# A 'Pointwise-Query, Listwise-Document' based Query Performance Prediction Approach

Suchana Datta
University College Dublin, Ireland
suchana.datta@ucdconnect.ie

Sean MacAvaney
University of Glasgow, UK
sean.macavaney@glasgow.ac.uk

Debasis Ganguly
University of Glasgow, UK
debasis.ganguly@glasgow.ac.uk

Derek Greene
University College Dublin, Ireland
derek.greene@ucd.ie

## ABSTRACT

The task of Query Performance Prediction (QPP) in Information Retrieval (IR) involves predicting the relative effectiveness of a search system for a given input query. Supervised approaches for QPP, such as NeuralQPP [23] are often trained on pairs of queries to capture their relative retrieval performance. However, pointwise approaches, such as the recently proposed BERT-QPP [1], are generally preferable for efficiency reasons. In this paper, we propose a novel end-to-end neural cross-encoder-based approach that is trained *pointwise* on individual queries, but *listwise* over the top ranked documents (split into chunks). In contrast to prior work, the network is then trained to predict the number of relevant documents in each chunk for a given query. Our method is thus a split-n-merge technique that instead of predicting the likely number of relevant documents in the top-$k$ [1], rather predicts the number of relevant documents for each fixed chunk size $p (p < k)$ and then aggregates them for QPP on top-$k$. Experiments demonstrate that our method is significantly more effective than other supervised and unsupervised QPP approaches yielding improvements of up to 30% on the TREC-DL'20 dataset and by nearly 9% for the MS MARCO Dev set.

## CCS CONCEPTS

• **Information systems** → **Query intent**; **Information retrieval query processing**.

## KEYWORDS

Supervised Query Performance Prediction, Listwise BERT-based cross encoders, Pointwise QPP training

**ACM Reference Format:**
Suchana Datta, Sean MacAvaney, Debasis Ganguly, and Derek Greene. 2022. A 'Pointwise-Query, Listwise-Document' based Query Performance Prediction Approach. In *Proc. SIGIR 2022*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3477495.3531821


*SIGIR '22, July 11–15, 2022, Madrid, Spain*

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531821

## 1 INTRODUCTION

The objective of Query Performance Prediction (QPP) is to automatically estimate the retrieval quality of a search system for a query without the presence of relevance assessments [10]. Traditionally, QPP approaches have mainly consisted of unsupervised methods [12] which use the specificity of query terms (e.g. estimated with collection statistics), coupled with information from the top-retrieved set of documents on how topically distinct they are with respect to the rest of the collection [6, 19–22, 24]. Here a more distinct top-retrieved set is potentially indicative of more effective retrieval. Aspects of the top-retrieved documents that have been reported to be most useful for the QPP task have included the document scores themselves (also known as retrieval status values or RSVs) [24], together with their standard deviations [22].

Recently, supervised approaches have been shown to outperform their unsupervised counterparts [1, 3, 7, 23]. The increase in effectiveness, however, comes at the cost of necessitating the availability of a set of queries with ground-truth relevance assessments, which are used for supervised training. The first supervised approach proposed for QPP, the NeuralQPP method [23], used pairs of queries to learn a binary indicator denoting which one of the pair leads to a better (or worse) retrieval effectiveness. The study [23] proposed using a combination of RSVs, the term-document matrix of the top-$k$ retrieved set, and word embedding features as inputs to a neural network.

The main disadvantage of a pairwise strategy is that the number of pairs is quadratic with respect to the training set size, thus causing a significant increase in training time [1]. As a solution, the authors of [1] showed that a pointwise approach, which makes use of a cross-encoding based interaction of the BERT vectors of constituent query and document terms (an architecture that has been shown to be effective for passage and document relevance ranking [14, 17]), can perform well at QPP in practice. Instead of predicting a relative measure of query difficulty across a pair, the training objective in this case seeks to directly predict a retrieval effectiveness measure (e.g. MRR@10).

Motivated by the success of BERT-based QPP models [1], a groupwise query estimation framework is proposed [3] that utilizes both cross-query and cross-document information across groups to learn the query performance predictor. Similar to BERT-QPP, this is also a regression-based model that predicts individual score for each query-document pair. However, the end QPP score per query is obtained by aggregating predictions in each group.

