



# RePair: An Extensible Toolkit to Generate Large-Scale Datasets for Query Refinement via Transformers

Yogeswar Lakshmi Narayanan  
University of Windsor, Canada  
lakshmiy@uwindsor.ca

Hossein Fani  
University of Windsor, Canada  
hfani@uwindsor.ca

## ABSTRACT

Query refinement is the process of transforming users' queries into new *refined* versions without semantic drift to enhance the relevance of search results. Prior query refiners were benchmarked on web query logs following *weak* assumptions that users' input queries within a search session are about a single topic and improve gradually, which is not necessarily accurate in practice. In this paper, we contribute RePair, an open-source configurable toolkit to generate *large-scale* gold-standard benchmark datasets whose pairs of (original query, refined versions) are *almost surely* guaranteed to be in the same semantic context. RePair takes a dataset of queries and their relevance judgements (e.g., msmarco or ao1), a sparse or dense retrieval method (e.g., bm25 or colbert), and an evaluation metric (e.g., map or mrr), and outputs refined versions of queries, each of which with the relevance improvement guarantees under the retrieval method in terms of the evaluation metric. RePair benefits from text-to-text-transfer-transformer (t5) to generate gold-standard datasets for any input query sets and is designed with extensibility in mind. Out of the box, RePair includes gold-standard datasets for ao1 and msmarco.passage as well as benchmark results of state-of-the-art supervised query suggestion methods on the generated datasets at [github.com/fani-lab/RePair](https://github.com/fani-lab/RePair).

## CCS CONCEPTS

• Information systems → Query suggestion; Query reformulation.

## KEYWORDS

Gold Standard Dataset; Query Refinement; Reproducibility;

### ACM Reference Format:

Yogeswar Lakshmi Narayanan and Hossein Fani. 2023. RePair: An Extensible Toolkit to Generate Large-Scale Datasets for Query Refinement via Transformers. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23)*, October 21–25, 2023, Birmingham, United Kingdom. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3583780.3615129>

## 1 INTRODUCTION

The foremost means of information retrieval, search engines, have difficulty searching into knowledge repositories, e.g., the web, since

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
CIKM '23, October 21–25, 2023, Birmingham, United Kingdom  
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 979-8-4007-0124-5/23/10...\$15.00  
<https://doi.org/10.1145/3583780.3615129>

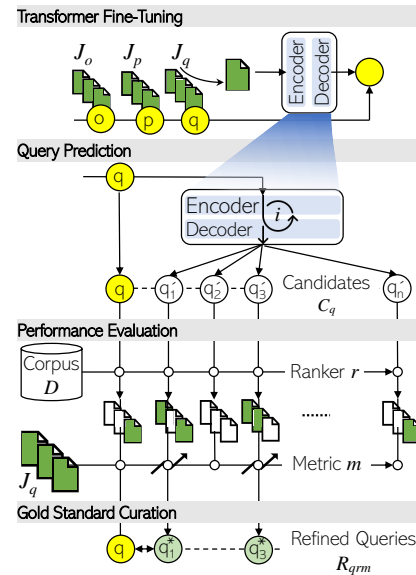


Figure 1: RePair's pipeline.

they are not tailored to the users' differing information needs. Formulating an effective query for an information retrieval system is challenging, even for experienced users, since it requires predicting which terms appear in documents relevant to the information need. Queries are under-specified or contain ambiguous terms that also retrieve *irrelevant* documents. Relieving the burden on the users, query refinement is the process of transforming the user's *original* query  $q$  into a newly *refined* version  $q^*$  that more accurately reflects the information need and enhances the relevance of search results.

Existing works can be divided into two categories: (1) query refinement methods and (2) benchmark datasets. On the one hand, we have supervised machine learning methods that predict refined versions of an original query [1, 3] given search session information such as user information [4], query time [6], and search history (query logs) [1]. For training and evaluation, such methods included web retrieval datasets like ao1 [12] or msmarco [10] following *weak* assumptions that users' input queries improve gradually within a search session, i.e., the last query where the user ends her search session is the refined version of her original query [3]. However, Tamannaee et al. [15] have shown potential *semantic (topic) drift* of queries in users' search session; a user might search for multiple topics in one session, and hence, irrelevant queries would be paired.

