WTR: A Test Collection for Web Table Retrieval

Zhiyu Chen
zhc415@lehigh.edu
Lehigh University
Bethlehem, PA, USA

Shuo Zhang
szhang611@bloomberg.net
Bloomberg
London, United Kingdom

Brian D. Davison
davison@cse.lehigh.edu
Lehigh University
Bethlehem, PA, USA

ABSTRACT

We describe the development, characteristics and availability of a test collection for the task of Web table retrieval, which uses a large-scale Web Table Corpora extracted from the Common Crawl. Since a Web table usually has rich context information such as the page title and surrounding paragraphs, we not only provide relevance judgments of query-table pairs, but also the relevance judgments of query-table context pairs with respect to a query, which are ignored by previous test collections. To facilitate future research with this benchmark, we provide details about how the dataset is pre-processed and also baseline results from both traditional and recently proposed table retrieval methods. Our experimental results show that proper usage of context labels can benefit previous table retrieval methods.

CCS CONCEPTS

• **Information systems** → **Structured text search**; **Test collections**; **Relevance assessment**.

KEYWORDS

datasets, table search, dataset search

ACM Reference Format:
Zhiyu Chen, Shuo Zhang, and Brian D. Davison. 2021. WTR: A Test Collection for Web Table Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3404835.3463260

1 INTRODUCTION

Nowadays, tables have been used in various research tasks such as table-based question answering [6, 27], table-to-text generation [7, 19], column type annotation [8, 15, 35], table retrieval and so on. Among those tasks, table retrieval has obtained much attention recently in the IR community, witnessed by unsupervised methods [9, 31, 39], semantic feature-based methods [37, 39] and neural methods [10, 25, 29, 30]. It is concerned with the task of retrieving a set of tables from a table corpus and ranking them in descending order based on relevance score to a keyword query. Each table is a set of cells arranged in rows and columns like a matrix. A cell of a table could be a single word, a real number, a phrase, or even a sentence. Depending on the source of a dataset, a table has associated context fields. For example, the tables from Wikipedia have a caption, page title and section title, while an arbitrary Web table might have not a section title but surrounding sentences immediately before or after the table.

Researchers from the database community have been studying table search for many years [2, 4, 12, 32–34, 40]. Zhang and Balog [39] formally re-introduced the table retrieval task and built the 1st test collection based on Wikipedia tables. However, there is a lack of public test collections on arbitrary Web tables for appropriate evaluation. Sun et al. [28] released an open domain dataset, WebQueryTable, for table retrieval, which was collected from logs of a commercial search engine[1]. Shraga et al. [25] constructed GNQtables dataset from a QA dataset. Given a question-answer pair from Google Natural Questions dataset [17], they extracted the table from the Wikipedia page where the answer is from and treated it as the ground truth for table retrieval. However, this dataset is not publicly available and the details of dataset pre-processing are missing which makes it difficult to be reproduced.

The main objective of this work is to create a new Web table retrieval (WTR) collection which has the following improvements compared with previous collections:

(1) **Diversity.** The WikiTables collection [39] and GNQtables [25] only include tables from Wikipedia, while our collection covers broader topics (61,086 domain names). Our collection can include any user-generated content and less than 1% of the tables are from Wikipedia. For the same query "Fast cars", the relevant tables in the WikiTables collection list facts about different car models, while tables in WTR can contain subjective information such as the example in Figure 1[2].
(2) **Rich context.** As shown in Figure 1, each table in our new collection has four **context fields**: page title, text before the table, text after the table, and entities that are linked to the DBpedia [18]. We also keep other metadata about the source Web page. Details are in Section 3.1.
(3) **Labels on multi-fields.** We notice that the relevance of a table and its context fields could be different. For example, the entities in Figure 1 are different car models and therefore can be considered as relevant to the query "Fast cars" while the page title "News development" is irrelevant. This is the first collection that has separate relevance labels for different sections of a Web page containing a table.
(4) **Reproducibility.** We describe the details of dataset pre-processing and release the run files of baseline methods for easy comparison.

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 ACM 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.
SIGIR '21, July 11–15, 2021, Virtual Event, Canada
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3463260

[1] We noticed a statistical inconsistency between released dataset and original paper.
[2] Wikipedia has strict content policies (e.g., neutral point of view) and therefore is more limited to factual knowledge.