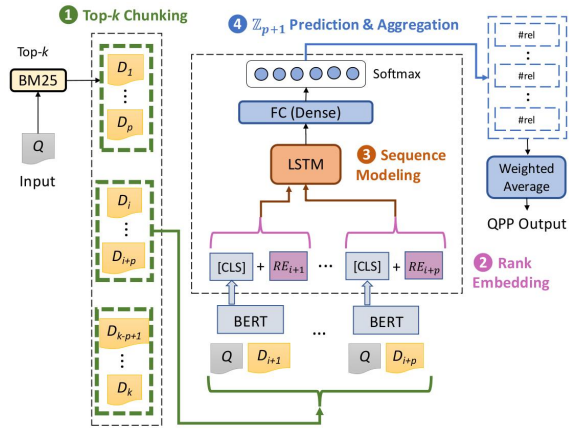



**Figure 1: Schematics of our proposed neural model 'qppBERT-PL' for a given query $Q$ and a list of top-ranked $k$ documents, $\{D_1, D_2, \ldots, D_k\}$ partitioned into $\lfloor k/p \rfloor$ chunks, each of size $p$. The query-document cross-encoded representations ($\Theta(Q, D_i)$) for each chunk along with the rank of a document in the form of BERT positional embeddings [9], are then encoded via an RNN (specifically, LSTM in our case). This is then passed through a fully connected layer (FC) terminating at a $p+1$ dimensional Softmax representing the probability of finding $r$ relevant documents within this $p$-sized chunk ($r \in \{0, 1, \ldots, p\}$). Through our experiments, we show that (1) top-$k$ chunking, (2) rank embedding, (3) sequence modeling, and (4) count prediction/aggregation are all important components of our approach.**

**Our Contributions**. The key contribution of this paper is a novel architecture and objective function for a pointwise neural query performance predictor. Instead of using a regression model to learn an aggregated retrieval effectiveness measure, we rather transform the pointwise QPP objective into a classification task, where we seek to explicitly predict the number of relevant documents likely to be observed in the set of top-retrieved documents. This choice is motivated by the first principle of QPP.

Moreover, in contrast to BERT-QPP [1], which models the top-retrieved documents as a *set*, our approach models this as a *sequence*, thus taking into account the relative positions (or ranks) of the top-$k$ documents. Hence, our proposed model can be categorized into a '**pointwise-query and listwise-document**' approach, which, to the best of our knowledge, has not been explored by the IR research community yet for the purpose of QPP. Our experiments demonstrate that the proposed approach achieves improvements of approximately 30% and 9% over the state-of-the-art QPP method BERT-QPP on TREC-DL'20 and MS MARCO Dev sets, respectively.

## 2 PROPOSED METHOD

In this section, we describe the details of our proposed neural architecture. The objective of the model is to predict the retrieval effectiveness or QPP score for a query $Q$, as a function of the query itself and the ordered set of top-retrieved documents $M_k = \{D_1, D_2, \ldots, D_k\}$:

$$\psi(Q, M_k) \mapsto \mathbb{R}. \tag{1}$$

**Network Architecture**. To model the input $M_k$ as an ordered set of documents, we make use of a recurrent neural network to encode the documents in sequence. Specifically, we use LSTM units [2, 13] for modeling the sequence (see Figure 1). An important decision choice with QPP estimators relates to the size of the top-retrieved document set. In the case of many popular QPP methods (e.g. NQC [22], WIG [24]), these have been reported to work well when using information from the top-100 documents. However, encoding such long sequences of 100 documents (which are themselves sequences of words) is likely to be noisy. Consequently, we segment the ordered set $M_k$ into equal sized partitions (chunks) of smaller ordered sets, namely

$$M_k = \bigcup_{i=1}^{\lfloor k/p \rfloor} \{M_k^{(i)}\} = \{D_1, \ldots, D_p\} \cup \ldots \{D_{k-p+1}, \ldots, D_k\}, \tag{2}$$

where $p (< k)$ is the size of each partition.