On the other hand, recently, we have observed new research efforts to produce benchmark datasets free of semantic drifts that are specifically designed to train and evaluate the efficacy of query refinement methods for web or non-web information retrieval systems [2, 15, 17]. Among the first, Tamannaee et al. [15] proposed

**Table 1: Comparison of existing systems for generating gold-standard datasets for query refinement.**

	open-source	toolkit	reproducible	scalable	dependencies	query sets	transformers	rankers	metrics	out of the box
ReQue [15]	✓	✓	✓	×	anserini pyserini trec_eval	any	×	sparse	trec_eval	robust04.bm25.map clueweb09.bm25.map clueweb12.bm25.map gov2.bm25.map
Arabzadeh et al. [2]	×	×	-	-	-	msmarco.passage	✓	bm25	map	msmarco.passage.bm25.map
RePair	✓	✓	✓	✓	pyserini trec_eval	any	✓	sparse dense	trec_eval	msmarco.passage.{bm25, colbert}.{map, mrr} aol.title.{bm25, colbert}.{map, mrr} aol.url.title.{bm25, colbert}.{map, mrr}

ReQue<sup>1</sup>, an open-source configurable and reproducible pipeline to control the semantic drift and generate a *gold*-standard dataset from an input set of original queries along with their relevance judgements. ReQue applies a host of unsupervised query refiners, from simple lexical lemmatizers to complex pseudo-relevance-based methods on the set of original queries. Then, those revised versions of the original query that improve the performance of a retrieval method (e.g., bm25) in terms of an evaluation metric (e.g., map) using the query’s relevance judgment  $\mathcal{J}_q$  were kept as the refined queries  $\mathcal{R}_q = \{q^*\}$ . This way, both the original query and the refined versions  $\mathcal{R}_q$  are *almost surely* guaranteed to be in the same semantic context. However, ReQue is computationally costly for large-scale query sets such as aol or msmarco due to its exhaustive application of all refinement methods on each query. To address scalability, Arabzadeh et al. [2] applied Nogueira et al.’s pretrained doct5query [11] transformer and generated refined questions (queries) for msmarco’s passage dataset for a question-answering task, known as msmarco.passage. Arabzadeh et al. fed an original question to the doct5query and selected the generated sequence of tokens as a refined version should it increase the bm25 retrieval performance based on the map metric. Arabzadeh et al.’s work is, however, case-specific, considerably less extensible, and heavily depends on doct5query; it is incapable of accommodating different or new datasets, let alone no reproducible implementation is available; only the final generated dataset is released.

In this paper, like ReQue [15], we propose a principled domain-agnostic pipeline to generate refined queries while withholding the same semantic context between (original query  $\rightarrow$  refined query) pairs. We contribute RePair, an open-source and configurable standard toolkit to support supervised query refinement research with large-scale gold-standard datasets from a variety of domains. As shown in Figure 1, RePair takes: (1) a query set  $Q = \{q\}$  along with relevance judgements  $\mathcal{J} = \{\mathcal{J}_q : q \in Q\}$  and optional contextual information such as time or user information  $X = \{X_q : q \in Q\}$ , (2) an information retrieval method (ranker),  $r$ , and (3) an evaluation metric,  $m$ , and outputs a gold-standard dataset  $\mathcal{R}_m = \{\mathcal{R}_{qrm}\}$  that includes  $\mathcal{R}_{qrm} = \{q^*\}$  for each of the queries in the input query set  $Q$  such that  $q^* \in \mathcal{R}_{qrm}$  retrieves better search results under the ranker  $r$  and evaluation metric  $m$ .

Table 1 compares RePair with related works. In contrast to existing systems, especially when scalability, reproducibility, and extensibility are of prime concerns, RePair employs an object-oriented design that is flexible to customization and smoothly incorporates a new query set as well as a new transformer along with parallel

```

# ./src/main.py
from dal.msmarco import *
from mdl.mt5w import *
def run(cmd=['pair', 'finetune', 'predict', 'search', 'eval', 'agg'],...)
    if 'msmarco.passage' in domain_list:
        if 'pair' in cmd: msmarco.to_pair(pairing, concat=True)
        if 'finetune' in cmd: finetune(...)
        if 'predict' in cmd: predict(...)
        if 'search' in cmd: search(...)
        if 'eval' in cmd: evaluate(...)
        if 'agg' in cmd: agg(...)
# ./src/dal/ds.py
def pair(datapath, Qfilename, concat=True):
def search(queries, ranker, topk, batch, ...):
def aggregate(original, changes, metrics, ...)
def box(original, changes, metrics, ...)
# ./src/dal/msmarco.py
@override
def pair(datapath, Qfilename, concat=True):
# ./src/dal/aol.py
@override
def pair(datapath, Qfilename, concat=True):
# ./src/mdl/mt5w.py => t5 wrapper
def finetune(pretrained_dir, in_type='query', out_type='doc', ...):
def predict(iter, output, vocab_model_path, gcloud=False):
# ./src/evl/trecw.py => trec_eval wrapper
def evaluate(in_docids, out_metrics, qrels, metric, topk=10, ...):

