient ideas in the description. Output the words in a python list format.

Figure 22: WhatsThatBook Keywords Extraction Prompt

Description_1:
{Description_1}
Query_1:
{Query_1}

Description_2:
{Description_2}
Query_2:
{Query_2}

Description_3:
{Description_3}
Query_3:
{Query_3}

Description_4:
{Description_4}
Query_4:
{Query_4}

Description_5:
{Description_5}
Query_5:
{Query_5}

Description_6:
{Description_6}
Query_6:
{Query_6}

Description_7:
{Description_7}
Query_7:
{Query_7}

Description_8:
{Description_8}
Query_8:
{Query_8}

Description_9:
{Description for new query}
Write Query_9 according to Description_9

Figure 23: WhatsThatBook Promptagator Prompt

Model (Method)	R@5	R@10	R@20	RP	NDCG@10	MRR@10	MAP
			E5-Large-v2				
Pretrained	14.67	25.98	42.15	38.18	71.98	14.34	40.52
Contrastive Fine-tuning	15.28	25.97	39.87	34.73	72.51	13.43	38.87
Promptagator	16.18	26.58	45.59	38.99	73.95	14.83	42.69
Document$_{40\% \text{ reg}}$	14.90 ↓ -1.28	27.65 ↑ 1.07	46.55 ↑ 0.96	39.33	74.28 ↑ 0.33	14.87	43.10 ↑ 0.41
Query$_{\text{reg}}$ $\circ$ Document$_{40\% \text{ reg}}$	15.85	27.44 ↑ 0.86	45.67	39.73 ↑ 0.74	73.96	16.10 ↑ 1.27	43.06 ↑ 0.37
Document$_{60\% \text{ reg}}$	15.23 ↓ -0.95	27.68 ↑ 1.10	46.57 ↑ 0.98	38.86	74.98 ↑ 1.03	15.14	43.26 ↑ 0.57
Query$_{\text{reg}}$ $\circ$ Document$_{60\% \text{ reg}}$	15.31 ↓ -0.87	27.74 ↑ 1.16	48.16 ↑ 2.57	39.60 ↑ 0.61	74.90 ↑ 0.95	15.43 ↑ 0.60	44.48 ↑ 1.79
Document$_{80\% \text{ reg}}$	14.68 ↓ -1.50	25.02 ↓ -1.56	43.89 ↓ -1.70	38.91	74.13 ↑ 0.18	13.51 ↓ -1.32	43.31 ↑ 0.62
Query$_{\text{reg}}$ $\circ$ Document$_{80\% \text{ reg}}$	15.63 ↓ -0.55	27.93 ↑ 1.35	46.75 ↑ 1.16	39.37	75.08 ↑ 1.13	16.62 ↑ 1.79	44.24 ↑ 1.55
Instr$_{\text{reg}}$	**16.38**	26.26	46.73 ↑ 1.14	40.31 ↑ 1.32	74.42 ↑ 0.47	17.06 ↑ 2.23	44.84 ↑ 2.15
Query$_{\text{reg}}$ $\circ$ Instr$_{\text{reg}}$	15.76 ↓ -0.42	**30.56** ↑ 3.98	**48.46** ↑ 2.87	**42.25** ↑ 3.26	**76.02** ↑ 2.07	**20.86** ↑ 6.03	**46.29** ↑ 3.60