Each partitioned list of documents $M_k^{(i)} = \{D_i, D_{i+1} \ldots, D_{i+p}\}$, along with the query $Q$, constitutes an input instance. We employ a BERT-based cross-encoder architecture to model the interactions between the query and the document terms, followed by an LSTM-encoded representation of this interaction sequence (Figure 1). Formally,

$$\begin{aligned} \Theta_{Q,D_i} &= \text{BERT}([\text{CLS}]q_o, q_1, \ldots, q_{|Q|}[\text{SEP}]d_1, d_2, \ldots, d_{|D_i|}) \\ \Theta_{Q,M_k^{(i)}} &= \text{LSTM}(\theta_{Q,D_i}, \ldots, \theta_{Q,D_{i+p}}; \theta_{LSTM}) \\ \hat{y}(Q, M_k^{(i)}) &= \text{SOFTMAX}(\phi^T . \Theta_{Q,M_k^{(i)}}), \end{aligned} \tag{3}$$

where $\theta_{LSTM}$ and $\phi$ denote the parameters corresponding to the LSTM and a fully connected layer, respectively, $\Theta_{Q,D_i}$ denotes the BERT [9] encoding of the query-document pair $(Q, D_i)$, and $\hat{y}(Q, M_k^{(i)})$ denotes the predicted output through a SOFTMAX layer.

We name our proposed method as **qppBERT-PL**, the naming convention being explained below. Since our proposed model makes use of a sequence of chunked documents, it can be categorized as a listwise-document approach, which is why we include the suffix 'L' (denoting Listwise). On the other hand, since we incorporate the relative position (rank) information of the documents (so as to distinguish one input chunk from another), we include the suffix 'P' in the name of the model to denote the Position or absolute ranks of documents.

**Incorporating Rank Embeddings**. Since we provide the information from the top-$k$ retrieved list as separate chunks $M_k^{(i)}$, each of size $p$, we require a way to establish a link between these input chunks. A convenient way of doing this is by incorporating positional information into the embedded documents ($\Theta_{Q,D_i}$). We borrow an idea from the BERT model itself, which uses positional embeddings to indicate the relative positions of each token in the input text. Similarly, for a chunk $M_k^{(i)}$ comprised of documents $D_i, D_{i+1}, \ldots, D_{i+p}$, we add an embedding tied to $i$ (i.e., the document's rank) to the $\Theta_{Q,D_i}$ representation. It is important to note that our objective here is to model the sequence of *documents*, and not the sequence of the words themselves, as is the case in many NLP tasks.

**Training Objective**. The ground-truth values that the network seeks to predict correspond to the number of relevant documents in each partition $M_k^{(i)}$, which is an integer between 0 (none of the documents are relevant) and $p$ (all documents are relevant). To account for the likelihood of these $p$+1 possible integer (categorical) values, we model the output layer as a $p+1$ dimensional Softmax.

**Inducing the Query Ranks from the Estimator**. As a final step, we compute a weighted average from the outputs of the network, $\hat{y}(Q, M_k^{(i)})$, predicted for each $p$-sized partition of the top-$k$ documents to obtain an aggregated score, which we eventually use to sort (in descending order) the queries. More precisely, the rationale for using a weighted average is to favour the predicted relevance contributions from the chunks towards the top of the ranked list, in comparison to the ones that are at the bottom. Formally,

$$\psi(Q, M_k) = \sum_{i=1}^{\lfloor k/p \rfloor} \frac{\hat{y}(Q, M_k^{(i)})}{i}. \quad (4)$$

The $\psi(Q, M_k)$ scores of Equation 4 are subsequently used to yield an ordering of the set of input queries.

## 3 EXPERIMENTAL SETUP

We now describe a set of experiments designed to answer the following research questions:

- RQ1: Does our proposed qppBERT-PL model improve query performance estimates?
- RQ2: Does listwise-by-document modeling improve the effectiveness of baselines that estimate the performance directly?
- RQ3: Are chunk-level predictions important for the effectiveness of the qppBERT-PL?
- RQ4: What is the effect of the Rank Embedding (RE)?

### 3.1 Dataset

We conduct experiments on the MS MARCO passage dataset [16], which comprises of over 8.8M passages, along with a set of over 500K topics and relevant document pairs. To evaluate the results of our QPP experiments, similar to those reported in [1], we use the validation set of relevance-assessed queries, commonly known as "Dev". As in [1], we report our experiments on the queries used in the TREC-DL tasks from 2019 and 2020 [4, 5]. Table 1 provides an overview of the three datasets used in this paper.

