Abstract

We describe our software submission to the ECIR 2024 Workshop on Open Web Search [1]. We submit the query performance prediction toolkit qptk that comes with 12 performance predictors to substantially simplify the re-use of those predictors in ad-hoc retrieval experiments. Therefore, we have extended qptk so that it can run in TIREx and that it can process arbitrary workloads using ir_datasets and PyTerrier indices as inputs. We execute qptk on all 23 test collections in TIREx that have PyTerrier indices available and make their predictions and the Docker image publicly available. Thereby, subsequent retrieval experiments can easily re-use the predictions by just downloading a few kilobytes, instead of having to run the system on their machine (which is still possible, e.g., by executing the Docker image on new or modified inputs). Our analysis on the 23 test collections highlights predictor performance variability, emphasizing the importance of standardized baselines and evaluation methods. Openly sharing predictions aims to enhance research accessibility, promoting broader utilization of query performance predictors and inspiring the development of novel prediction and evaluation techniques.

1. Introduction

Query Performance Prediction (QPP) is defined as assessing the quality of the query effectiveness in the absence of relevance judgements [2, 3]. This task is particularly important under three major aspects. First, it allows for some form of automatic evaluation of the Information Retrieval (IR) system, thus reducing the cost of collecting highly expensive manual relevance annotations. Secondly, it can be used as a feature for IR tasks such as reranking, model selection [3, 4], and rank fusion [5]. From the query perspective, it can be used to carry out query suggestion [3, 4] or to identify particularly challenging queries, so that the system administrators can operate failure analysis and enhance the model on those specific queries [3].

While QPP holds significant potential for enhancing IR, its progress faces a major challenge linked to the evaluation and comparison methodologies employed. Building on discussions from the recent QPP++ workshop [6], we assert that a fundamental issue in QPP research lies in the lack of result reproducibility and stability. First of all, most of the traditional QPP approaches have been designed and tested considering the Query Language Model [7] as the underlying retrieval engine. Therefore, this might induce a general instability of the models when considering different retrieval models, especially if they are based on a widely different rationale, such as neural IR models [8, 9, 10]. This also reflects on the possible instability.
due to the usage of different hyper-parameters for the IR model, different implementations of it, or different query normalization strategies, such as the stemmer used or stopping list considered [4, 11, 12]. Secondly, QPP models are typically characterized by one or more parameters, such as the length of the retrieved list when considering post-retrieval models. If not properly tuned, such parameters might result in particularly ineffective predictions. Finally, QPP models are known to operate differently on different collections [2]: using the wrong QPP for a given collection might lead to low prediction performance and thus to a generally weak baseline. To summarize, when experimenting with QPP, three major aspects might substantially impact the performance of (baseline) predictors: the underlying retrieval model, the hyperparameters of the predictor, and the collection considered. Furthermore, the same aspects hinder the reproducibility of the QPP: it is not uncommon in the QPP scenario to observe the same model performing differently from paper to paper. This is often due to naturally occurring differences in the implementation, but also due to different experimental settings (i.e., the retrieval model, and/or hyperparameters).

To alleviate these limitations and foster simplified, reproducible and stable experimentation in QPP, we describe our approach to embed the Query Performance Prediction ToolKit (qpptk) [11, 12, 13] into TIRA [14] / TIREx [15]. This setup allows us to provide a solid and stable set of baselines for future experiments: when carrying out new experiments, we can ensure that the implementation and hyper-parameters of the IR models underneath remain the same as they are archived within TIREx. Secondly, it allows us to test the same QPP over the abundance of collections that are already available in TIREx. This, in turn, allows us to provide the research community with a large amount of – already computed – shared baselines for future QPP experiments. Finally, re-using cached outputs can help fast prototyping: when developing a new QPP, the practitioner can delegate secondary aspects, such as setting up the IR system or processing the corpus, to cached TIREx outputs, focusing exclusively on the development of the QPP. Our implementation is publicly available.¹

The remainder of the paper is organized as follows: Section 2 provides the background on the QPP methods, and describes the main QPP models that have been implemented in qpptk. Section 3 details how our qpptk component submitted to TIREx can be used to obtain reproducible QPP results. Section 4 reports our experimental analysis. Finally, section 5 draws the conclusions and outlines our future work.