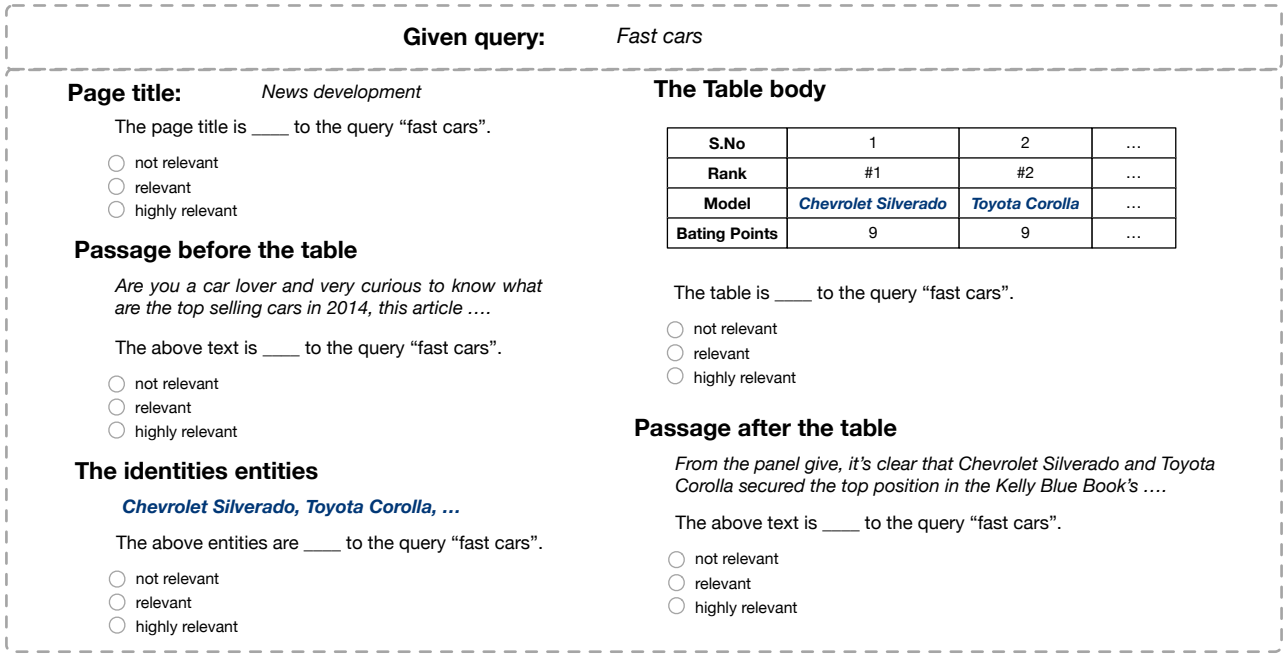

Figure 1: Illustration of the crowdsourcing interface design.

The WTR collection contains not only 6,949 annotated query-table pairs but also query-context pairs for each context field which are ignored in previous test collections. In addition, we provide details of how the corpus is pre-processed (Section 2) and rankings of both traditional and recently-proposed table search methods (Section 3 and 4), making a future comparison with other work easier. Our experimental results show that the performance of table retrieval models can be jeopardized if trained on datasets with annotation bias. With the WTR collection, we provide potential new research directions for researchers such as designing new models for table retrieval considering the relevance of context fields (Section 5).

2 CONSTRUCTING THE TEST COLLECTION

In this section, we describe the test collection, including the table corpus, queries and the process of collecting relevance assessments.

2.1 Query and Table Corpus

We build the test collection with the English subset of WDC Table Corpus 2015[3] which includes 50.8M relational HTML tables extracted from the July 2015 Common Crawl. All the tables are highly relational with an indexed core column including entities or a header row describing table attributes. Zhang et al. [41] further process the subset with novel entity discovery and link identified entities with their aliases to DBpedia. This results in 16.2M tables after retaining only those with at least one identified entity. Note that the WDC Table Corpus also includes Wikipedia pages (less than 1%) and can be considered as an extension of WikiTables collection [39]. However, this new collection contains tables from 61,086 different domain names and covers a broader range of topics.

The same set of queries with WikiTables collection [39] is used which includes 30 queries from Cafarella et al. [3] collected from Amazon's Mechanical Turk[4] platform and 30 queries from Venetis et al. [32] collected from query logs of searching for structured data.

2.2 Relevance Assessments

For our new collection, we first use a pooling method to fetch the candidate tables for all queries. Then for each query, we employ three independent annotators for relevance judgments.

2.2.1 Pooling. As a standard practice of IR test collection construction, we fetch the candidates by retrieving the top 20 tables using multiple unsupervised methods. Specifically, we use BM25 [23] to retrieve seven indexed fields: *Table*, *Caption*, *Page title*, *Header*, *TextBefore*, *TextAfter* and *Catchall*. The details of those fields are described in Section 3.1. The final assessment pool contains 6,949 query-table pairs.