In contrast to the MS MARCO queries, the ones in the TREC-DL tasks use depth pooling for relevance assessments, and hence are associated with a higher number of relevant documents, on an average, per query. TREC-DL uses a graded relevance. For our experiments, as per the official metric used in the track, we treat only the relevance level of 2 as relevant for computing the AP values. In line with prior work, we estimate the performance of BM25 results and we perform indexing and BM25 retrieval using PISA [15]. For QPP evaluation, we employ the commonly used correlation measures - Pearson's-$r$ and Kendall's-$\tau$ (denoted, respectively, as P-$r$ and K-$\tau$). While the former is a standard statistical correlation measure between two sets of values, the latter is a measure of the relative number of agreements in ranking order between two ordered sets of values.

**Table 1: Average number of relevant documents for each set of queries used for the evaluation of QPP in our experiments.**

| | MS MARCO Dev | TREC-DL'19 | TREC-DL'20 |
|---|---|---|---|
| #Queries | 6980 | 43 | 54 |
| Avg#rel | 1.1 | 58.2 | 38.7 |

**Table 2: A summary of extensions of the originally proposed BERT-QPP method [1], which act as ablations in our study. The original BERT-QPP method acts as one of our baselines. Prediction type $[0, 1]$ indicates a regression model, whereas $\mathbb{Z}_n$ denotes an $n$-class classification.**

| | Model | Type | Pred. | Seq. | Chunked | RE |
|---|---|---|---|---|---|---|
| $g$ | BERT-QPP [1] | Baseline | $[0, 1]$ | ✗ | ✗ | ✗ |
| $h$ | + Seq. | Ablation | $[0, 1]$ | ✓ | ✗ | ✗ |
| $i$ | + Seq. + RankEmb | Ablation | $[0, 1]$ | ✓ | ✗ | ✓ |
| $j$ | qppBERT-PL | Proposed | $\mathbb{Z}_{p+1}$ | ✓ | ✓ | ✓ |
| $k$ | − Seq. | Ablation | $\mathbb{Z}_{k+1}$ | ✗ | ✗ | ✓ |
| $l$ | − Chunked | Ablation | $\mathbb{Z}_{k+1}$ | ✓ | ✗ | ✓ |
| $m$ | − RankEmb | Ablation | $\mathbb{Z}_{p+1}$ | ✓ | ✓ | ✗ |
| $n$ | − Chunked − RankEmb | Ablation | $\mathbb{Z}_{k+1}$ | ✓ | ✗ | ✗ |

### 3.2 Methods Investigated

**Unsupervised baselines**. As in [1], we compare with a number of traditional unsupervised QPP approaches as baselines: Clarity [6], Weighted Information Gain (WIG) [24], and Normalized Query Commitment (NQC) [22]. A generalized model of NQC which claims that NQC computation can be derived as a scaled calibrated-mean estimator, namely SCNQC [19] is also considered as a baseline in this paper. As an additional baseline, we employ UEF [21], a method which applies a base QPP estimator to aggregate estimates from a number of subsets sampled from the top-retrieved set. The contribution from each subset depends on the relative stability in the rank order before and after relevance feedback with that set. As the base estimator for UEF, we used NQC, since it yields the most effective results compared to other post-retrieval estimators.

Similar to [1], we use a small subset, comprised of 100 queries, randomly sampled from the MS MARCO Dev topic set, to tune the hyper-parameters of the unsupervised baseline QPP approaches. The two main tuned hyper-parameters were the number of top-retrieved documents considered for the post-retrieval QPP methods (such as NQC, WIG and SCNQC), and the number of documents for relevance feedback in UEF. The baseline method SCNQC involves a number of hyper-parameters in terms of scaling and calibrating the NQC estimation, which we tune via a grid search as prescribed in [19].

**Supervised baselines**. As one of the supervised baselines, we consider **NeuralQPP** [23]. Unlike BERT-QPP and our proposed approach, this method is not an end-to-end supervised model as it requires inputs in the form of the outputs from several QPP estimators. It then employs weak supervision to learn an optimal combination of the estimators. As the next baseline method, we use the cross-encoder version of BERT-QPP [1] (as it outperforms the bi-encoder version). We refer to this baseline as **BERT-QPP**.