2. Background

We describe here the predictors implemented within qpptk. According to the classical separation, predictors are divided into pre- and post-retrieval.

Before introducing the predictors, we provide here the notation adopted in the remainder of this work. Let $q$ be a query and $d$ a document belonging to a corpus of documents $C$ with $|C| = N$. Without loss of generality, we call $s(q,d)$ the score assigned by an arbitrary ranking model to the document $d$ in response to the query $q$. We call $D@k$ the list of the top-$k$ documents retrieved in response to the query. Additionally, given a term $t$, we define $f_t = |\{d \in C : t \in d\}|$ the document frequency (i.e., the number of documents the term appears

¹Code: https://github.com/Zendelo/QPP-EnhancedEval/tree/qpptk-dev
in), while $f_{d,t} = |\{w \in d : w = t\}|$ is the term frequency (i.e., the number of times the term appear in document $d$).

### 2.1. Pre-Retrieval Predictors

Pre-retrieval predictors are those predictors that base their prediction based only on the query tokens and the (indexed) corpus. We implemented three major categories of pre-retrieval predictors: those based on Inverse Document Frequency (IDF), those relying on Similarity between the query and the collection (SCQ), and those based on score variability (VAR).

**Inverse Document Frequency (IDF)-based Predictors** [16, 17] The IDF predictors rely on computing the IDF for each query term:

$$IDF(t) = \ln \left( 1 + \frac{N}{f_t} \right)$$

then, this predictor can be instantiated in two ways, either by computing the average IDF over all query terms (avgIDF) or by computing the maximum IDF (maxIDF) over the query terms. The rationale is that if the terms have high inverse document frequency, they are highly characterizing (i.e., they are contained in a few documents and are very specific). Therefore, it is more likely that documents containing such terms will be relevant, indicating how the retrieval will perform.

**Similarity between a Query and a Collection (SCQ)-based Predictors** [16] The term-wise SCQ score is defined as follows:

$$SCQ(t) = (1 + \ln(f_{c,t})) \cdot \left( 1 + \frac{N}{f_t} \right)$$

where $f_{c,t}$ is the number of times the term appears in the corpus. Once the SCQ has been computed for each term, it is possible to aggregate it either by summing the SCQ for all the query terms (SCQ), averaging (avgSCQ), or by computing the maximum (maxSCQ).

**Variability (VAR)-Based Predictors** [16] To compute the VAR predictors, it is first necessary to compute the weight $w_{d,t}$ of each query term $t$ with respect to each document $d$ as the TFIDF score $w_{d,t} = 1 + \ln(f_{d,t}) \cdot IDF(t)$. Called $D_t$ the set of documents containing $t$, the prediction weight of each term is defined as follows:

$$VAR(t) = \sum_{t \in q} \sqrt{\frac{1}{f_t} \sum_{d \in D_t} (w_{d,t} - \overline{w_t})^2}$$

where $\overline{w_t}$ is the average weight $w_{d,t}$ over the documents in $D_t$. As for the previous cases, the VAR score is aggregated over the query terms using either the sum (sumVAR), maximum (maxVAR), or the average (avgVAR).
Post-Retrieval Predictors

Post-retrieval predictors utilize both the query and the top-k retrieved documents to formulate their prediction. Among post-retrieval predictors, we recognize three major classes: those based on the coherence of the retrieved list (e.g., clarity), those based on the distribution of the scores (e.g., NQC, WIG, SMV), and those relying on the robustness of the results – i.e., how much the introduction of noise in the query or the collection changes the retrieved ranked list – such as the UEF framework.

Clarity [17] This represents one of the seminal efforts in the QPP domain. The approach consists of computing the language model for the first top-k documents, which we refer to as $\theta_{D@k}$. Then, this language model is compared with the language model of the entire corpus $\theta_C$. The rationale is that the language model for the first top-k is highly divergent from the language model of the entire collection, then documents are highly coherent internally and this hints at an effective retrieval. More in detail:

$$\text{Clarity}(q) = \sum_{w \in V} p(w | \theta_{D@k}) \frac{p(w | \theta_{D@k})}{p(w | \theta_C)},$$

where $V$ is the vocabulary and $p(w | \theta)$ is the probability of observing the token $w$ according to $\theta$ the language model.