```

**Figure 2: RePair’s driver.**

processing of large-scale datasets. Community members can use RePair to generate new gold-standard datasets for any input query sets and their associated relevance judgements. The codebase, tutorials, and case studies on msmarco.passage and aol can be obtained with cc-by-nc-sa-4.0 license at [github.com/fani-lab/RePair](https://github.com/fani-lab/RePair).

## 2 SYSTEM OVERVIEW

Figure 1 demonstrates the RePair’s components and workflow with the driver code shown in Figure 2, which are explained hereafter.

### 2.1 Transformer Fine-Tuning [pair, finetune]

In RePair, we have implemented a flexible component for training or fine-tuning a transformer  $\tau$  based on any pairing strategies of queries and relevant documents, i.e.,  $(\mathcal{J}_q \rightarrow q)$  or  $(q \rightarrow \mathcal{J}_q)$ , and in case there are more than one relevant documents for a query,  $|\mathcal{J}_q| > 1$ , whether to replicate the pairs (default) or to concatenate  $(\mathcal{J}_q^+)$  the relevant documents as a single document. Training or fine-tuning can happen on subparts or the entire text body of the relevant documents as in aol where documents are webpages including url, title, and text. RePair can augment the query’s contextual information such as the user or time information as a pretext to

<sup>1</sup>[github.com/fani-lab/ReQue](https://github.com/fani-lab/ReQue)

```

# ./src/param.py
'box':{
'gold': 'refined_q_metric >= original_q_metric and refined_q_metric > 0',
'platinum': 'refined_q_metric > original_q_metric',
'diamond': 'refined_q_metric > original_q_metric and refined_q_metric == 1'
'new_criteria': '(refined_q_metric - original_q_metric) >= +0.2'}

```

Figure 3: Selection criteria for being a refined query in eq.1.

the input of the transformer as  $(\mathcal{X}_q : \mathcal{J}_q \rightarrow q)$  or  $(\mathcal{X}_q : q \rightarrow \mathcal{J}_q)$ . The configuration parameters in this component are the original query set  $Q = \{q\}$ , relevant documents of queries  $\mathcal{J} = \{\mathcal{J}_q\}$ , paring strategy  $(\mathcal{J}_q \rightarrow q)$  or  $(q \rightarrow \mathcal{J}_q)$  as well as to replicate or concatenate pairs when  $|\mathcal{J}_q| > 1$  and optional  $\mathcal{X}_q$ .

## 2.2 Refined Query Prediction [predict]

The purpose of this component is to predict a set of candidate queries  $C_q = \{q'\}$ , each of which has the potential to serve as a refined query for an original query  $q$ . The configuration parameter in this component is a trained or fine-tuned transformer  $\tau$  and an input to the transformer that could be a query  $q$  or its relevant documents  $\mathcal{J}_q$  with an optional context  $\mathcal{X}_q$ . We feed input to  $\tau$  and the output prediction is considered as a candidate query  $q'$ ; notationally,  $\tau(\cdot) = q'$ . Modern transformers apply top- $k$  random selection [5] at their decoders as opposed to beam search [14] to generate novel outputs and avoid common phrases and repetitive text. Top- $k$  random selection yields non-deterministic output generation during inference given the same input, i.e.,  $\tau_i(\cdot) = q'_i$  that can be employed to generate a collection of candidate queries  $C_q = \{q'_i\}_{i=1}^k$ , as opposed to a single candidate query  $C_q = q'$ .