**Table 3: A comparison between the QPP effectiveness obtained by qppBERT-PL and the baselines.** The improvements of the best results obtained with qppBERT-PL (bold-faced) vs. the best performing baseline, BERT-QPP, are significant (t-test with 95% confidence).

| Type | | Models | MS MARCO Dev | | | | TREC-DL'19 | | | | TREC-DL'20 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | MRR@10 | | AP@100 | | MRR@10 | | AP@100 | | MRR@10 | | AP@100 | |
| | | | P-$r$ | K-$\tau$ | P-$r$ | K-$\tau$ | P-$r$ | K-$\tau$ | P-$r$ | K-$\tau$ | P-$r$ | K-$\tau$ | P-$r$ | K-$\tau$ |
| Baselines | $a$ | NQC [22] | 0.331 | 0.298 | 0.285 | 0.227 | 0.239 | 0.185 | 0.183 | 0.107 | 0.259 | 0.243 | 0.179 | 0.124 |
| | $b$ | Clarity [6] | 0.173 | 0.248 | 0.172 | 0.207 | 0.156 | 0.147 | 0.096 | 0.113 | 0.239 | 0.215 | 0.107 | 0.129 |
| | $c$ | WIG [24] | 0.193 | 0.215 | 0.215 | 0.203 | 0.192 | 0.133 | 0.133 | 0.089 | 0.260 | 0.241 | 0.143 | 0.096 |
| | $d$ | UEF(NQC) [21] | 0.347 | 0.313 | 0.294 | 0.227 | 0.254 | 0.235 | 0.189 | 0.112 | 0.275 | 0.291 | 0.200 | 0.126 |
| | $e$ | SCNQC [19] | 0.334 | 0.310 | 0.304 | 0.228 | 0.261 | 0.251 | 0.204 | 0.123 | 0.284 | 0.298 | 0.215 | 0.141 |
| | $f$ | NeuralQPP [8] | 0.215 | 0.197 | 0.173 | 0.193 | 0.156 | 0.126 | 0.129 | 0.133 | 0.271 | 0.253 | 0.133 | 0.112 |
| | $g$ | BERT-QPP [1] | 0.520 | 0.411 | 0.326 | 0.301 | 0.350 | 0.363 | 0.268 | 0.202 | 0.343 | 0.341 | 0.233 | 0.195 |
| | $h$ | + Seq. | 0.463 | 0.360 | 0.301 | 0.312 | 0.345 | 0.333 | 0.265 | 0.193 | 0.277 | 0.218 | 0.258 | 0.190 |
| | $i$ | + Seq. + RankEmb | 0.473 | 0.370 | 0.328 | 0.285 | 0.323 | 0.332 | 0.253 | 0.167 | 0.303 | 0.236 | 0.252 | 0.172 |
| Ours | $j$ | qppBERT-PL | **0.562** | **0.448** | **0.354** | **0.327** | **0.413** | **0.403** | **0.301** | **0.247** | **0.422** | **0.392** | **0.303** | **0.251** |
| | $k$ | – Seq. | 0.512 | 0.386 | 0.303 | 0.283 | 0.357 | 0.349 | 0.274 | 0.193 | 0.345 | 0.320 | 0.271 | 0.200 |
| | $l$ | – Chunked | 0.520 | 0.413 | 0.331 | 0.274 | 0.373 | 0.326 | 0.290 | 0.225 | 0.370 | 0.333 | 0.297 | 0.231 |
| | $m$ | – RankEmb | 0.519 | 0.392 | 0.320 | 0.267 | 0.361 | 0.328 | 0.285 | 0.232 | 0.352 | 0.331 | 0.293 | 0.215 |
| | $n$ | – Chunked – RankEmb | 0.405 | 0.329 | 0.293 | 0.285 | 0.309 | 0.299 | 0.260 | 0.159 | 0.217 | 0.198 | 0.199 | 0.184 |

**Ablations derived from the baseline BERT-QPP**. The BERT-QPP paper uses only a single document to train a regression model for predicting a target IR evaluation metric value. Since our model makes use of more than one document, we extend the original BERT-QPP by additionally encoding the information from top-$k$ retrieved as a sequence (similar to our proposed approach) so as to ensure a more fair comparison.

In one of these extended versions (see row $h$ of Table 2), we only include the information from the top-$k$ as a flat sequence (no chunking), whereas in the other version, we include the rank embedding information similar to our model (see row $i$ of Table 2). One of the main differences of our proposed model qppBERT-PL with the ones shown in the extensions to BERT-QPP (rows $h$ and $i$ of Table 2) is that the latter ones are regression models (see the 'Pred.' column of the table). These extensions to the BERT-QPP approach act as ablations to our complete model setup, and allow us to conduct more fair and comprehensive comparisons.