Weighted Information Gain (WIG) [18] This predictor represents one of the first efforts in utilizing the distribution of the scores for the top-k retrieved documents to determine the retrieval performance. More in detail, the prediction is given as the average difference between the score of the top-k documents retrieved and the score that the entire corpus would obtain in response to the query, which acts as a regularization component.

$$WIG(q) = \frac{1}{k \sqrt{|q|}} \sum_{d \in D@k} (s(q, d) - s(q, C)).$$

Normalized Query Commitment (NQC) [19] This predictor is in line with WIG, with the main difference that, in this case, the statistic of interest is the variance of the scores of the first top-k documents, normalized by the score that the entire corpus would obtain in response to the query:

$$NQC(q) = \sqrt{\frac{1}{k} \sum_{d \in D@k} (s(q, d) - \bar{s}_{D@k})^2 \overline{s(q, C)}},$$

where $\bar{s}_{D@k} = \frac{1}{k} \sum_{d \in D@k} s(q, d)$.

Score Magnitude and Variance (SMV) [20] This QPP combines NQC and WIG, by taking into account both the magnitude of the scores, as well as their variance and it is defined as follows:

$$SMV(q) = \frac{1}{k} \sum_{d \in D@k} \left( s(q, d) \cdot \frac{\ln s(q, d)}{\bar{s}_{D@k}} \right).$$
3. Porting qpp\textsubscript{TK} to TIRE\textsubscript{x} for Simplified QPP Experiments

The qpp\textsubscript{TK} toolkit currently implements eight pre-retrieval and four post-retrieval QPP methods. Leveraging a PyTerrier index, the toolkit is designed with extensibility in mind, allowing for easy integration of additional methods. You can access qpp\textsubscript{TK} on GitHub.\footnote{https://github.com/Zendelo/QPP-EnhancedEval.} In this work, we integrate qpp\textsubscript{TK} into the TIRE\textsubscript{x} framework \cite{1}.\footnote{See https://www.tira.io/tirex. Accessed on 28-03-2024.} We dockerize qpp\textsubscript{TK} so that it can run in the TIRA sandbox in TIRE\textsubscript{x} and adopt it so that it can use arbitrary inputs from ir\textsubscript{Datasets} \cite{21}. To reduce the effort of running performance predictions, we configure qpp\textsubscript{TK} in TIRE\textsubscript{x} so that it uses the the PyTerrier \cite{22} Indexer that is dockerized in TIRA as previous stage so that it runs against prebuilt PyTerrier indices which makes its execution faster.

The incorporation of qpp\textsubscript{TK} into TIRE\textsubscript{x} serves multiple purposes. Firstly, it establishes a stable and reproducible baseline for future QPP experiments. Secondly, it allows the testing of identical QPP methods across all collections available in TIRE\textsubscript{x}, contributing multiple shared baselines to the research community. Lastly, it facilitates swift and seamless prototyping. Developers can focus exclusively on QPP method development and evaluation, delegating secondary tasks like setting up the IR system or processing the corpus to TIRE\textsubscript{x}.

To use qpp\textsubscript{TK} in TIRE\textsubscript{x}, users must install the TIRA package and execute the following code:\footnote{Assuming that PyTerrier and ir\textsubscript{Datasets} are already installed.}

```python
from tira.rest_api_client import Client
from tira.third_party_integrations import ensure_pyterrier_is_loaded

global_path = ensure_pyterrier_is_loaded()

import pyterrier as pt

tira = Client()

dataset = 'antique-test-20230107-training'
# load the dataset
pt_dataset = pt.get_dataset(f"irds:ir-benchmarks/{dataset}\")

# initialize QPP transformer
qpp_predictions = tira.pt.transform_queries(
    'ir-benchmarks/qpp\textsubscript{TK}/all-predictors',
    dataset)
# returns a DataFrame with all the predictions
qpp_predictions(pt_dataset.get_topics('query'))
```

Listing 1: Example of using qpp\textsubscript{TK} within TIRE\textsubscript{x}.