2.2.2 Collecting Relevance Judgments. In the WikiTables collection, a table, along with other context fields such as page title and section title, is presented to an annotator, while only a single relevance label is assigned. However, a table does not always have the same relevance label as context fields. For example, the entities in Figure 1 representing different car models are relevant to the query "Fast cars" while the page title "News development" is irrelevant. Therefore, for each record (i.e., query-table pair) in our new collection, we ask the annotator to judge the relevance of each field concerning a query as shown in Figure 1. We use Amazon's Mechanical Turk to collect all the judgments based on a three-point scale:

[3] http://webdatacommons.org/webtables/2015/EnglishStatistics.html

[4] https://www.mturk.com/

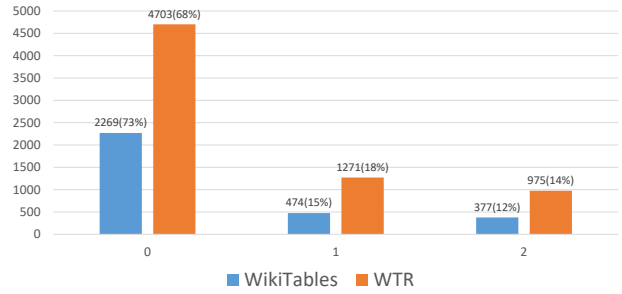

Figure 2: Comparison of relevance labels between WikiTables and WTR collection.

- **Irrelevant (0)**: this field is irrelevant to the query (i.e., based on the context you would not expect this to be shown as a result from a search engine).
- **Relevant (1)**: this field provides relevant information about the query (i.e., you would expect this Web page to be included in the search results from a search engine but not among the top results).
- **Highly relevant (2)**: this field provides ideal information about the query (i.e., you would expect this Web page ranked near the top of the search results).

To control the annotation quality, 135 query-table pairs for one query annotated by the authors are taken as testing questions (where at least two of the experts agreed on the relevance label). Each of the remaining query-table pairs is paired with a randomly sampled testing question, which means a single assignment consists of two query-table pairs. Based on the testing questions, we regularly examine the following metrics:

- **Assignment accuracy**: For an assignment, if 3 out of 5 relevance judgments corresponding to gold labels of a testing question are correct, then the assignment accuracy is 60%.
- **Worker accuracy**: If a worker submits the results of 5 assignments, there are $5 \times 5 = 25$ labels of 5 testing questions. If 20 out of the 25 labels are corrected, then the worker accuracy is 80%.
- **Approval rate**: If a worker submits 50 assignments and 30 assignments are approved, then the approval rate of the worker is 60%.

We first approve those assignments with at least 60% assignment accuracy or the assignments from workers whose worker accuracy is at least 80%. Then we only allow those workers whose approval rates are at least 50% to continue the remaining annotation process. We collected 3 judgments for each record and paid workers 5 cents per assignment. In Table 1, we show the Fleiss' Kappa inter-annotator agreement of different sections. From the table, we can see that the annotators disagree with the judgments on entities the most and make fair agreements on other sections. To determine the relevance label of each section to a query, we took the majority vote among three judgments. In case of a tie, we took the average of relevance scores as the final judgment. We compare the statistics of labels between WTR and WikiTables collection [39] in Figure 2. We can see that the proportion of 0,1,2 labels in the two datasets are similar but our collection is more than twice the size of the WikiTables collection.

2.2.3 Inter-field analysis. One of our main contributions is that we provide the relevance annotations of context fields of a table. The authors of WikiTables collection [39] only asked workers to assign a single label for a table, but the context information (e.g., page title) was also presented to workers, which may mislead the workers. For example, if a page title is relevant while the table is not, then a worker may still consider it as relevant. Besides, lack of context could lead to misunderstanding of tables. For example, the table in Figure 1 provides a ranked list of cars. Without surrounding passages saying the ranking criteria are subjective ratings from customers, an annotator may think the cars in the table are ranked by speed and annotates it as relevant to the query "fast cars".