			SimCSE				
Pretrained	14.38	23.24	41.83	36.27	70.81	**24.83**	42.02
Contrastive Fine-tuning	15.60	27.70	44.99	39.06	73.18	22.41	42.87
Promptagator	16.35	28.98	46.69	40.64	73.97	16.54	44.12
Document$_{40\% \text{ reg}}$	**17.70** ↑ 1.35	**29.85** ↑ 0.87	45.30 ↓ -1.39	40.43	74.49 ↑ 0.52	16.49	44.67 ↑ 0.55
Query$_{\text{reg}}$ $\circ$ Document$_{40\% \text{ reg}}$	16.06	28.77	45.46 ↓ -1.23	40.41	73.23 ↓ -0.74	16.56	43.70 ↓ -0.42
Document$_{60\% \text{ reg}}$	17.08 ↑ 0.73	28.80	45.33 ↓ -1.36	39.91 ↓ -0.73	74.70 ↑ 0.73	17.75 ↑ 1.21	44.50 ↑ 0.38
Query$_{\text{reg}}$ $\circ$ Document$_{60\% \text{ reg}}$	17.04 ↑ 0.69	28.63	46.68	40.59	74.65 ↑ 0.68	18.32 ↑ 1.78	45.05 ↑ 0.93
Document$_{80\% \text{ reg}}$	15.01 ↓ -1.34	26.37 ↓ -2.61	42.53 ↓ -4.16	38.32 ↓ -2.32	73.69 ↓ -0.28	18.68 ↑ 2.14	42.22 ↓ -1.90
Query$_{\text{reg}}$ $\circ$ Document$_{80\% \text{ reg}}$	15.92	27.32 ↓ -1.66	45.70 ↓ -0.99	39.59 ↓ -1.05	73.86	18.54 ↑ 2.00	43.96
Instr$_{\text{reg}}$	16.50	27.81 ↓ -1.17	46.68	**40.71**	**75.60** ↑ 1.63	19.71 ↑ 3.17	**45.85** ↑ 1.73
Query$_{\text{reg}}$ $\circ$ Instr$_{\text{reg}}$	16.06	28.25 ↓ -0.73	**47.61** ↑ 0.92	39.88 ↓ -0.76	75.43 ↑ 1.46	22.14 ↑ 5.60	45.52 ↑ 1.40
			RoBERTa				
Pretrained	11.96	21.45	34.77	30.68	66.86	8.56	34.14
Contrastive Fine-tuning	14.75	25.95	43.32	37.68	73.55	**20.79**	41.67
Promptagator	13.03	24.50	42.88	36.65	72.23	16.82	40.19
Document$_{40\% \text{ reg}}$	12.54	24.94	44.89 ↑ 2.01	36.86	72.94 ↑ 0.71	18.16 ↑ 1.34	41.65 ↑ 1.46
Query$_{\text{reg}}$ $\circ$ Document$_{40\% \text{ reg}}$	14.66 ↑ 1.63	27.58 ↑ 3.08	47.41 ↑ 4.53	38.54 ↑ 1.89	74.04 ↑ 1.81	18.81 ↑ 1.99	42.45 ↑ 2.26
Document$_{60\% \text{ reg}}$	14.11 ↑ 1.08	25.80 ↑ 1.30	43.99 ↑ 1.11	37.03	72.94 ↑ 0.71	16.84	41.77 ↑ 1.58