The code in Listing 1 demonstrates how using only a few lines of code, users can access the TIRA API, load a dataset, and initialize a QPP transformer. The transformer then returns a DataFrame with all the predictions. This example illustrates the ease of use and the potential of the TIRE\textsubscript{x} framework in facilitating QPP experiments.
4. Experimental Analysis

In the experimental phase, we perform retrievals using BM25, followed by the application of qpptrk to the 23 test collections within TIREx with PyTerrier indices, covering diverse retrieval scenarios, such as argumentation [23, 24, 23, 24, 25], web search [26, 27, 28, 29, 30, 31], news search [32, 33, 34, 35], question answering [36, 37, 38], and medical domains [39, 40, 41, 42, 43, 44, 45], and the historical Cranfield collection [46, 47]. The retrieval evaluation is conducted using the Average Precision (AP) metric, in line with prior work on QPP. The initial analysis involves comparing the outputs of QPP methods through the computation of the Kendall’s $\tau$ correlation coefficient with the AP values per-query. Kendall’s $\tau$ serves as a common metric in scrutinizing and comparing QPP method performances. The results are presented in Figure 1. Each cell encapsulates the $\tau$ coefficient’s dispersion for the respective QPP method for the corresponding dataset. For post-retrieval methods, these boxplots additionally incorporate the distinct parameters utilized for each method.

The observed boxplots highlight a substantial variation in prediction quality across datasets, covering a correlation spectrum from -0.25 to 0.5. This outcome underscores the dataset-centric nature of QPP method performance, aligning with established observations in prior research.

Given that the correlation coefficients are strongly dependent on the sample, we also make all the results available on TIREx, enabling anyone to reproduce and extend our analysis with minimal effort, thus fostering the reproducibility of QPP research.

To investigate the performance of QPP methods in more detail, we present a heatmap in Figure 2. This heatmap offers a detailed perspective on the Kendall’s $\tau$ coefficients for the pre-retrieval QPP methods, with each cell denoting the $\tau$ coefficient for a particular QPP method and dataset.

The heatmap visually illustrates the performance variations of QPP methods across the diverse datasets. Our observation from the heatmap highlights the dataset-centric nature of QPP method performance – no single method consistently outperforms others across all datasets. However, certain methods demonstrate superior performance on average, such as MaxVAR, AvgVAR and MaxSCQ. Notably, the pre-retrieval QPP methods operate independently of the retrieval model, making the prediction values agnostic to the retrieval model used. Nevertheless, it is essential to recognize that the correlation is measured with the retrieval model, introducing a dependency on the retrieval model employed. Consequently, direct comparisons across different publications are challenging unless the same index, retrieval model, and associated components

---

are employed consistently.

To further demonstrate the potential of QPP@TIREx for the community, we conduct a comprehensive ANalysis Of the VAriance (ANOVA) on the pre-retrieval QPP methods, employing 10 different retrieval models (rankers) and 23 diverse datasets. The rankers include BM25, LGD, PL2, TF-IDF, ANCE-Cosine, ColBERT, SBERT, MonoT5-3b, MonoT5-Base and DirichletLM. We randomly sample 30 queries from each dataset, reflecting the maximum number available across all datasets. We then compute sARE error values for each query across all datasets and rankers, utilizing them for the ANOVA analysis. The results, presented in Table 1, reveal that the ranker, dataset, and QPP factors are all statistically significant, indicating that at least one of the levels of each factor has a statistically significant effect on the prediction quality. The effect sizes are small, with the dataset factor having the largest effect size on prediction quality. The interaction effects Ranker:Dataset and Dataset:QPP are also significant, with Dataset:QPP having the most substantial effect size. This underscores the dependency of pre-retrieval QPP method performance on the dataset, with the influence of the retrieval model being dataset-specific.

Similarly to correlation, the values are computed per each combination of dataset, ranker, and QPP method.

