Q2R: A Query-to-Resolution System for Natural-Language Queries

Shiau Hong Lim
IBM Research, Singapore
shonglim@sg.ibm.com

Laura Wynter
IBM Research, Singapore
lwynter@sg.ibm.com

Abstract

We present a system for document retrieval that combines direct classification with standard content-based retrieval approaches to significantly improve the relevance of the retrieved documents. Our system exploits the availability of an imperfect but sizable amount of labeled data from past queries. For domains such as technical support, the proposed approach enhances the system's ability to retrieve documents that are otherwise ranked very low based on content alone. The system is easy to implement and can make use of existing text ranking methods, augmenting them through the novel Q2R orchestration framework. Q2R has been extensively tested and is in use at IBM.

1 Introduction

A document retrieval system typically solves a text ranking problem defined as follows: given a query x, a relevance score $s(x, y)$ is computed for each document y in the target collection $\mathcal{D}$. Thus, the text ranking problem can be equivalently cast as a relevance-based binary classification problem (Lin et al., 2020), where for each (x, y) pair, the label is either "relevant" or "not-relevant". A learned probabilistic model can be used to provide the score where $s(x, y) \propto \Pr(\text{relevant}|x, y)$.

Typically, computing the relevance score $s(x, y)$ involves using the content of each document y. For example, the keyword-based approach BM25 (Robertson and Zaragoza, 2009) employs sparse bag-of-words representations of the query x and the content y, $f_q(x)$ and $f_d(y)$, and then $s(x, y)$ is given by the inner product $\langle f_q(x), f_d(y)\rangle$.

Modern deep learning approaches learn a parametric classifier $s_\theta(x, y)$ that takes as input the concatenated content of x and y. Such approaches may be computationally costly since $s_\theta(x, y)$ needs to be evaluated for every $y \in \mathcal{D}$. A two-stage approach is typically employed where a small $\mathcal{D}' \subset \mathcal{D}$ is first retrieved through a fast keyword-based method, then re-ranked with $s_\theta(x, y)$. An alternative to this approach is to learn a dense-representation (e.g. Reimers and Gurevych (2019)) for both f_q and f_d and compute $s(x, y) = \xi(f_q(x), f_d(y))$ where ξ is easy to compute (e.g. the dot product) and $f_d(y)$ can be pre-computed for every $y \in \mathcal{D}$. For simple ξ, the top-scoring documents can be easily retrieved via approximate nearest-neighbors (ANN) techniques. The latter is appealing for real-world applications due to its computational advantage.

While content-based methods have proven effective for general-purpose document ranking and are in widespread use, there are circumstances where using the content of the target documents is less effective. A primary example is when the document content is technical and queries are structurally and linguistically different. Consider the medical domain where documents concern medical treatments. Queries may describe symptoms experienced, which need not be included in a database of treatments. The availability of labeled examples that map symptoms to treatment plans motivates mapping queries from the labeled pairs to the best treatment documents. The same occurs in other domains such as information technology (IT), law, etc. For these technical domains, where such curated historical data often exists, we propose an approach based on direct classification that relies on learning a classifier from the queries themselves to the identity of the target document, without the need for the content of the target documents. The proposed method is complementary to content-based approaches. Hence, to cover potential new queries where similar labeled examples do not exist, we provide an ensemble paradigm, called the Q2R Orchestrator – Q2R stands for "Query-to-Resolution" – to obtain the best of both worlds.

In the proposed approach, each document $y \in \mathcal{D}$ is viewed as a class. We thus have a multiclass classification problem with $|\mathcal{D}|$ classes. In prac-

Proceedings of NAACL-HLT 2022: Industry Track Papers, pages 353 - 361

tice, $|\mathcal{D}|$ can be huge and is likely to increase with time. Therefore, a parametric method for learning a classifier may not be a good fit. We thus propose a nonparametric approach based on kernel K-nearest-neighbors (KNN), which readily handles a growing set of documents, $\mathcal{D}$, and requires only occasional re-tuning.