To study the discrepancy of relevance judgments among different fields, we could make the assumption that the four relevance labels of context fields are also relevance judgments for the Web table. It resembles the situation in which we ask five annotators to judge every query-table pair and we want to know the agreement between any two raters (fields). The Cohen's Kappa statistics between any two raters (fields) are summarized in Table 2. As we can see from the table, different context fields have different levels of agreement with *Table*. Among all the context fields, *TextAfter* has the most agreement with *Table* while *Page Title* has the least agreement with *Table*. *Page Title* often includes little information and even if its content is relevant the worker can hardly recognize it. However, the content in *TextAfter* is more likely to be the paragraph that explains the details of the *Table* which deepens the reader's understanding. As shown in Figure 1, *Page Title* "News Developments" seems to be a general title of a news website and tells nothing related to the query or *Table* on the same page. But the content in *TextAfter* further illustrates the *Table*.

3 EXPERIMENTS

In this section, we first describe the details of data pre-processing and indexing steps so that the same corpus can be reproduced. Then we present the baseline methods of table retrieval.

3.1 Preprocessing and Indexing

Leveraging the entity linking results of Zhang et al. [41] which are formed as <table mention, entity entries> pairs[5], we construct the table corpus by extracting the original tables from the English subset of WDC Table Corpus 2015[6] and mapping mentions to KB entries. This corpus is comprised of 3M relational tables. In addition to the original fields (cf. Table 3), each table has an *Entities* field,

[5]https://zenodo.org/record/3627274#.YC91NehKh1Q
[6]http://data.dws.informatik.uni-mannheim.de/webtables/2015-07/englishCorpus/compressed/

Table 1: The Fleiss' Kappa inter-annotator agreement of different sections.

Field	Fleiss' Kappa
page title	0.28
text before the table	0.26
table	0.27
entities	0.20
text after the table	0.23

Table 2: The Cohen's kappa inter-annotator agreement between any two fields.

	TextBefore	Page title	Table	Entities	TextAfter
TextBefore	1	0.44	0.41	0.34	0.41
Page title	0.44	1	0.39	0.33	0.39
Table	0.41	0.39	1	0.4	0.43
Entities	0.34	0.33	0.4	1	0.37
TextAfter	0.41	0.39	0.43	0.37	1

Table 3: Table fields used when constructing the corpus.

Field	Definition
Page title	Title of the Web page where the table lies. It is determined from HTML tags
TextBefore	200 words before the table
Caption	Caption of the table
Table	The extracted table from a Web page
Header	Headers of the table if exist
Entities	Entities in the tables identified by the novel entity discovery process [41]
TextAfter	200 words after the table
Orientation	Orientation can be either horizontal or vertical. A horizontal table has attributes in columns while the attributes of a vertical table are represented in rows
URL	The original web address of the Web page
Key Column	For a horizontal table, the key column is the one contains the names of the entities
Catchall	The concatenation of page title, caption, table, text before and after the table

listing all the in-KB entities identified by [41]. We use Elasticseach[7] for indexing the tables, separated by the fields in Table 3. The scripts to download, preprocess and index the data are also available on our GitHub repository.[8]

3.2 Baseline Methods

A line of work has developed for the ad hoc table retrieval task. We consider the classic and some of the state-of-the-art methods as baseline approaches, which include both unsupervised and supervised methods.

- **Single-field document ranking:** Each table is represented as a single document [4]. In our corpus, the content in the *Catchall* field described in Table 3 is used as the table representation. Then classic IR methods like Language Models with Dirichlet smoothing are used for ranking.
- **Multi-field document ranking:** Each table is represented as a multi-field document [22]. We use the following five fields: *page title*,*TextBefore*,*Table*, *TextAfter* and *Header*. Unlike Pimplikar and Sarawagi [22] and Zhang and Balog [39], we do not consider *Caption*. Since we find that few tables have non-empty table captions and most tables with captions are from Wikipedia.
- **LTR: L**earning-**T**o-**R**ank [39] is a feature-engineering approach leveraging table structure and lexical features (Features 1-12 in Table 4). A random forest is used to fit the ranking features in a pointwise manner.

[7]https://www.elastic.co/downloads/past-releases/elasticsearch-5-3-0
[8]https://github.com/Zhiyu-Chen/Web-Table-Retrieval-Benchmark

Table 4: Features extracted from Web tables which are used in LTR and STR methods.