## 2.3 Performance Evaluation

This component of RePair has two subcomponents:

**2.3.1 Relevant Documents Retrieval [search].** Given the candidate queries  $C_q$  generated by the refined query prediction component, RePair searches for the relevant documents for both the original query  $q$  and each of the candidate queries  $q' \in C_q$  from the corpus. Hence, RePair has two configuration parameters: the corpus  $\mathcal{D}$  and an information retrieval method, called ranker  $r$ , that retrieves relevant documents and ranks them based on relevance scores. RePair integrates `pyserini` [8], which provides efficient implementations of sparse and dense rankers, including `bm25`, `qld`, and `colbert` [7]. As seen in Figure 1, RePair’s output in this component is a ranked list of documents retrieved by the ranker  $r$  for the original query  $q$  as well as each of the candidate queries  $q'$ .

**2.3.2 Retrieval Evaluation [eval].** Given an original query  $q$  and its relevance judgements  $\mathcal{J}_q$ , i.e., true ranked list of relevant documents for  $q$ , RePair evaluates each of its candidates queries  $q' \in C_q$  based on how they improve the performance of ranker  $r$  with respect to an evaluation metric  $m$ . Then, a candidate  $q'$  that provides performance improvements compared to the original query  $q$  is selected as the refined query  $q^* \in \mathcal{R}_{qrm}$  for that original query under the ranker  $r$  and metric  $m$ . As a result, this component has a single configuration parameter; an evaluation metric  $m$ . Formally, a list of refined queries  $\mathcal{R}_{qrm}$  is computed based on:

$$\mathcal{R}_{qrm} = \{q^* = q' \in C_q | r_m(q' : \mathcal{J}_q) > r_m(q : \mathcal{J}_q)\} \quad (1)$$

qid	order	query	bm25.map
0cc411681d1441	-1	staple com	0.037
0cc411681d1441	pred.8	staple pubs	1.0
0cc411681d1441	pred.7	staple england pub	0.5
0cc411681d1441	pred.1	staple east of england	0.1
0cc411681d1441	pred.10	staple	0.0385
0cc411681d1441	pred.3	staple england	0.0385

Figure 4: RePair’s gold-standard dataset file structure.

Table 2: Stats on `msmarco.passage` and `aol` query sets.

	$ Q $	avg $ q $	$ \mathcal{D} $	avg $ \mathcal{J}_q $	#users
<code>msmarco.passage</code>	502,939	5.9675	8,841,823	61.6940	-
<code>aol.title<sup>2</sup></code>	4,459,613	3.5849	1,473,341		
<code>aol.url.title</code>	4,672,506	3.5817	1,525,586	21.8372	657,426

where  $r_m(\cdot : \mathcal{J}_q)$  is the performance of the ranker  $r$  given the relevance judgement  $\mathcal{J}_q$  measured by the evaluation metric  $m$ . To calculate  $r_m(\cdot : \mathcal{J}_q)$ , RePair internally uses `trec_eval` which provides the standard implementation of information retrieval metrics such as `map`, `mrr`, and `ndcg`. In RePair, we crown a refined query  $q^*$  as an *oracle* query  $\overset{\circlearrowleft}{q}$  if it achieves the maximum retrieval performance, i.e.,  $r_m(\overset{\circlearrowleft}{q}) = 1$ . More interestingly, since RePair stores the performances for all the candidate queries in files, we can customize eq.1 or include multiple selection criteria, as shown in Figure 3.

## 2.4 Gold-Standard Dataset Curation [agg]

Given the original query set  $Q = \{q\}$ , RePair aggregates the evaluation results  $\mathcal{R}_{qrm}$  for  $q \in Q$  based on eq. 1, and finally stores the gold-standard dataset  $\mathcal{R}_{rm}$  in `Q.r.m.agg.gold.tsv` file, each entry of which includes:

- (1) `qid`: id of the original query  $q \in Q$ ;
- (2) `order`: -1 if query is an original query, else `pred.i` as the  $i$ -th candidate query that becomes a refined query in *descending* order of metric values;
- (3) `query`: text body of the original query or the refined query;
- (4) `r.m`: performance of query under the ranker  $r$  and metric  $m$ .

For instance, the gold-standard dataset for `aol.title` using the retrieval method `bm25` and based on the evaluation metric `map` is stored in `aol.title.bm25.map.agg.gold.tsv` and few of its entries are shown in Figure 4. As seen, for the original query ‘`staple com`’, the retrieval performance is  $0.037$  while its best refined query ‘`staple pubs`’ is an oracle with a `map=1.0`.