The KNN approach takes the similarity between a new query x and past queries x' in the training set, rather than the contents of the documents y. In addition to bypassing the problem of using long document content, this approach allows retrieving documents not reachable through content alone. This ability is valuable in application domains such as technical support, where the content may not be well-represented in pretrained language models. The effectiveness of the KNN approach is, however, limited by the availability of labeled training examples and their coverage in terms of the "reachable" documents in $\mathcal{D}$. The Q2R Orchestrator thus combines highly accurate results from KNN for queries where labeled training data is sufficiently similar with a standard content-based retrieval system, for non-similar queries, through a learned orchestrator. Empirically we show that the resulting system benefits from both components.

The three main contributions of Q2R are as follows: (i) Q2R adds a direct classification component to document retrieval based on kernel KNN that enhances the ability to retrieve relevant documents. (ii) Q2R makes use of a labeled data set to train a symmetric query-to-query similarity metric for the kernel KNN, which enhances considerably the system performance, and (iii) Q2R blends the results from the KNN and content-based retrieval methods through an optimized orchestrator.

2 Related Work

For a survey on text ranking, especially modern transformer-based approaches, we refer the reader to Lin et al. (2020). The majority of text-ranking approaches, driven by publicly available datasets such as those from TREC (Voorhees, 2004) and more recently MS MARCO (Nguyen et al., 2016), are content-based. These approaches range from keyword-based, such as BM25 (Robertson and Zaragoza, 2009), to the recent BERT-based (Devlin et al., 2019) models such as re-ranking (Nogueira and Cho, 2019; Dai and Callan, 2019; MacAvaney et al., 2019; Li et al., 2020) and full-ranking with dense-representations (Reimers and Gurevych, 2019; Karpukhin et al., 2020; Khattab and Zaharia, 2020; Xiong et al., 2021).

In terms of ensembling multiple document retrieval approaches, the recent focus has been on the computational cost, where faster techniques are used to pre-filter the large document pool, to be re-ranked by computationally more expensive but more accurate techniques. A good example is the work by Ganhotra et al. (2020), which combines a series of traditional IR techniques with neural approaches.

While content-based approaches benefit greatly from models pre-trained with large corpora, they are at a disadvantage in specialized domains involving technical support documents. In such domains, the "resolution" documents given a query need not have a high relevance score based on the content alone. Document expansion techniques can play a role but often fall short as compared to direct classification, as we demonstrate in this work. To the best of our knowledge, there are no existing works that combine a content-based approach with direct classification as proposed in this work.

The proposed Q2R Orchestrator learns a separate classifier to choose results from either the content-based or the direct classification approaches. Traditional fusion techniques (Fox and Shaw, 1994; Vogt and Cottrell, 1999; Aslam and Montague, 2001) can be used here and in some settings may further improve the retrieval performance. We leave this as possible future work.

3 Method

3.1 Kernel KNN

A key component of Q2R is direct classification through kernel KNN. Let $\mathcal{Z} = \{(x_1, y_1), (x_2, y_2) \ldots, (x_N, y_N)\}$ be the training set of query-document pairs, where x_i is the text of a query and y_i the identifier of the document that was matched to each query in a curated dataset. We emphasize that y_i here refers to the document *identity* only, and not its content. Note that there may be more than one historical document y_i for any given historical query x_i. Furthermore, there are often many examples $x_i, x_j, x_i \neq x_j$ with the same document label $y_i = y_j$; this motivates the use of a kernel-weighted voting paradigm. For now, we assume that a feature function f is given and $f(x) \in \Phi$ is defined for each x, where Φ is a finite-dimensional Euclidean space.

The kernel KNN is a generative model for clas-

sification where the class conditional distributions $p(X|Y)$ are represented by a mixture:

$$p(X=x|Y=y) = \frac{1}{|\mathcal{Z}_y|} \sum_{(x',y')\in\mathcal{Z}_y} \psi\left(f(x) - f(x')\right)$$

where $\mathcal{Z}_y = \{(x', y') \in \mathcal{Z} : y' = y\}$ and $\psi : \Phi \to \mathbb{R}$ is a *kernel* function, with the following properties:

$$\psi(u) \geq 0, \int_\Phi \psi(u)du = 1.$$

A frequently used, smooth kernel function is the Gaussian kernel $\psi(u) \propto \exp\{-\frac{\|u\|^2}{2}\}$.