ID	Description	Dim.
1	Number of query terms	1
2	Sum of query IDF scores (from indexed fields except *Caption*)	6
3	The number of rows in the table	1
4	The number of columns in the table	1
5	The number of empty table cells	1
6	Ratio of table size to page size	1
7	Total query term frequency in the leftmost column cells	1
8	Total query term frequency in second-to-leftmost column cells	1
9	Total query term frequency in the table body	1
10	Ratio of the number of query tokens found in page title to total number of tokens	1
11	Ratio of the number of query tokens found in table title to total number of tokens	1
12	Language modeling score between query and multi-field document representation of the table	1
13	Four semantic similarities between the query and table represented by bag-of-word embeddings.	4
14	Four semantic similarities between the query and table represented by bag-of-entities.	4
15	Four semantic similarities between the query and table represented by bag-of-graph embeddings.	4

- **STR:** The **S**emantic-**T**able-**R**etrieval approach [39] extends LTR with semantic features such as bag-of-categories, bag-of-entities, word embeddings, and graph embeddings. These embeddings are fused in different strategies [38] to generate ranking features (Features 13-15 in Table 4). Like LTR, a random forest is used for pointwise regression.
- **BERT-ROW-MAX** Chen et al. [10] propose different content selectors to select the most salient pieces of a table as BERT input. BERT-ROW-MAX, which uses BERT as the backbone and max salience selector with row items, achieves state-of-the-art performance on previous table retrieval datasets.

Multimodal **T**able **R**etrieval (MTR) [25] is also a recently proposed approach which learns a joint representation of the query and the different table sections. However, we find their repository[9] is inaccessible which leads to reproducibility difficulties. We also find their reported results of baselines (LTR, STR,etc.) on WikiTables collection are from the original runs in [39]. It is unfair to directly compare the evaluation results since their model and baselines are not trained on the same splits used in 5-fold cross validation. Therefore, their conclusions are not convincing and we did not attempt to implement their method from scratch.

3.3 Implementation Details

For single-field and multi-field document ranking, we use the implementations from Nordlys [13] which has interfaces to Elasticsearch. Since the features used by LTR rely on the source of tables and some features extracted from Wikipedia tables are not available for Web tables (e.g., number of page views), we generate the features which

[9]https://github.com/haggair/gnqtables

Table 5: The evaluation results of compared table retrieval baseline methods.

Model	MAP	P@5	P@10	NDCG@5	NDCG@10
Single-field document ranking	0.5071	0.4380	0.3966	0.4124	0.4851
Multi-field document ranking	0.5104	0.4407	0.3943	0.4201	0.4916
LTR	0.5878	0.5300	0.4433	0.5313	0.5870
STR	0.6210	0.5493	0.4727	0.5585	0.6258
BERT-ROW-MAX(base)	**0.6346**	**0.5713**	**0.4800**	**0.5737**	**0.6327**

Table 6: Table retrieval evaluation results with different annotation "bias". We highlight the cases where the results are improved over the original method.

Model	MAP	P@5	P@10	NDCG@5	NDCG@10
STR	0.6210	0.5493	0.4727	0.5585	0.6258
STR (Max-Entities)	**0.6261 (+0.82%)**	**0.5573 (+1.46%)**	**0.4763 (+0.77%)**	**0.5634 (+0.88%)**	**0.6281 (+0.36%)**
STR (Min-Entities)	0.5893 (-5.11%)	0.5253 (-4.37%)	0.4533 (-4.10%)	0.5251 (-5.99%)	0.5872 (-6.18%)
STR (Max-PageTitle)	0.6137 (-1.18%)	0.5487 (-0.12%)	0.4677 (-1.06%)	0.5517 (-1.22%)	0.6154 (-1.67%)
STR (Min-PageTitle)	0.6060 (-2.42%)	0.5473 (-0.36%)	0.4613 (-2.40%)	0.5419 (-2.98%)	0.6041 (-3.47%)
STR (Max-TextBefore)	0.6127 (-1.33%)	0.5380 (-2.06%)	0.4703 (-0.50%)	0.5472 (-2.03%)	0.6171 (-1.39%)
STR (Min-TextBefore)	0.6021 (-3.04%)	0.5480 (-0.24%)	0.4547 (-3.81%)	0.5471 (-2.04%)	0.6003 (-4.08%)
STR (Max-TextAfter)	0.6203 (-0.12%)	**0.5533 (+0.73%)**	**0.4770 (+0.91%)**	0.5545 (-0.72%)	**0.6265 (+0.11%)**
STR (Min-TextAfter)	0.6007 (-3.26%)	0.5327 (-3.03%)	0.4577 (-3.17%)	0.5294 (-5.21%)	0.5970 (-4.61%)