## 3 CASE STUDY

Out of the box, RePair includes gold-standard datasets for `aol` [12] and `msmarco.passage` [10] query sets based on `bm25` (sparse) and `colbert` (dense) as the ranker, and `mrr` and `map` as the evaluation metrics. Table 2 summarizes these datasets. For `msmarco.passage`, we selected the training part. We set the pairing strategy to  $(\mathcal{J}_q^+ \rightarrow q)$  where  $\mathcal{J}_q^+$  is the concatenation of all the relevant passages if  $|\mathcal{J}_q| > 1$ . Also, `msmarco.passage` has no contextual information,  $\mathcal{X}_q = \emptyset$ . For `aol`, we used the reconstructed version of `aol` by Sean et al. [9], which includes users’ url clicks as relevance judgements as well as the crawled at-the-time (2006) urls’ webpages in triples (`url`, `title`, `text`). We built two corpora for `aol` whose documents are either i) `title`, or ii) concatenation of `url` and `title`, i.e., `url.title` of webpages. An important aspect of `aol` is the query’s

<sup>2</sup>While a webpage has `url`, it may be missing `title`. In `aol.title`, we filter out queries whose relevant webpages have no `title`.

**Table 3: Sample queries paired with oracle query ( $q^{\dagger}$ ), refined query ( $q^*$ ), or none ( $\mathcal{R}_{qrm} = \emptyset$ ) for  $r = \text{colbert}$  and  $m = \text{map}$ .**

	qid	$q$	$\text{colbert}_{\text{map}}(q)$	$q^*$	$\text{colbert}_{\text{map}}(q^*)$
msmarco.passage	100310	cost analyst accountant responsibilities	0.5000	what's the job description for a cost analyst	<b>1.0000</b>
	1001344	where to file 941-pr without payment	0.5714	where is 941 form filed in texas	0.6429
	100161	cortex definition	<b>1.0000</b>	where is the cortex in term	0.0108
aol.title	ffb1ffc470a8df	free kids activities	0.1111	free kids crafts	<b>1.0000</b>
	ff2b7cf89c900c	helping parents handle aggressive behavior in children	0.0345	child behavior conflicts	0.1250
	0026104dd77f50	virginia office attorney general	0.5000	free credit reports simon cowell	0.0128
aol.url.title	0511a70716628f	risks of vaginal obstetrical ultrasound	0.3300	ultrasound assisted pregnancy test	1.0000
	0598a9db866c6e	best sailor	0.1660	godfather in sailor moon fanfiction	0.5000
	4001d7502818af3	starck realtor	1.0000	real estate chicagoland	0.0200

**Table 4: Stats on gold-standard datasets for  $r = \text{bm25}$ .**

	$\text{avg } r_m(q)$	$ R $	$\text{avg }  q^* $	%	$\text{avg } r_m(q^*)$	$\Delta\%$	$\#q^{\dagger}$	%
$m = \text{map}$								
msmarco.passage	0.0862	414,337	7.4419	82%	0.5704	+562%	176,922	35%
aol.title	0.0252	2,583,023	3.1270	58%	0.4175	+1,556%	649,764	14%
aol.url.title	0.0271	2,421,347	3.5354	52%	0.3997	+1,374%	591,001	13%
$m = \text{mrr}$								
msmarco.passage	0.1795	472,553	7.3389	93%	0.6137	+593%	185,414	36%
aol.title	0.1967	2,276,965	3.0543	51%	0.7497	+281%	1,069,531	23%
aol.url.title	0.1754	2,845,642	3.4778	60%	0.6063	+245%	1,037,103	22%

contextual information  $X_q$  such time and userid, which can be employed for personalized or temporal query refinement [16, 18].

We selected t5-base transformer with 220 million parameters as the transformer  $\tau$  and fine-tuned it for 4,000 epochs. At the refined query prediction step, we used top-10 random sampling decoder and predicted 10 candidate queries, i.e.,  $|C_q| = 10$ , by running  $\tau_i(\mathcal{J}_q) = q'_i$  for  $1 \leq i \leq 10$ . Statistics, including the average size of  $q^*$  and the average mrr and map improvement rate for each of these gold-standard datasets have been reported in Table 4 for bm25 (sparse) and Table 5 for colbert (dense).