Given a query x, classification is done based on the posterior, given by:

$$\begin{aligned} & p(Y = y|X = x) \\ & \propto p(X = x|Y = y)p(Y = y) \\ & = \left(\frac{1}{|\mathcal{Z}_y|} \sum_{(x',y')\in\mathcal{Z}_y} \psi\left(f(x) - f(x')\right)\right)\left(\frac{|\mathcal{Z}_y|}{N}\right) \\ & \propto \sum_{(x',y')\in\mathcal{Z}_y} \psi\left(f(x) - f(x')\right). \end{aligned}$$

In practice, the computation of the posterior $p(Y|X)$ is restricted to only the K nearest neighbors of x in the feature space Φ. Let $\mathcal{Z}^K(x) \subset \mathcal{Z}$ be the set of K nearest neighbors of x in Φ based on $f(x)$ and $\mathcal{Z}_y^K(x) = \mathcal{Z}^K(x) \cap \mathcal{Z}_y$, then the kernel KNN relevance score between x and y is defined as

$$s^K(x, y) := \sum_{(x',y')\in\mathcal{Z}_y^K(x)} \psi\left(f(x) - f(x')\right). \quad (1)$$

Here, K is a hyperparameter that is optimized using a separate validation set. The feature function f plays a critical role and is optimized through metric learning on $\mathcal{Z}$ (Section 3.2). Notice that the relevance score s^K between a query x and a document y as defined in (1) depends only on the features of x (the query) and x' (training queries) and never on the contents of the document y.

3.2 Metric Learning

Q2R improves the relevance score s^K in (1) by fixing the kernel ψ and optimizing the feature function f through metric learning. Assume that f is parameterized by $\theta \in \Theta$, and denote the particular instance f_θ. In general, Θ can be a space of neural networks, and f_θ can range from linear to very complex nonlinear mappings.

The objective is to find f such that $f(x)$ is close to $f(x')$ if both (x, y) and (x', y) are in $\mathcal{Z}$. In other words, queries that have the same answers should be close to each other in the feature space. We use the triplet loss (2), a widely used objective function for metric learning (Weinberger and Saul, 2009; Schroff et al., 2015).

The idea is to create a set $\mathcal{T}$ of "triplets" (x_a, x_p, x_n) from $\mathcal{Z}$. Each triplet contains an anchor example x_a, a positive example x_p that belongs to the same class as x_a and a negative example x_n that belongs to a different class. Given $\mathcal{T}$, we find:

$$\begin{aligned} \theta^* = \arg\min_{\theta\in\Theta} \sum_{(x_a,x_p,x_n)\in\mathcal{T}} & \max\{0, 1 + \\ & \|f_\theta(x_a) - f_\theta(x_p)\| - \|f_\theta(x_a) - f_\theta(x_n)\|\}. \end{aligned} \quad (2)$$

For large $\mathcal{Z}$, the number of triplets can be huge. We propose an iterative sampling approach similar to that in Xiong et al. (2021) to optimize f_θ as follows:

1. Initialize θ randomly.
2. Set $\mathcal{T} \leftarrow \emptyset$.
3. For each $(x, y) \in \mathcal{Z}$,
   (a) Sample (x', y') from $\mathcal{Z}_y - \{(x, y)\}$ with weight $\psi(f_\theta(x) - f_\theta(x'))$; let $x_p \leftarrow x'$.
   (b) Sample (x'', y'') from $\mathcal{Z} - \mathcal{Z}_y$ with weight $\psi(f_\theta(x) - f_\theta(x''))$; let $x_n \leftarrow x''$.
   (c) Let $x_a \leftarrow x$, add (x_a, x_p, x_n) to $\mathcal{T}$.
4. Solve (2) for θ^*; let $\theta \leftarrow \theta^*$.
5. Evalute f_θ on validation set. Stop if no improvement after sufficiently many iterations.
6. Otherwise, go to step 2.

In Step 3, note that both the positive and the negative examples are sampled based on their similarities to the anchor example, preferring the more similar ones. Empirically we find that this approach performs better than always choosing a "hard" triplet i.e. picking the most dissimilar positive examples and the most similar negative examples. One reason could be that positive examples form clusters that are far from each other and including such distant examples in the triplet may actually harm the learning process.