Query$_{\text{reg}}$ $\circ$ Document$_{60\% \text{ reg}}$	14.75 ↑ 1.72	28.34 ↑ 3.84	47.25 ↑ 4.37	39.43 ↑ 2.78	74.61 ↑ 2.38	20.48 ↑ 3.66	43.33 ↑ 3.14
Document$_{80\% \text{ reg}}$	14.47 ↑ 1.44	25.53 ↑ 1.03	42.47	37.57 ↑ 0.92	72.61 ↑ 0.38	15.10 ↓ -1.72	41.55 ↑ 1.36
Query$_{\text{reg}}$ $\circ$ Document$_{80\% \text{ reg}}$	**15.84** ↑ 2.81	**27.96** ↑ 3.46	46.96 ↑ 4.08	39.09 ↑ 2.44	**74.71** ↑ 2.48	20.49 ↑ 3.67	**43.41** ↑ 3.22
Instr$_{\text{reg}}$	13.80 ↑ 0.77	25.00	45.78 ↑ 2.90	36.29	73.53 ↑ 1.30	18.24 ↑ 1.42	41.87 ↑ 1.68
Query$_{\text{reg}}$ $\circ$ Instr$_{\text{reg}}$	15.20 ↑ 2.17	26.88 ↑ 2.38	**47.44** ↑ 4.56	**39.38** ↑ 2.73	74.08 ↑ 1.85	19.31 ↑ 2.49	42.52 ↑ 2.33
			SPECTER-v2				
Pretrained	13.59	25.63	41.92	35.91	71.46	21.74	38.84
Contrastive Fine-tuning	14.56	25.18	**45.64**	36.76	71.46	20.49	40.45
Promptagator	14.13	23.67	39.22	36.70	71.55	22.37	39.10
Document$_{40\% \text{ reg}}$	15.39 ↑ 1.26	25.15 ↑ 1.48	43.39 ↑ 4.17	36.89	72.11 ↑ 0.56	21.66 ↓ -0.71	40.48 ↑ 1.38
Query$_{\text{reg}}$ $\circ$ Document$_{40\% \text{ reg}}$	14.19	24.99 ↑ 1.32	42.66 ↑ 3.44	36.39	71.49	21.40 ↓ -0.97	39.77 ↑ 0.67
Document$_{60\% \text{ reg}}$	15.41 ↑ 1.28	26.79 ↑ 3.12	44.19 ↑ 4.97	36.90	72.81 ↑ 1.26	21.35 ↓ -1.02	41.07 ↑ 1.97
Query$_{\text{reg}}$ $\circ$ Document$_{60\% \text{ reg}}$	14.95 ↑ 0.82	25.81 ↑ 2.14	43.30 ↑ 4.08	38.38 ↑ 1.68	72.19 ↑ 0.64	21.60 ↓ -0.77	40.55 ↑ 1.45
Document$_{80\% \text{ reg}}$	**17.02** ↑ 2.89	**28.63** ↑ 4.96	44.97 ↑ 5.75	**38.35** ↑ 1.65	**73.94** ↑ 2.39	**22.78** ↑ 0.41	**42.37** ↑ 3.27
Query$_{\text{reg}}$ $\circ$ Document$_{80\% \text{ reg}}$	14.70 ↑ 0.57	26.85 ↑ 3.18	44.66 ↑ 5.44	37.61 ↑ 0.91	73.07 ↑ 1.52	22.65	41.70 ↑ 2.60
Instr$_{\text{reg}}$	14.73 ↑ 0.60	25.82 ↑ 2.15	41.50 ↑ 2.28	35.91 ↓ -0.79	73.06 ↑ 1.51	21.05 ↓ -1.32	39.24 ↑ 0.14
Query$_{\text{reg}}$ $\circ$ Instr$_{\text{reg}}$	14.50 ↑ 0.37	25.32 ↑ 1.65	41.44 ↑ 2.22	36.01 ↓ -0.69	72.46 ↑ 0.91	22.26	39.19