**Sparse Retrieval:** Table 4 demonstrates that RePair could improve more than half of the original queries for msmarco.passage and all variations of aol with respect to map. The largest gold-standard dataset is associated with aol.title with about 2.6M pairs of  $(q \rightarrow q^*)$ , which is 58% of original queries with an average map increase from 0.025 to 0.417, around +1,500% increase. However, the richest one is msmarco.passage where about 82% of the original queries have been improved in terms of map from 0.086 to 0.570. Notably, in the gold-standard datasets on msmarco.passage and aol, more than 35% and 13% of queries have been paired with oracle refined queries with  $\text{map} = 1.0$ .

**Dense Retrieval** On the one hand, although RePair supports colbert as a dense information retrieval method, like other dense retrieval methods, it is tied to extreme resource consumption to run it on the entire set of candidate queries for msmarco.passage and aol query sets. On the other hand, although the gold-standard datasets for these query sets using bm25 include a substantial amount of pairs, there are original queries  $q$  none of their candidate queries  $q'$  were selected as  $q^*$  due to them being *hard* for bm25 to fetch relevant documents and improve the evaluation metrics such as mrr and map, that is,  $\mathcal{R}_{qrm} = \emptyset$  for  $r = \text{bm25}$ . In msmarco.passage, there are 16,841 and 16,227 original queries where bm25 was unable to improve map and mrr, respectively. In aol.title and aol.url.title, the numbers are 159,670 and 210,581 based on map and 104,715 and 206,119 based on mrr. To show RePair's colbert feature and its feasibility to improve upon bm25 while saving computational cost, we ran RePair with colbert for all original queries with no refined queries with respect to map for msmarco.passage (16,841 queries). For aol.title and aol.url.title, we randomly sampled the

**Table 5: colbert's improvement on bm25.map hard queries.**

	$\text{bm25}_{\text{map}}(q)$	$\text{colbert}_{\text{map}}(q)$	$\text{bm25}_{\text{map}}(q^*)$	$\text{colbert}_{\text{map}}(q^*)$	$\Delta+$
msmarco.passage	0.3729	0.4237	0.1612	0.3366	4,580
aol.title	0.2657	0.2165	0.0727	0.0970	3,355
aol.url.title	0.2310	0.1894	0.0588	0.0691	2,715

**Table 6: Supervised methods' performance.**

gold-standard dataset	model	rouge-1	bleu	f1
msmarco.passage.bm25.map	t5-base	<b>42.96</b>	21.15	<b>41.91</b>
	acg [3]	37.79	<b>21.66</b>	39.37
	hred-qs [13]	31.07	16.78	31.29
aol.title.bm25.map	t5-base	<b>15.55</b>	<b>6.25</b>	<b>14.42</b>
	acg [3]	11.20	4.41	11.87
	hred-qs [13]	6.05	3.79	6.30
aol.url.title.bm25.map	t5-base	<b>22.13</b>	<b>8.73</b>	<b>20.73</b>
	acg [3]	11.21	3.86	11.94
	hred-qs [13]	6.72	3.60	7.03

same number of hard queries. From Table 5, colbert could provide additional refined queries to hard queries in msmarco.passage and aol, e.g., +4,580 refined queries to msmarco.passage's hard queries. Table 3 shows sample queries paired with oracle queries ( $q^{\dagger}$ ), refined queries ( $q^*$ ), or none ( $\mathcal{R}_{qrm} = \emptyset$ ) for colbert as the ranker and map as the metric.

#### 4 BENCHMARKS ON REPAIR'S DATASETS

We employed supervised query suggestion methods, including pre-trained t5-base, acg [3], and hred-qs [13], to demonstrate a benchmarking sample on RePair's generated gold-standard datasets. Given the gold-standard datasets generated using  $r = \text{bm25}$  and  $m = \text{map}$ , we trained all models on 70% of  $(q \rightarrow q^*)$  pairs for 100 epochs. The models were evaluated on the remaining 30% pairs as the test set using rouge-1, bleu, and f1-measure. From Table 6, t5-base outperformed hred-qs and acg for msmarco.passage.bm25.map and aol.\*. We also observe a performance drop for the baselines for aol.\* which is attributed to the small query sizes; the average query size in msmarco.passage is 7.4419 tokens whereas aol.title and aol.url.title have queries with average sizes of 3.1270 and 3.5354 tokens, respectively.