**Ablations of our proposed model**. In relation to our proposed model qppBERT-PL, we study several ablations by selectively removing one or more sources of information. First, instead of encoding the information from top-$k$ as a sequence, we simply use the top-retrieved documents, the only difference with BERT-QPP now being we include the rank embedding (see row $k$ of Table 2).

As our second ablation, instead of presenting a partitioned input of the top-$k$ documents to the qppBERT-PL, we learn to predict the number of relevant documents on the entire top-$k$ set by applying a $(k+1)$ dimensional Softmax (see row $l$ of Table 2).

Similarly, we derive our third ablation by removing the rank embedding information from qppBERT-PL (see row $m$). Finally, we remove both the chunk-based workflow and the rank embedding information to derive the ablation, shown in row $n$ of Table 2.

**Implementation-specific details**. All the supervised methods were trained on the MS MARCO training split of the data. The dimension of the hidden layer for the LSTM cells was set to 768 and that of the dense layer was set to 100, i.e., $\theta_{LSTM} \in \mathbb{R}^{768}$ and $\phi \in \mathbb{R}^{100}$ (see Equation 3). For the supervised models, we executed one epoch through the training set with a batch size of 16 as prescribed by the BERT-QPP paper [1]. For the classification methods, we used a cross-entropy loss. Parameter updates were performed using the Adam optimizer with a learning rate of 0.01[1].

For the regression-based methods (rows $g$ to $i$ of Table 2), we used the AP@100 values as the ground-truth for regression. In contrast, the ground-truth for our proposed model and its ablation variants (methods in the bottom group of Table 2) is the number of relevant documents of each chunk, or the total number of relevant documents in top-$k$, if chunking is not applied.

# 4 RESULTS

## 4.1 Main Observations

Table 3 presents a summary of the results of our experiments. We first observe that our proposed approach, qppBERT-PL (row $j$), outperforms all baseline approaches — including BERT-QPP ($g$), the prior state-of-the-art approach — for all three datasets and across all measures. The relative improvement ranges from 8.1% (MRR@10 P-$r$ on MS MARCO Dev) to 30.0% (AP@100 P-$r$ on TREC-DL'20), clearly showing a marked increase in the ability to predict query performance. The relative improvement on datasets with deeply-annotated labels (TREC-DL'19 and 20) were consistently higher than on the sparsely-annotated MS MARCO Dev set (+11.0–30.0% compared to +8.1–9.0%). All other baselines we explored (rows $a$–$f$) were considerably weaker still. These results clearly answer **RQ1:** We find that our proposed qppBERT-PL is more effective at predicting query performance than other known methods.

To test whether our proposed sequential modeling approaches can benefit the previous state-of-the-art model as well, we conduct ablations on BERT-QPP. Rows $h$ and $i$ show two versions of BERT-QPP that are generalised to closer match qppBERT-PL by, rather than consuming only the top retrieved item, taking the top-$k$ (and optionally including rank embeddings). We find that this approach

[1]Source code is available at https://github.com/suchanadatta/qppBERT-PL.git

**Figure 2:** Per-query comparisons of QPP effectiveness between qppBERT-PL and BERT-QPP in terms of scaled Absolute Rank Error (sARE) [11] computed with MRR (left) and AP (right). Comparisons are made on the TREC-DL dataset, comprising 97 queries. It can be seen that our method (qppBERT-PL) exhibits lower (hence more effective) sARE values on an average (bars with smaller heights).

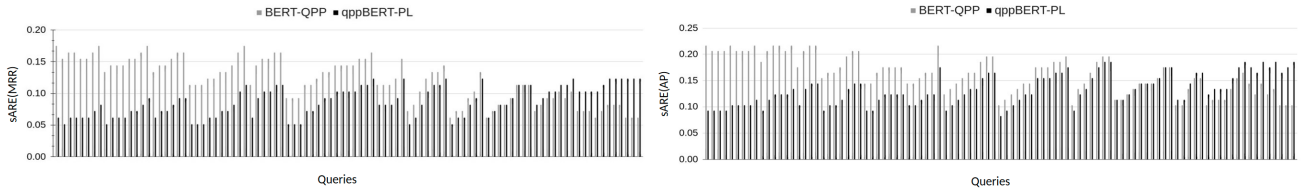
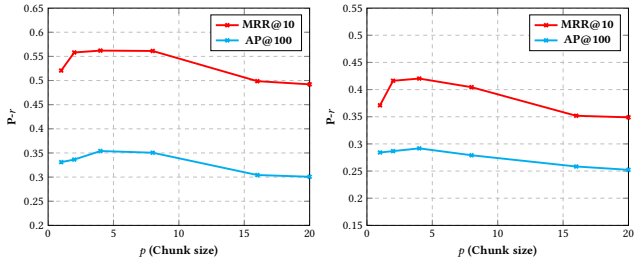