Dataset	# URLs	# Training	# Validation	# Test
Twitter	3585	9622	493	472
Telco	1214	31,096	3859	3732
IBM	149,729	433,369	24,082	22,143

Table 1: Specifications of the Data Sets

Step 3 can be repeated for each $(x, y) \in \mathcal{Z}$ to produce multiple triplets. In our implementation, for each anchor, we sample one triplet using the weighted distribution as described above and another triplet using uniform weights. For large $\mathcal{Z}$, one can use a subset of $\mathcal{Z}$ in Step 3. For general neural networks, (2) can be solved using stochastic gradient descent or its variants on minibatches from $\mathcal{T}$.

3.3 The Q2R Orchestrator

As noted in Section 3.1, the content of a document y is never used in the direct classification approach via kernel KNN, only its identity. This relies on the presence of a sufficient number of labeled examples (x, y) in $\mathcal{Z}$ for each $y \in \mathcal{D}$. In practice, this will be possible for some, but not all, y, especially when the collection of documents, $\mathcal{D}$, is large. To provide answers to previously unseen, or under-represented documents in $\mathcal{Z}$, Q2R makes use of a standard content-based retrieval approach in conjunction with the kernel KNN method. This is done through the Q2R Orchestrator.

Suppose that s^C is the relevance score for a content-based approach while s^K is the relevance score for kernel KNN described above. Suppose that the objective is to return the top R documents. For a given query x, let $\mathcal{Y}^C = \{y^C_{(1)} \dots y^C_{(R)}\} \subset \mathcal{D}$ be the top R documents based on s^C and respectively $\mathcal{Y}^K = \{y^K_{(1)} \dots y^K_{(R)}\} \subset \mathcal{D}$ the top R documents based on s^K. The question of interest is formulated as a binary decision: decide whether to select $\mathcal{Y}^C$ or $\mathcal{Y}^K$ as the set of results to provide to the user. The Q2R Orchestrator thus trains a binary classifier to make this decision. For efficiency we use a linear classifier trained using logistic regression. To construct the training set, we identify examples in the validation set where the ground truth is contained in $\mathcal{Y}^C$ or $\mathcal{Y}^K$, but not both. The input features for the classifier include minimally $(s^C(x, y^C_{(1)}), \dots, s^C(x, y^C_{(R)}), s^K(x, y^K_{(1)}), \dots, s^K(x, y^K_{(R)}))$, and may include other features such as confidence intervals.

4 Experiments

4.1 Data Sets

We focus on the task of natural-language retrieval of technical documents. We evaluate the proposed method along with baseline methods on datasets in which labeled examples are available.[1] In particular, we use the Twitter and Telco datasets described in Ganhotra et al. (2020) along with an IBM dataset.

The Twitter dataset is publicly available. It contains 10,587 labeled examples. Each example consists of a sequence of dialog messages and a URL document as the answer label. The set is split $90\%/5\%/5\%$ for train/validation/test, respectively. The Telco set contains 38,687 examples, with an $80\%/10\%/10\%$ train/validation/test split. The IBM set is an order of magnitude larger than the Telco set with a $90\%/5\%/5\%$ train/validation/test split. Table 1 summarizes the data sets used in our experiments.

4.2 Models

Q2R allows for virtually any content-based method to be used in conjunction with the kernel KNN component. Furthermore, thanks to the availability of training examples, the content-based approach itself can be improved by augmenting the content of the documents with the labeled examples using text from the corresponding training examples (Amitay et al., 2005). We show that this augmentation improves considerably the performance of the content-based approaches.

Here, as a baseline content-based method, we use BM25 (Robertson and Zaragoza, 2009). The variant wherein each document $y \in \mathcal{D}$ is augmented with text from $\{x : (x, y) \in \mathcal{Z}^{\text{Train}}\}$ is referred to as *BM25-aug*. We also include results obtained using the information retrieval method proposed by Ganhotra et al. (2020) (IRC, short for IR-Cascade) as well as ESIM (Chen et al., 2017). In addition, we examine a number of content-based

[1] The documents in the IR data sets Robust04 and MS MARCO are not sufficiently well-covered by the training set for use with Q2R.

methods based on Sentence-BERT (Reimers and Gurevych, 2019). We fine-tuned a pretrained DistilBERT model (Sanh et al., 2019) on our data sets using the triplet loss. To deal with long documents, we use a similar technique as in Dai and Callan (2019). We evaluate the following variants:

- SBERT-First (SB-F): Only the beginning of each document is used, up to the maximum sequence length of the model.
- SBERT-MaxP (SB-M): Each document is segmented into overlapping sliding windows. For training, each sub-document is assumed relevant. For evaluation, the maximum relevance score over all sub-documents is used.
- SBERT-aug-MaxP (SB-aug): SBERT-MaxP with augmented content as described above.