apply to the new test collection. The original STR calculates four semantic representations for queries and tables. Since there are no Wikipedia categories for all Web tables, we use three semantic representations for queries/tables: bag-of-entities, word embeddings, and graph embeddings. For each type of semantic representation, we use early fusion, late-max, late-sum, and late-max, as described in Zhang and Balog [38] to obtain four semantic matching scores, which results in 12 semantic matching features in total for each query-table pair. We summarize the features in Table 4, where features 1-12 are used in LTR and features 1-15 are used in STR. The scikit-learn[10] implementation of random forest is used for both LTR and STR. We set the number of trees to 1000 and a maximum number of features in each tree to 3. For BERT-ROW-MAX, we use the pre-trained BERT-base-cased model[11] which consists of 12 layers of Transformer blocks. As in Chen et al. [10], we train the model with 5 epochs, and batch size is set to 16. The Adam optimizer [16] with a learning rate of 1e-5 is used to optimize BERT-ROW-MAX. A linear learning rate decay schedule with a warm-up ratio of 0.1 is also used. We train all the models using 5-fold cross-validation (w.r.t. NDCG@5). To facilitate fair comparison in future work, we prepared the five data splits used for cross-validation in our WTR repository[8], which is not provided by previous work [39].

4 RESULTS AND ANALYSIS

4.1 Overall Performance

In Table 5 we report on the performance of different table retrieval methods. The conclusion is consistent with previous work [10, 39] that BERT-ROW-MAX achieves the best overall evaluation metrics. STR outperforms LTR and other unsupervised methods significantly. However, different from the results on WikiTables collection, BERT-ROW-MAX does not outperform STR sufficiently. We speculate that the new corpus may include more noise which makes it more difficult for neural models to learn features. In contrast, LTR and STR are trained on curated features which are more robust to noise in raw text.

4.2 Utilizing Labels of Context Fields

Recall that we have explicitly collected the relevance judgments of different fields in Section 2.2, we observe that a context field does not always have the same relevance label as the table. This discrepancy could result in bias in the annotation process of WikiTables collection. For example, a strict annotator may think a table should be annotated as relevant when all the context fields are also relevant. While a lax annotator may consider a table as relevant when any of the context fields provides related information even if the table itself is not relevant. To investigate how the labels of context fields can help table retrieval, e.g., how the annotation bias could potentially affect the models, we employ two strategies to resemble a strict annotator and a lax annotator: given the original label l_t of a table and the label l_c of one context field, we re-annotate l_t as either $max(l_t, l_c)$ or $min(l_t, l_c)$. Then we can use the "new" labels to train the models but evaluate on original labels.

We take the STR approach as the example and in total form 8 variations for STR which are trained with "biased" labels by combining the min/max strategy with four context fields. We summarize the results of STR trained on new labels in Table 6. For example, STR (max-entity) means STR is trained with the new labels where the maximum relevance among a table and its *Entities* field is used as the training label for each query-table pair. Note that we only change the labels in the training set and keep the original testing

[10] https://scikit-learn.org/
[11] https://huggingface.co/transformers/pretrained_models.html

set since the task is still table retrieval rather than context field retrieval. We can observe that the performance of majority models decreases compared with the model trained on the original labels, which demonstrates that the biased annotation can harm the model training. Among those cases, STR (Min-Entities) has the most performance drop. Nevertheless, there are a few cases where the performance slightly increased. It is worth noting that STR (max-entity) performs better than the model trained on the original labels on all the evaluation metrics. We observe the same results on other supervised methods. Table retrieval and entity retrieval may be highly synergistic tasks. Taking good advantage of labels from *Entities* can help table retrieval while misusing them can lead to a large performance drop.

5 DISCUSSION

5.1 Availability

We publish the following resources in our WTR repository[8]:

- **Scripts**: for reproducibility, we release the scripts to download and preprocess the WDC Table corpus, and match the entity linking results in [41].
- **WTR table dump**: for ease of use, we release the pre-processed table corpus used for indexing and the pooling results used for relevance judgments.
- **Annotations**: we provide separate files containing the relevance judgments for query-table and query-context pairs. We also provide the five splits used in this paper for cross-validation.
- **Baselines**: we release the run files of baselines presented in this paper. For STR and LTR, we also release the reproduced features.

5.2 Future Directions

We propose the following research questions to indicate future research directions on utilizing this new test collection.

- **RQ1**: *Do current table retrieval methods have good transferability?*

The WTR collection is complementary to the previous WikiTables collection [39] and covers a broader range of topics. Wikipedia pages usually have a similar structure and the tables are well-formatted. However, the structure of different websites can be very different. For example, the page title of Wikipedia usually represents an entity or a very informative event. While for some websites, the page title can be very general and does not describe anything specific about the Web page (e.g., example in Figure 1). Therefore, it is possible that some features extracted from one domain do not generalize well to another domain. For example, when we reproduce the features of LTR and STR, we disregard some features proposed for Wikipedia specifically. It is an interesting question to study the transferability of different models and what features extracted from one dataset (training set) also generalize to another dataset (testing set).