Table 4: DORIS-MAE Full Experimental Results

Model (Method)	R@5	R@10	R@20	RP	NDCG@10	MRR@10	MAP
E5-Large-v2							
Pretrained	59.37	76.74	88.16	23.33	47.75	38.96	39.84
Contrastive Fine-tuning	70.69	85.86	94.77	29.34	56.15	47.13	47.57
Promptagator	69.41	83.35	94.11	30.38	55.14	46.67	47.24
Document$_{40\%\ reg}$	**78.72** ↑ 9.31	90.43 ↑ 7.08	**96.46** ↑ 2.35	39.40 ↑ 9.02	64.24 ↑ 9.10	56.45 ↑ 9.78	56.46 ↑ 9.22
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	77.76 ↑ 8.35	88.89 ↑ 5.54	96.21 ↑ 2.10	38.20 ↑ 7.82	63.00 ↑ 7.86	55.24 ↑ 8.57	55.38 ↑ 8.14
Document$_{60\%\ reg}$	78.16 ↑ 8.75	**90.85** ↑ 7.50	96.41 ↑ 2.30	39.63 ↑ 9.25	**64.41** ↑ 9.27	**56.57** ↑ 9.90	**56.53** ↑ 9.29
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	78.34 ↑ 8.93	90.52 ↑ 7.17	96.37 ↑ 2.26	39.06 ↑ 8.68	64.10 ↑ 8.96	56.24 ↑ 9.57	56.24 ↑ 9.00
Document$_{80\%\ reg}$	77.37 ↑ 7.96	89.81 ↑ 6.46	95.57 ↑ 1.46	38.14 ↑ 7.76	63.14 ↑ 8.00	55.16 ↑ 8.49	55.21 ↑ 7.97
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	78.44 ↑ 9.03	90.82 ↑ 7.47	96.25 ↑ 2.14	**39.69** ↑ 9.31	64.36 ↑ 9.22	56.49 ↑ 9.82	56.45 ↑ 9.21
Instr$_{reg}$	72.06 ↑ 2.65	86.23 ↑ 2.88	95.70 ↑ 1.59	32.14 ↑ 1.76	57.76 ↑ 2.62	49.17 ↑ 2.50	49.58 ↑ 2.34
Query$_{reg}$ ∘ Instr$_{reg}$	71.59 ↑ 2.18	85.12 ↑ 1.77	95.18 ↑ 1.07	31.65 ↑ 1.27	56.89 ↑ 1.75	48.37 ↑ 1.70	48.86 ↑ 1.62
SimCSE							
Pretrained	49.03	68.92	81.12	15.65	39.23	30.35	31.42
Contrastive Fine-tuning	36.70	51.22	64.79	11.18	28.85	22.12	23.57
Promptagator	69.49	84.45	92.54	31.91	56.38	47.97	48.36
Document$_{40\%\ reg}$	74.92 ↑ 5.43	86.37 ↑ 1.92	93.36 ↑ 0.82	35.54 ↑ 3.63	60.24 ↑ 3.86	52.42 ↑ 4.45	52.60 ↑ 4.24
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	**76.07** ↑ 6.58	**87.46** ↑ 3.01	**93.97** ↑ 1.43	**36.96** ↑ 5.05	**61.38** ↑ 5.00	**53.57** ↑ 5.60	**53.73** ↑ 5.37
Document$_{60\%\ reg}$	73.20 ↑ 3.71	84.71	92.10	33.47 ↑ 1.56	58.46 ↑ 2.08	50.58 ↑ 2.61	50.84 ↑ 2.48
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	75.75 ↑ 6.26	86.58 ↑ 2.13	93.19 ↑ 0.65	36.68 ↑ 4.77	60.94 ↑ 4.56	53.24 ↑ 5.27	53.44 ↑ 5.08
Document$_{80\%\ reg}$	70.38	81.60 ↓ -2.85	89.67 ↓ -2.87	31.42	55.76	47.97	48.39
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	74.10 ↑ 4.61	85.13	92.59	35.71 ↑ 3.80	59.79 ↑ 3.41	52.18 ↑ 4.21	52.45 ↑ 4.09
Instr$_{reg}$	70.97	86.14 ↑ 1.69	93.91 ↑ 1.37	32.18	57.41	48.80	49.14
Query$_{reg}$ ∘ Instr$_{reg}$	71.46 ↑ 1.97	86.34 ↑ 1.89	93.78 ↑ 1.24	33.18 ↑ 1.27	58.16 ↑ 1.78	49.78 ↑ 1.81	50.04 ↑ 1.68
RoBERTa							