**Figure 3:** Sensitivity of qppBERT-PL on the MS MARCO Dev set (left) and the TREC-DL query set (right) with respect to the chunk size parameter ($p$).




tends to lead to a decrease in QPP performance. In two cases, the approaches can lead to a slight increase in performance (AP@100 K-$\tau$ on MS MARCO Dev and AP@100 P-$r$ on TREC-DL'19). Meanwhile, note that the sequence modeling appears to be critical for the success of qppBERT-PL; when sequence modeling is removed from the model (row $k$), QPP performance drops considerably. These results suggest that attempting to predict ranking effectiveness scores directly from sequences is challenging for models to learn using existing techniques, answering **RQ2**.

We now explore the effect of the proposed rank embedding and chunking components of qppBERT-PL. We observe that when we remove chunking (i.e., make a binary decision about each individual document, row $l$) or the rank embedding (i.e., do not provide the model with information about the absolute rank of the documents, row $m$), QPP performance drops to the level of around or below BERT-QPP. When both of these components are removed (row $n$), the performance drops even further. These results suggest that not only information about surrounding documents is necessary to estimate QPP well, but also the absolute rank of the documents within the ranked list.[2] The former observation is aligned with finding in neural ranking (via "duo" models [18]), but to the best of our knowledge, the latter has not been observed in other contexts. To answer **RQ3** and **RQ4**, we find that both chunking and Rank Embeddings are critical components of our proposed method.

## 4.2 Analysis

In this section, we report the per-query QPP effectiveness of TREC-DL topic set. For this, we employ the metric scaled Absolute Rank Error (sARE) proposed in [11]. More concretely speaking, the sARE metric computes the absolute difference between the position (rank) of a query when ordered by a ground-truth retrieval effectiveness metric (e.g. AP) and when ordered by the estimated QPP scores.

Figure 2 plots the sARE metric values for each query for our method and the best performing baseline, namely BERT-QPP (as observed from Table 3). By comparing the plots in Figure 2, both with MRR (sARE(MRR)) and AP (sARE(AP)), we observe that qppBERT-PL leads to lower rank errors than BERT-QPP on the TREC-DL dataset (on an average the dark-shaded bars are shorter than the lightly shaded ones).

As further analysis, we investigate how sensitive our proposed model, qppBERT-PL, is to the chunk size ($p$). We conduct a grid search for the optimal chunk size over the set $\{1, 2, 4, 8, 16, 20\}$. Figure 3 shows that the best results are obtained on both MS MARCO Dev and TREC-DL with a chunk size of 4. We observe that our method is somewhat insensitive to the chunk size parameter when $p$ is in the range between 2 and 8. Beyond this range, i.e., for too small or too large values of $p$, the effectiveness of qppBERT-PL decreases considerably.

The sensitivity plot of Figure 3 thus illustrates that prediction usually works well when the model is able to leverage information from a set of documents rather than a single one, as seen from the relatively low value of QPP effectiveness obtained with $p = 1$. However, using information from too many documents has a likely effect of confusing the model as can be seen from the decrease in QPP effectiveness for $p > 8$.

## 5 DISCUSSION AND CONCLUSIONS

In this paper we have proposed qppBERT-PL, a 'Pointwise-Query, Listwise-Document' approach for query performance prediction. We found that the model yields up to a 30% relative improvement in QPP. To the best of our knowledge, this is the first contribution in QPP that transforms the pointwise QPP objective into a listwise classification task.

Since being a BERT-based model qppBERT-PL is at present restricted by the maximum length of a BERT sequence (512 tokens), in future we plan to generalize the model to work with longer documents. This may be done by segmenting a long document into smaller passages and then aggregating the information from these passages in an effective manner. We are also interested to explore other neural architectures and training objectives to reduce the computation time of this listwise-document approach.

[2]It is important to remember, however, that the query performance itself is induced from individual chunk estimations in a final step, where rank information is provided.