- **RQ2**: *How can the relevance signals from context fields further improve the table retrieval task?*

In our experiments, we find that with a simple strategy to combine the relevance labels of the table and its *Entities* field, the model performance can increase (cf. Sect. 4.2). A natural follow-up question is: how can we design a model that automatically utilizes the labels of context fields to help table retrieval? This task is similar to multi-field document retrieval [24, 36] where the models take the advantage of document structure and handle multiple document fields. In the setting of Web table retrieval, multiple context fields can be created even when they are not tagged in the HTML source code. Zamani et al. [36] treat the body of a page as a single field, while for table search the body text can be further split into the passages before and after the table. Though previous works in multi-field document retrieval utilize all fields, the field-level relevance is not considered due to the lack of such annotations. Therefore, the WTR dataset can also be used as a collection for multi-field document retrieval.

- **RQ3**: *Could other tasks benefit table retrieval or vice versa?*

Though the objective of table retrieval is to return a list of relevant tables given a user query, a table alone may not satisfy a user's information need. For example, the table in Figure 1 provides a ranked list of cars but it does not tell how the scores are rated. Given the query "fast cars", a user may think the cars in the table are ranked by speed. However, the surrounding passages say the ranking criteria are subjective ratings from customers. Without any explanation about the table, a user could misunderstand the semantic meanings in the table. Therefore, context information is very important for a user to understand the table accurately. If we also consider passage retrieval as an auxiliary task for table retrieval, the search results presented to users can be more understandable.

In addition, Web table retrieval is also relevant to the task of entity retrieval [1, 5, 14]. Our test collection includes a large number of entities extracted from tables and linked to DBpedia. If a table is relevant, then it is very likely that the entities in the table are also relevant. Therefore, a model trained for table retrieval may also help entity retrieval. Our experimental results in Section 4 also indicate that table retrieval and entity retrieval are synergistic. The recent signs of progress in multi-task learning [20, 26] and pretraining techniques [11, 21] have shown that a model pre-trained or jointly trained on related tasks can help learn more generalized features and improve the model performance on target tasks. It will be interesting to study how we can utilize the datasets for entity retrieval [1, 14] to train models for table retrieval, or use the WTR collection to help train models for entity search.

6 CONCLUSION

In this work, we describe a new test collection WTR for the task of table retrieval. Compared with previous datasets, WTR covers a broader range of topics and includes tables from over 61,000 different domain names. We present a detailed walk-through of how the test collection was created. In addition to the WTR collection, we offer the runs of baseline methods and feature sets used in previous work. We also discussed a number of interesting opportunities for researchers to use WTR in their future work.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1816325. We thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] Krisztian Balog and Robert Neumayer. 2013. A test collection for entity search in DBpedia. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 737–740.

[2] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on Wikipedia. In *Proceedings of the ACM SIGKDD workshop on Interactive Data Exploration and Analytics*. 18–26.

[3] Michael J Cafarella, Alon Halevy, and Nodira Khoussainova. 2009. Data integration for the relational web. *Proceedings of the VLDB Endowment* 2, 1 (2009), 1090–1101.

[4] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: Exploring the power of tables on the Web. *Proceedings of the VLDB Endowment* 1, 1 (2008), 538–549.

[5] Jing Chen, Chenyan Xiong, and Jamie Callan. 2016. An empirical study of learning to rank for entity search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 737–740.

[6] Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020. HybridQA: A Dataset of Multi-Hop Question Answering over Tabular and Textual Data. *Findings of the Association for Computational Linguistics (EMNLP)* (2020), 1026–1036.

[7] Zhiyu Chen, H. Eavani, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with Pre-trained Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 183–190.

[8] Zhiyu Chen, Haiyan Jia, Jeff Heflin, and Brian D. Davison. 2018. Generating schema labels through dataset content analysis. In *Companion Proceedings of the The Web Conference*. 1515–1522.

[9] Zhiyu Chen, Haiyan Jia, Jeff Heflin, and Brian D Davison. 2020. Leveraging schema labels to enhance dataset search. In *European Conference on Information Retrieval*. Springer, 267–280.

[10] Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D Davison. 2020. Table search using a deep contextualized language model. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 589–598.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1. 4171–4186.

[12] Sainyam Galhotra and Udayan Khurana. 2020. Semantic Search over Structured Data. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 3381–3384.