### Table 1: ANOVA Results for Pre-retrieval QPP Methods

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SCQ</th>
<th>AvgSCQ</th>
<th>AvgIDF</th>
<th>MaxIDF</th>
<th>SumVAR</th>
<th>MaxSUMVAR</th>
<th>AvgVAR</th>
<th>MaxVAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>antique-test</td>
<td>0.03</td>
<td>0.08</td>
<td>0.08</td>
<td>0.06</td>
<td>0.06</td>
<td>0.19</td>
<td>0.13</td>
<td>0.11</td>
</tr>
<tr>
<td>argme-touche-2020-task-1</td>
<td>0.03</td>
<td>0.15</td>
<td>0.21</td>
<td>0.06</td>
<td>-0.02</td>
<td>0.11</td>
<td>-0.01</td>
<td>-0.01</td>
</tr>
<tr>
<td>argme-touche-2021-task-1</td>
<td>0.01</td>
<td>0.14</td>
<td>-0.08</td>
<td>-0.15</td>
<td>0.08</td>
<td>0.08</td>
<td>0.09</td>
<td>0.01</td>
</tr>
<tr>
<td>cord19-fulltext-trec-covid</td>
<td>0.06</td>
<td>0.10</td>
<td>-0.04</td>
<td>-0.02</td>
<td>0.09</td>
<td>0.11</td>
<td>0.17</td>
<td>0.10</td>
</tr>
<tr>
<td>cranfield</td>
<td>0.09</td>
<td>0.03</td>
<td>0.07</td>
<td>0.08</td>
<td>0.08</td>
<td>0.06</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>disks45-nocr-trec-robust-2004</td>
<td>0.06</td>
<td>0.15</td>
<td>0.20</td>
<td>0.20</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.26</td>
</tr>
<tr>
<td>disks45-nocr-trec7</td>
<td>0.13</td>
<td>0.28</td>
<td>0.24</td>
<td>0.18</td>
<td>0.26</td>
<td>0.29</td>
<td>0.30</td>
<td>0.33</td>
</tr>
<tr>
<td>disks45-nocr-trec8</td>
<td>0.03</td>
<td>0.21</td>
<td>0.24</td>
<td>0.26</td>
<td>0.23</td>
<td>0.34</td>
<td>0.34</td>
<td>0.37</td>
</tr>
<tr>
<td>gov-trec-web-2002</td>
<td>0.08</td>
<td>0.16</td>
<td>0.14</td>
<td>0.15</td>
<td>0.10</td>
<td>0.17</td>
<td>0.19</td>
<td>0.21</td>
</tr>
<tr>
<td>gov-trec-web-2003</td>
<td>-0.19</td>
<td>0.18</td>
<td>0.20</td>
<td>0.16</td>
<td>-0.12</td>
<td>0.18</td>
<td>0.24</td>
<td>0.22</td>
</tr>
<tr>
<td>gov-trec-web-2004</td>
<td>0.21</td>
<td>0.17</td>
<td>0.19</td>
<td>0.18</td>
<td>0.22</td>
<td>0.17</td>
<td>0.14</td>
<td>0.16</td>
</tr>
<tr>
<td>gov2-trec-tb-2004</td>
<td>0.10</td>
<td>0.14</td>
<td>0.09</td>
<td>0.10</td>
<td>0.21</td>
<td>0.22</td>
<td>0.22</td>
<td>0.23</td>
</tr>
<tr>
<td>gov2-trec-tb-2005</td>
<td>0.13</td>
<td>0.11</td>
<td>0.03</td>
<td>0.01</td>
<td>0.18</td>
<td>0.09</td>
<td>0.15</td>
<td>0.12</td>
</tr>
<tr>
<td>medline-2004-trec-genomics-2004</td>
<td>-0.05</td>
<td>0.12</td>
<td>0.17</td>
<td>0.03</td>
<td>-0.02</td>
<td>0.18</td>
<td>0.13</td>
<td>0.18</td>
</tr>
<tr>
<td>medline-2004-trec-genomics-2005</td>
<td>-0.21</td>
<td>0.02</td>
<td>-0.03</td>
<td>-0.07</td>
<td>-0.16</td>
<td>0.01</td>
<td>0.06</td>
<td>0.01</td>
</tr>
<tr>
<td>medline-2017-trec-pm-2017</td>
<td>0.33</td>
<td>-0.01</td>
<td>0.15</td>
<td>0.11</td>
<td>0.26</td>
<td>-0.03</td>
<td>0.13</td>
<td>-0.22</td>
</tr>
<tr>
<td>medline-2017-trec-pm-2018</td>
<td>0.10</td>
<td>0.02</td>
<td>0.02</td>
<td>0.03</td>
<td>0.15</td>
<td>-0.00</td>
<td>0.10</td>
<td>-0.01</td>
</tr>
<tr>
<td>msMarco-passage-trec-dl-2019-judged</td>
<td>0.02</td>
<td>-0.06</td>
<td>0.11</td>
<td>0.12</td>
<td>0.04</td>
<td>0.01</td>
<td>0.16</td>
<td>0.18</td>
</tr>
<tr>
<td>msMarco-passage-trec-dl-2020-judged</td>
<td>-0.07</td>
<td>-0.00</td>
<td>0.23</td>
<td>0.15</td>
<td>0.02</td>
<td>-0.15</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>ncorpus-test</td>
<td>0.17</td>
<td>0.25</td>
<td>0.10</td>
<td>0.10</td>
<td>0.23</td>
<td>0.25</td>
<td>0.22</td>
<td>0.22</td>
</tr>
<tr>
<td>wapo-v2-trec-core-2018</td>
<td>0.14</td>
<td>0.10</td>
<td>-0.08</td>
<td>0.03</td>
<td>0.17</td>
<td>0.09</td>
<td>0.01</td>
<td>0.16</td>
</tr>
</tbody>
</table>