***Acknowledgement***. The first and the fourth authors were partially supported by Science Foundation Ireland (SFI) grant number SFI/12/RC/2289_P2.

## REFERENCES


[1] Arabzadeh, N., Khodabakhsh, M., and Bagheri, E. *BERT-QPP: Contextualized Pre-Trained Transformers for Query Performance Prediction.* Association for Computing Machinery, New York, NY, USA, 2021, p. 2857–2861.

[2] Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., and Inkpen, D. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics ACL 2017, Volume 1: Long Papers* (2017), pp. 1657–1668.

[3] Chen, X., He, B., and Sun, L. Groupwise query performance prediction with bert. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II* (Berlin, Heidelberg, 2022), Springer-Verlag, p. 64–74.

[4] Craswell, N., Mitra, B., Yilmaz, E., and Campos, D. Overview of the TREC 2020 deep learning track. In *Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020* (2020), vol. 1266 of *NIST Special Publication*.

[5] Craswell, N., Mitra, B., Yilmaz, E., Campos, D., and Voorhees, E. M. Overview of the trec 2019 deep learning track, 2020.

[6] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (2002), SIGIR '02, p. 299–306.

[7] Datta, S., Ganguly, D., Greene, D., and Mitra, M. Deep-qpp: A pairwise interaction-based deep learning model for supervised query performance prediction. In *WSDM* (2022), ACM, pp. 201–209.

[8] Dehghani, M., Zamani, H., Severyn, A., Kamps, J., and Croft, W. B. Neural ranking models with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference* (2017), SIGIR '17, p. 65–74.

[9] Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. vol. abs/1810.04805.

[10] Diaz, F. Performance prediction using spatial autocorrelation. In *Proceedings of the 30th Annual International ACM SIGIR Conference* (2007), SIGIR '07, p. 583–590.

[11] Faggioli, G., Zendel, O., Culpepper, J. S., Ferro, N., and Scholer, F. An enhanced evaluation framework for query performance prediction. In *Advances in Information Retrieval* (Cham, 2021), Springer International Publishing, pp. 115–129.

[12] Hauff, C., Hiemstra, D., and de Jong, F. A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM CIKM* (2008), CIKM '08, p. 1419–1420.

[13] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural Computation 9*, 8 (1997), 1735–1780.

[14] MacAvaney, S., Yates, A., Cohan, A., and Goharian, N. Cedr: Contextualized embeddings for document ranking. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2019).

[15] Mallia, A., Siedlaczek, M., Mackenzie, J., and Suel, T. PISA: performant indexes and search for academia. In *Proceedings of the Open-Source IR Replicability Challenge co-located with 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, OSIRRC@SIGIR 2019* (2019), pp. 50–56.

[16] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@NIPS* (2016), vol. 1773 of *CEUR Workshop Proceedings*.

[17] Nogueira, R., and Cho, K. Passage re-ranking with bert. *ArXiv abs/1901.04085* (2019).

[18] Nogueira, R., Yang, W., Cho, K., and Lin, J. J. Multi-Stage Document Ranking with BERT. *ArXiv abs/1910.14424* (2019).

[19] Roitman, H. Normalized query commitment revisited. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2019), SIGIR'19, Association for Computing Machinery, p. 1085–1088.

[20] Roitman, H., Erera, S., and Weiner, B. Robust standard deviation estimation for query performance prediction. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval* (New York, NY, USA, 2017), ICTIR '17, Association for Computing Machinery, p. 245–248.

[21] Shtok, A., Kurland, O., and Carmel, D. Using statistical decision theory and relevance models for query-performance prediction. In *In Proc. of SIGIR'10* (2010), SIGIR '10, p. 259–266.

[22] Shtok, A., Kurland, O., Carmel, D., Raiber, F., and Markovits, G. Predicting query performance by query-drift estimation. *ACM Trans. Inf. Syst. 30*, 2 (2012).

[23] Zamani, H., Croft, W. B., and Culpepper, J. S. Neural query performance prediction using weak supervision from multiple signals. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2018), SIGIR '18, Association for Computing Machinery, p. 105–114.

[24] Zhou, Y., and Croft, W. B. Query performance prediction in web search environments. In *Proc. 30th International ACM SIGIR Conference* (New York, NY, USA, 2007), SIGIR '07, Association for Computing Machinery, p. 543–550.