[13] Faegheh Hasibi, Krisztian Balog, Darío Garigliotti, and Shuo Zhang. 2017. Nordlys: A toolkit for entity-oriented and semantic search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1289–1292.

[14] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. DBpedia-entity v2: a test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1265–1268.

[15] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. 2019. Sherlock: A deep learning approach to semantic data type detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1500–1508.

[16] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations*.

[17] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[18] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[19] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[20] Xiaodong Liu, Pengcheng He, W. Chen, and Jianfeng Gao. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 4487–4496.

[21] Matthew E. Peters, Mark Neumann, Robert L Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 43–54.

[22] Rakesh Pimplikar and Sunita Sarawagi. 2012. Answering Table Queries on the Web using Column Keywords. *Proceedings of the VLDB Endowment* 5 (2012), 908–919.

[23] Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

[24] Stephen Robertson, Hugo Zaragoza, and Michael Taylor. 2004. Simple BM25 extension to multiple weighted fields. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management*. 42–49.

[25] Roee Shraga, Haggai Roitman, Guy Feigenblat, and Mustafa Cannim. 2020. Web Table Retrieval using Multimodal Deep Learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1399–1408.

[26] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning?. In *International Conference on Machine Learning*. PMLR, 9120–9132.

[27] Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. 2016. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web*. 771–782.

[28] Yibo Sun, Zhao Yan, Duyu Tang, Nan Duan, and Bing Qin. 2019. Content-based table retrieval for web queries. *Neurocomputing* 349 (2019), 183–189.

[29] Mohamed Trabelsi, Zhiyu Chen, Brian D Davison, and Jeff Heflin. 2020. A hybrid deep model for learning to rank data tables. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 979–986.

[30] Mohamed Trabelsi, Zhiyu Chen, Brian D. Davison, and Jeff Heflin. 2020. Relational Graph Embeddings for Table Retrieval. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 3005–3014.

[31] Mohamed Trabelsi, Brian D Davison, and Jeff Heflin. 2019. Improved table retrieval using multiple context embeddings for attributes. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 1238–1244.

[32] Petros Venetis, Alon Halevy, Jayant Madhavan, Marius Paşca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering Semantics of Tables on the Web. *Proc. VLDB Endow.* 4, 9 (June 2011), 528–538. https://doi.org/10.14778/2002938.2002939

[33] Robert H. Warren and Frank Wm. Tompa. 2006. Multi-column substring matching for database schema translation. In *Proceedings of the 32nd International Conference on Very Large Data Bases*. VLDB Endowment, 331–342.

[34] Chuan Xiao, Wei Wang, Xuemin Lin, and Haichuan Shang. 2009. Top-k set similarity joins. In *IEEE 25th International Conference on Data Engineering*. IEEE, 916–927.

[35] Yang Yi, Zhiyu Chen, Jeff Heflin, and Brian D Davison. 2018. Recognizing quantity names for tabular data. In *Joint Proceedings of the First International Workshop on Professional Search (ProfS2018); the Second Workshop on Knowledge Graphs and Semantics for Text Retrieval, Analysis, and Understanding (KG4IR); and the International Workshop on Data Search (DATA:SEARCH'18) Co-located with ACM SIGIR, Ann Arbor, Michigan, USA, July 12, 2018 (CEUR Workshop Proceedings, Vol. 2127)*. CEUR-WS.org, 68–73.

[36] Hamed Zamani, Bhaskar Mitra, Xia Song, Nick Craswell, and Saurabh Tiwary. 2018. Neural ranking models with multiple document fields. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. 700–708.

[37] Li Zhang, Shuo Zhang, and Krisztian Balog. 2019. Table2vec: Neural word and entity embeddings for table population and retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1029–1032.

[38] Shuo Zhang and Krisztian Balog. 2017. Design Patterns for Fusion-Based Object Retrieval. In *Advances in Information Retrieval*, Joemon M Jose, Claudia Hauff, Ismail Sengor Altıngovde, Dawei Song, Dyaa Albakour, Stuart Watt, and John Tait (Eds.). 684–690.

[39] Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of The Web Conference*. 1553–1562.

[40] Shuo Zhang and Krisztian Balog. 2020. Web Table Extraction, Retrieval, and Augmentation: A Survey. *ACM Trans. Intell. Syst. Technol.* 11, 2, Article 13 (Jan. 2020), 35 pages.

[41] Shuo Zhang, Edgar Meij, Krisztian Balog, and Ridho Reinanda. 2020. Novel entity discovery from web tables. In *Proceedings of The Web Conference*. 1298–1308.