**Figure 2:** Heatmap illustrating the Kendall’s τ coefficients for the pre-retrieval QPP methods. Each cell denotes the τ coefficient for a specific QPP method and dataset.
Table 1
ANOVA analysis of the pre-retrieval QPP methods. The table presents the degrees of freedom (DF), sum of squares (SS), mean square (MS), F-statistic (F), p-value (PR(>F)), and the effect size ($\omega^2$) for the ranker, dataset, and QPP factors. Note, that $\omega^2$ is ill-defined for non-significant effects.

<table>
<thead>
<tr>
<th></th>
<th>DF</th>
<th>SS</th>
<th>MS</th>
<th>F</th>
<th>PR(&gt;F)</th>
<th>$\omega^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ranker</td>
<td>9</td>
<td>6.290</td>
<td>0.699</td>
<td>14.025</td>
<td>&lt;0.001</td>
<td>0.002</td>
</tr>
<tr>
<td>Dataset</td>
<td>22</td>
<td>23.247</td>
<td>1.057</td>
<td>21.206</td>
<td>&lt;0.001</td>
<td>0.008</td>
</tr>
<tr>
<td>QPP</td>
<td>7</td>
<td>4.260</td>
<td>0.609</td>
<td>12.214</td>
<td>&lt;0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>Ranker:Dataset</td>
<td>198</td>
<td>26.832</td>
<td>0.136</td>
<td>2.720</td>
<td>&lt;0.001</td>
<td>0.006</td>
</tr>
<tr>
<td>Ranker:QPP</td>
<td>63</td>
<td>1.344</td>
<td>0.021</td>
<td>0.428</td>
<td>1.000</td>
<td>-0.001</td>
</tr>
<tr>
<td>Dataset:QPP</td>
<td>154</td>
<td>33.808</td>
<td>0.220</td>
<td>4.406</td>
<td>&lt;0.001</td>
<td>0.010</td>
</tr>
<tr>
<td>Residual</td>
<td>52117</td>
<td>2596.945</td>
<td>0.050</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Ranker:QPP interaction, however, is not found to be significant by itself.

5. Conclusion and Future Work

This paper introduces the qppTK toolkit, encompassing a diverse array of pre-retrieval and post-retrieval QPP methods. We present the integration of qppTK into TIREx, to simplify rapid and efficient experimentation with QPP methods.

The integration helps to establish stable and reproducible baselines for forthcoming QPP experiments, testing identical QPP methods across all available TIREx collections, and contributing multiple shared baselines to the research community. Currently, available QPP methods in qppTK include eight pre-retrieval methods and four post-retrieval methods, with the potential for further expansion. Moving forward, our future work involves expanding the qppTK toolkit to incorporate additional QPP methods and enhancing its integration into TIREx. All code and data used in this work are accessible on GitHub and Zenodo,\footnote{https://zenodo.org/records/10852738} encouraging the research community to utilize and build upon our contributions.

References


[33] E. M. Voorhees, D. Harman, Overview of the eight text retrieval conference (trec-8), in:
TREC, 1999.


[45] V. Boteva, D. G. Ghalandari, A. Sokolov, S. Riezler, A full-text learning to rank dataset for