For our proposed KNN-based direct-classification component, we use kernel KNN with the following similarity metrics:

- (BOW) A simple TF-IDF weighted bag-of-words (BOW) representation for $f(\cdot)$ and $\psi(u) \propto 1 - \frac{1}{2}\|u\|^2$. Each feature vector $f(x)$ is normalized such that $\|f(x)\| = 1$, in which case it is straightforward to see that $\psi(f(x) - f(x')) = \langle f(x), f(x') \rangle$.
- (LinNet) We use (2) to train a linear transformation (LinNet) that maps the BOW vectors to a low-dimensional space. The feature dimension is a hyperparameter optimized on a validation set. We use dimension 200 throughout. For ψ we use the Gaussian kernel.
- (Transformers) For $f(x)$, fine-tune a pretrained transformer architecture using (2). We use DistilBERT (Sanh et al., 2019) (labeled KNN-DB) and MPNET (Song et al., 2020) (labeled KNN-MP), both with final 768-dimensional feature vectors. For ψ, we again use the Gaussian kernel.

For the kernel KNN models, we use a validation set to select the number of neighbors, $K \in \{5, 10, 20, 40, 80, 160, 320, 640\}$. The nearest-neighbor search can be done via an index by approximate-KNN (Malkov and Yashunin, 2020), and is as such nearly as fast as BM25.

	MRR	Recall@ 1	3	5
Content-based				
BM25	0.079	0.051	0.087	0.114
BM25-aug	0.498	0.403	0.561	0.629
IRC	0.498	0.417	0.547	0.606
ESIM	0.380	0.261	0.460	0.519
SB-F-0	0.030	0.015	0.028	0.042
SB-M-0	0.028	0.013	0.028	0.042
SB-aug-0	0.299	0.220	0.333	0.409
SB-F	0.449	0.375	0.489	0.545
(A) SB-M	0.482	0.384	0.536	0.598
(B) SB-aug	0.546	0.449	0.600	**0.663**
Kernel KNN (ours)				
BOW	0.477	0.409	0.525	0.568
LinNet	0.504	0.441	0.559	0.619
(C) KNN-DB	0.542	0.462	0.612	**0.661**
Q2R Orchestrator (ours)				
(A)+(C)	**0.557**	0.466	**0.621**	**0.665**
(B)+(C)	**0.552**	**0.473**	0.608	0.657

Table 2: Results on the Twitter set

4.3 Results

The results for the Twitter set are shown in Table 2, in terms of both the Mean Reciprocal Rank (MRR) as well as Recall@R for $R \in \{1, 3, 5\}$. The results reported in Ganhotra et al. (2020) were obtained by restricting the answer set to URLs from the same company/domain; here we use the full URL set, making the problem more challenging.

For the content-based approach, document-expanded BM25-aug and IRC far outperform vanilla BM25. Also included are results for the three "SB-x-0" variants, which use the pretrained DistilBERT model without any fine-tuning on the Twitter training set. Again, we see that augmentation makes a huge difference. The best-performing content-based approach is the fine-tuned SB-aug. The KNN classification methods based on bag-of-words perform similarly to the content-based methods but are outperformed by KNN-DB. Finally, it is clear that combining both approaches with the Q2R orchestrator results in the best performance.