Pretrained	30.51	40.56	49.68	9.84	23.56	18.49	19.54
Contrastive Fine-tuning	59.12	74.36	84.67	24.33	47.57	39.65	40.29
Promptagator	61.51	76.67	88.19	26.16	49.80	41.85	42.51
Document$_{40\%\ reg}$	67.38 ↑ 5.87	**81.87** ↑ 5.20	91.03 ↑ 2.84	30.26 ↑ 4.10	54.54 ↑ 4.74	46.43 ↑ 4.58	46.85 ↑ 4.34
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	67.29 ↑ 5.78	81.64 ↑ 4.97	**91.08** ↑ 2.89	29.21 ↑ 3.05	54.04 ↑ 4.24	45.85 ↑ 4.00	46.27 ↑ 3.76
Document$_{60\%\ reg}$	67.51 ↑ 6.00	81.78 ↑ 5.11	90.77 ↑ 2.58	**30.50** ↑ 4.34	**54.85** ↑ 5.05	**46.85** ↑ 5.00	**47.26** ↑ 4.75
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	67.71 ↑ 6.20	81.71 ↑ 5.04	90.99 ↑ 2.80	29.74 ↑ 3.58	54.41 ↑ 4.61	46.34 ↑ 4.49	46.73 ↑ 4.22
Document$_{80\%\ reg}$	**67.82** ↑ 6.31	80.97 ↑ 4.30	90.18 ↑ 1.99	30.47 ↑ 4.31	54.62 ↑ 4.82	46.79 ↑ 4.94	47.20 ↑ 4.69
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	67.38 ↑ 5.87	81.42 ↑ 4.75	90.70 ↑ 2.51	30.09 ↑ 3.93	54.53 ↑ 4.73	46.59 ↑ 4.74	46.97 ↑ 4.46
Instr$_{reg}$	64.47 ↑ 2.96	79.29 ↑ 2.62	90.09 ↑ 1.90	26.77 ↑ 0.61	51.49 ↑ 1.69	43.20 ↑ 1.35	43.78 ↑ 1.27
Query$_{reg}$ ∘ Instr$_{reg}$	65.14 ↑ 3.63	80.38 ↑ 3.71	90.26 ↑ 2.07	27.59 ↑ 1.43	52.25 ↑ 2.45	43.89 ↑ 2.04	44.39 ↑ 1.88
SPECTER-v2							
Pretrained	38.91	56.57	71.79	11.06	30.91	23.23	24.65
Contrastive Fine-tuning	41.90	59.65	75.70	12.25	33.20	25.26	26.75
Promptagator	38.64	58.34	75.15	9.59	30.24	21.77	23.40
Document$_{40\%\ reg}$	56.09 ↑ 17.45	74.12 ↑ 15.78	86.32 ↑ 11.17	21.02 ↑ 11.43	45.00 ↑ 14.76	36.30 ↑ 14.53	37.18 ↑ 13.78
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	51.28 ↑ 12.64	69.75 ↑ 11.41	83.33 ↑ 8.18	17.57 ↑ 7.98	40.59 ↑ 10.35	31.81 ↑ 10.04	32.96 ↑ 9.56
Document$_{60\%\ reg}$	58.87 ↑ 20.23	75.45 ↑ 17.11	86.93 ↑ 11.78	22.11 ↑ 12.52	46.55 ↑ 16.31	37.88 ↑ 16.11	38.69 ↑ 15.29
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	54.22 ↑ 15.58	71.81 ↑ 13.47	84.97 ↑ 9.82	19.35 ↑ 9.76	42.90 ↑ 12.66	34.21 ↑ 12.44	35.26 ↑ 11.86
Document$_{80\%\ reg}$	**61.02** ↑ 22.38	**76.84** ↑ 18.50	**87.48** ↑ 12.33	**24.39** ↑ 14.80	**48.68** ↑ 18.44	**40.24** ↑ 18.47	**40.94** ↑ 17.54
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	57.55 ↑ 18.91	74.67 ↑ 16.33	86.65 ↑ 11.50	22.00 ↑ 12.41	46.01 ↑ 15.77	37.44 ↑ 15.67	38.27 ↑ 14.87
Instr$_{reg}$	43.26 ↑ 4.62	63.89 ↑ 5.55	80.16 ↑ 5.01	12.95 ↑ 3.36	34.80 ↑ 4.56	26.07 ↑ 4.30	27.50 ↑ 4.10
Query$_{reg}$ ∘ Instr$_{reg}$	38.57	60.47 ↑ 2.13	77.12 ↑ 1.97	10.72 ↑ 1.13	31.72 ↑ 1.48	23.15 ↑ 1.38	24.69 ↑ 1.29