#### 5 CONCLUSION AND FUTURE WORK

We presented RePair, an open-source toolkit for query refinement research. RePair features (1) reproducible, extensible, and domain-agnostic pipeline for generating gold-standard datasets while controlling semantic drifts, (2) easy reconfiguration for new query sets in web or non-web domains with the ability to incorporate additional search session information, (3) modular design that is flexible to customization and integration of new transformers, retrieval methods, or evaluation metrics, and (4), out of the box, large-scale gold-standard datasets for aol and msmarco-passage query sets. Future directions include extensions to personalized and time-aware datasets for query refinement, and hybrid retrieval methods.

## REFERENCES

- [1] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. 2019. Context Attentive Document Ranking and Query Suggestion. In *42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*. 385–394.
- [2] Negar Arabzadeh, Amin Bigdeli, Shirin Seyedsalehi, Morteza Zihayat, and Ebrahim Bagheri. 2021. Matches Made in Heaven: Toolkit and Large-Scale Datasets for Supervised Query Reformulation. In *CIKM '21: The 30th International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). ACM, 4417–4425. <https://doi.org/10.1145/3459637.3482009>
- [3] Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to Attend, Copy, and Generate for Session-Based Query Suggestion. In *2017 ACM on Conference on Information and Knowledge Management*. 1747–1756.
- [4] Pierre Erbacher, Ludovic Denoyer, and Laure Soulier. 2022. Interactive Query Clarification and Refinement via User Simulation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 2420–2425. <https://doi.org/10.1145/3477495.3531871>
- [5] Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 889–898. <https://doi.org/10.18653/v1/P18-1082>
- [6] Jiafeng Guo, Gu Xu, Hang Li, and Xueqi Cheng. 2008. A unified and discriminative model for query refinement. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong (Eds.). ACM, 379–386. <https://doi.org/10.1145/1390334.1390400>
- [7] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 39–48. <https://doi.org/10.1145/3397271.3401075>
- [8] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 2356–2362. <https://doi.org/10.1145/3404835.3463238>
- [9] Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2022. Reproducing Personalised Session Search Over the AOL Query Log. In *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part 1 (Lecture Notes in Computer Science, Vol. 13185)*, Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty (Eds.). Springer, 627–640. [https://doi.org/10.1007/978-3-030-99736-6\\_42](https://doi.org/10.1007/978-3-030-99736-6_42)
- [10] 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. In *NIPS 2016*. [http://ceur-ws.org/Vol-1773/CoCoNIPS\\_2016\\_paper9.pdf](http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf)
- [11] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* 6 (2019).
- [12] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st international conference on Scalable information systems*. 1–es.
- [13] Alessandro Sordani, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A Hierarchical Recurrent Encoder-Decoder for Generative Context-Aware Query Suggestion. In *CIKM 2015*. ACM, 553–562.
- [14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (Eds.). 3104–3112. <https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html>
- [15] Mahtab Tamannaee, Hossein Fani, Fattane Zarrinkalam, Jamil Samouh, Samad Paydar, and Ebrahim Bagheri. 2020. ReQue: A Configurable Workflow and Dataset Collection for Query Refinement. In *CIKM2020*. ACM, 3165–3172. <https://doi.org/10.1145/3340531.3412775>
- [16] Thanh Vu, Alistair Willis, Udo Kruschwitz, and Dawei Song. 2017. Personalised Query Suggestion for Intranet Search with Temporal User Profiling. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval, CHIIR 2017, Oslo, Norway, March 7-11, 2017*, Ragnar Nordlie, Nils Pharo, Luanne Freund, Birger Larsen, and Dan Russel (Eds.). ACM, 265–268. <https://doi.org/10.1145/3020165.3022129>
- [17] George Zerveas, Ruochen Zhang, Leila Kim, and Carsten Eickhoff. 2020. Brown University at TREC Deep Learning 2019. *CoRR abs/2009.04016* (2020). [arXiv:2009.04016](https://arxiv.org/abs/2009.04016) <https://arxiv.org/abs/2009.04016>
- [18] Jianling Zhong, Weiwei Guo, Huiji Gao, and Bo Long. 2020. Personalized Query Suggestions. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1645–1648. <https://doi.org/10.1145/3397271.3401331>