Table 5: ArguAna Full Experimental Results

Model (Method)	R@5	R@10	R@20	RP	NDCG@10	MRR@10	MAP
E5-Large-v2							
Pretrained	18.41	22.21	27.06	9.34	15.25	13.07	13.84
Contrastive Fine-tuning	37.25	43.79	49.85	24.29	33.16	29.87	30.74
Promptagator	41.53	48.34	54.66	27.23	37.07	33.56	34.43
Document$_{40\%\ reg}$	41.40	47.56 ↓ -0.78	53.69 ↓ -0.97	27.83 ↑ 0.60	37.01	33.72	34.56
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	45.50 ↑ 3.97	51.65 ↑ 3.31	58.08 ↑ 3.42	30.58 ↑ 3.35	40.50 ↑ 3.43	37.01 ↑ 3.45	37.86 ↑ 3.43
Document$_{60\%\ reg}$	44.15 ↑ 2.62	50.84 ↑ 2.50	55.69 ↑ 1.03	30.07 ↑ 2.84	39.64 ↑ 2.57	36.16 ↑ 2.60	36.93 ↑ 2.50
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	**46.23** ↑ 4.70	**52.61** ↑ 4.27	**59.76** ↑ 5.10	**31.75** ↑ 4.52	**41.58** ↑ 4.51	**38.13** ↑ 4.57	**39.02** ↑ 4.59
Document$_{80\%\ reg}$	28.30 ↓ -13.23	33.42 ↓ -14.92	39.85 ↓ -14.81	16.04 ↓ -11.19	24.05 ↓ -13.02	21.13 ↓ -12.43	22.07 ↓ -12.36
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	44.80 ↑ 3.27	51.04 ↑ 2.70	58.21 ↑ 3.55	30.87 ↑ 3.64	40.29 ↑ 3.22	36.94 ↑ 3.38	37.85 ↑ 3.42
Instr$_{reg}$	42.15 ↑ 0.62	48.33	53.02 ↓ -1.64	27.57	37.33	33.89 ↑ 0.33	34.67
Query$_{reg}$ ∘ Instr$_{reg}$	44.90 ↑ 3.37	50.38 ↑ 2.04	56.28 ↑ 1.62	29.07 ↑ 1.84	39.29 ↑ 2.22	35.79 ↑ 2.23	36.66 ↑ 2.23
SimCSE							
Pretrained	16.96	20.76	24.98	9.62	14.58	12.67	13.44
Contrastive Fine-tuning	26.78	31.74	37.15	15.73	23.11	20.42	21.28
Promptagator	30.97	37.35	44.26	18.57	27.15	23.99	24.95
Document$_{40\%\ reg}$	32.10 ↑ 1.13	37.43	43.07 ↓ -1.19	20.29 ↑ 1.72	28.30 ↑ 1.15	25.45 ↑ 1.46	26.34 ↑ 1.39
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	34.63 ↑ 3.66	40.71 ↑ 3.36	47.44 ↑ 3.18	22.03 ↑ 3.46	30.64 ↑ 3.49	27.51 ↑ 3.52	28.46 ↑ 3.51
Document$_{60\%\ reg}$	33.03 ↑ 2.06	38.09	43.96	20.47 ↑ 1.90	28.87 ↑ 1.72	25.97 ↑ 1.98	26.89 ↑ 1.94
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	34.84 ↑ 3.87	40.84 ↑ 3.49	47.08 ↑ 2.82	22.16 ↑ 3.59	30.77 ↑ 3.62	27.64 ↑ 3.65	28.58 ↑ 3.63
Document$_{80\%\ reg}$	19.75 ↓ -11.22	24.46 ↓ -12.89	29.89 ↓ -14.37	10.20 ↓ -8.37	16.73 ↓ -10.42	14.33 ↓ -9.66	15.20 ↓ -9.75
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	33.76 ↑ 2.79	39.17 ↑ 1.82	45.63 ↑ 1.37	20.71 ↑ 2.14	29.34 ↑ 2.19	26.26 ↑ 2.27	27.21 ↑ 2.26
Instr$_{reg}$	33.81 ↑ 2.84	39.23 ↑ 1.88	45.79 ↑ 1.53	21.61 ↑ 3.04	29.74 ↑ 2.59	26.79 ↑ 2.80	27.72 ↑ 2.77
Query$_{reg}$ ∘ Instr$_{reg}$	**37.34** ↑ 6.37	**43.37** ↑ 6.02	**50.45** ↑ 6.19	**24.57** ↑ 6.00	**33.18** ↑ 6.03	**30.02** ↑ 6.03	**30.99** ↑ 6.04
RoBERTa							
Pretrained	2.84	3.60	5.33	1.11	2.17	1.73	2.10
Contrastive Fine-tuning	23.59	29.09	35.27	13.80	20.69	18.10	19.05
Promptagator	27.23	33.07	39.14	15.83	23.69	20.78	21.74
Document$_{40\%\ reg}$	28.30 ↑ 1.07	34.47 ↑ 1.40	40.89 ↑ 1.75	16.94 ↑ 1.11	24.79 ↑ 1.10	21.81 ↑ 1.03	22.77 ↑ 1.03
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	32.83 ↑ 5.60	39.55 ↑ 6.48	46.38 ↑ 7.24	20.55 ↑ 4.72	29.05 ↑ 5.36	25.82 ↑ 5.04	26.79 ↑ 5.05
Document$_{60\%\ reg}$	28.67 ↑ 1.44	35.14 ↑ 2.07	41.49 ↑ 2.35	17.00 ↑ 1.17	25.14 ↑ 1.45	22.06 ↑ 1.28	23.03 ↑ 1.29
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	34.30 ↑ 7.07	**40.86** ↑ 7.79	**47.52** ↑ 8.38	**21.29** ↑ 5.46	**30.15** ↑ 6.46	**26.83** ↑ 6.05	**27.82** ↑ 6.08
Document$_{80\%\ reg}$	21.74 ↓ -5.49	27.23 ↓ -5.84	33.16 ↓ -5.98	12.11 ↓ -3.72	18.91 ↓ -4.78	16.35 ↓ -4.43	17.23 ↓ -4.51
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	**33.69** ↑ 6.46	39.74 ↑ 6.67	46.23 ↑ 7.09	20.69 ↑ 4.86	29.41 ↑ 5.72	26.20 ↑ 5.42	27.18 ↑ 5.44
Instr$_{reg}$	28.12 ↑ 0.89	34.36 ↑ 1.29	40.80 ↑ 1.66	16.28 ↑ 0.45	24.51 ↑ 0.82	21.46 ↑ 0.68	22.43 ↑ 0.69
Query$_{reg}$ ∘ Instr$_{reg}$	33.09 ↑ 5.86	39.65 ↑ 6.58	46.09 ↑ 6.95	20.57 ↑ 4.74	29.28 ↑ 5.59	26.06 ↑ 5.28	27.03 ↑ 5.29
SPECTER-v2							
Pretrained	2.98	4.22	5.26	0.97	2.43	1.87	2.18
Contrastive Fine-tuning	7.85	9.60	12.56	3.72	6.36	5.35	5.89
Promptagator	6.56	8.46	11.81	3.60	5.72	4.88	5.40
Document$_{40\%\ reg}$	9.98 ↑ 3.42	12.52 ↑ 4.06	16.12 ↑ 4.31	4.99 ↑ 1.39	8.55 ↑ 2.83	7.32 ↑ 2.44	7.92 ↑ 2.52
Query$_{reg}$ ∘ Document$_{40\%\ reg}$	10.84 ↑ 4.28	12.76 ↑ 4.30	16.26 ↑ 4.45	5.30 ↑ 1.70	8.81 ↑ 3.09	7.57 ↑ 2.69	8.18 ↑ 2.78
Document$_{60\%\ reg}$	8.64 ↑ 2.08	11.42 ↑ 2.96	14.49 ↑ 2.68	4.51 ↑ 0.91	7.46 ↑ 1.74	6.26 ↑ 1.38	6.80 ↑ 1.40
Query$_{reg}$ ∘ Document$_{60\%\ reg}$	7.54 ↑ 0.98	10.09 ↑ 1.63	13.29 ↑ 1.48	4.03 ↑ 0.43	6.55 ↑ 0.83	5.48 ↑ 0.60	6.01 ↑ 0.61
Document$_{80\%\ reg}$	7.39 ↑ 0.83	10.27 ↑ 1.81	13.19 ↑ 1.38	3.62	6.46 ↑ 0.74	5.31 ↑ 0.43	5.87 ↑ 0.47
Query$_{reg}$ ∘ Document$_{80\%\ reg}$	8.14 ↑ 1.58	11.28 ↑ 2.82	14.69 ↑ 2.88	4.43 ↑ 0.83	7.33 ↑ 1.61	6.13 ↑ 1.25	6.69 ↑ 1.29
Instr$_{reg}$	10.49 ↑ 3.93	13.30 ↑ 4.84	16.80 ↑ 4.99	6.05 ↑ 2.45	9.30 ↑ 3.58	8.07 ↑ 3.19	8.67 ↑ 3.27
Query$_{reg}$ ∘ Instr$_{reg}$	**11.08** ↑ 4.52	**13.83** ↑ 5.37	**17.36** ↑ 5.55	**6.09** ↑ 2.49	**9.59** ↑ 3.87	**8.28** ↑ 3.40	**8.91** ↑ 3.51

Table 6: WhatsThatBook Full Experimental Results