Table 3 shows a breakdown of the results based on training-set coverage of the ground truth URLs. For each test query, we use the term "training coverage" to refer to the number of training examples that share the same ground truth URL. In classifier terms this is equivalent to the size of the training

Training coverage	0	1-9	10-55	56-338	339+
# Test queries	96	93	100	91	92
Content-based					
BM25	0.075	0.131	0.160	0.021	0.001
BM25-aug	0.046	0.373	0.531	0.815	0.749
IRC	0.022	0.354	0.507	0.711	0.917
ESIM	0.063	0.149	0.371	0.559	0.776
SB-F	0.144	0.363	0.453	0.417	0.884
(A) SB-M	**0.149**	0.351	0.447	0.577	0.904
(B) SB-aug	0.089	**0.382**	0.612	0.782	0.884
Kernel KNN (ours)					
BOW	0.000	0.268	0.420	0.817	0.913
LinNet	0.000	0.311	0.485	0.863	0.891
(C) KNN-DB	0.000	0.269	**0.656**	**0.879**	0.923
Q2R Orchestrator (ours)					
(A)+(C)	0.090	0.313	0.608	0.859	**0.937**
(B)+(C)	0.069	0.368	0.602	0.815	**0.930**

Table 3: MRR by training coverage on Twitter set. Training coverage of 0 means documents not in training set.

	MRR	Recall@ 1	3	5
Content-based				
BM25	0.033	0.012	0.034	0.049
BM25-aug	0.337	0.201	0.408	0.510
(A) IRC	0.444	0.294	0.532	0.633
ESIM	0.458	0.299	0.554	0.658
(B) IRC+E	0.481	0.327	0.596	**0.691**
SB-F	0.428	0.280	0.519	0.616
SB-M	0.432	0.287	0.524	0.612
SB-aug	0.420	0.284	0.511	0.596
Kernel KNN (ours)				
BOW	0.484	0.346	0.569	0.656
(C) LinNet	0.496	0.370	0.599	0.665
KNN-DB	0.454	0.321	0.546	0.627
(D) KNN-MP	0.493	0.360	0.591	0.661
Q2R Orchestrator (ours)				
(A)+(C)	0.496	0.371	0.598	0.664
(A)+(D)	**0.510**	0.372	0.603	0.676
(B)+(D)	**0.515**	**0.381**	**0.623**	**0.694**

Table 4: Results on the Telco set

set with the same label. Training coverage of 0 means no such document was in the training set.

Naturally, for kernel KNN classification, we expect higher recall performance for queries with larger training coverage. This can be observed in Table 3 for all kernel-KNN models which far outperform on queries with coverage more than 10. On the other hand, kernel KNN has 0-recall for documents with 0 coverage since no neighbors could vote for such documents. The content-based approaches are able to perform on such queries. Q2R benefits from both approaches, performing well on queries with both high and low training coverage.

MRR	Train	Validation	Test
SB-F	0.498	0.433	0.428
SB-M	0.506	0.438	**0.432**
SB-aug	0.841	0.430	0.420

Table 5: Investigating Sentence-BERT performance on the Telco set.

Table 4 shows the results for the Telco set. The Telco set breakdown by training coverage is provided in the Appendix in Table 8. We see that among content-based approaches, the BERT-based approaches are no longer superior. Interestingly, SB-aug performs worse than SB without augmentation. Table 5 reveals that even though the final model is picked based on the validation-set results, there may be overfitting on the training set. Amongst the KNN models, LinNet performs the best overall, slightly better than the transformer-based model. However, the breakdown in Table 8 shows that each excels in different subsets of test queries. Q2R, combining IRC and KNN-MP results in better performance than combining IRC and LinNet.

Train. coverage	0	1-2	3-4	5-9	10-18	19-36	37-71	72-163	164-467	471+
# Test queries	2457	2649	1576	2255	2243	2170	2191	2194	2204	2204
Content-based										
(A) COMBL	**0.166**	**0.118**	0.108	0.093	0.088	0.078	0.079	0.080	0.071	0.060
BM25	0.111	0.087	0.089	0.076	0.069	0.068	0.075	0.074	0.067	0.048
(B) BM25-aug	0.077	0.100	**0.128**	**0.164**	**0.198**	0.225	0.294	0.303	0.302	0.262
(C) IRC	0.002	0.070	0.110	0.143	0.187	0.221	0.290	0.340	0.404	0.418
Kernel KNN (ours)										
BOW	0.000	0.024	0.058	0.103	0.154	0.200	0.295	0.357	0.465	0.647
(D) LinNet	0.000	0.032	0.069	0.115	0.190	**0.246**	**0.351**	**0.412**	**0.519**	**0.689**
Q2R Orchestrator (ours)										
(A)+(D)	0.050	0.052	0.081	0.117	0.188	0.241	0.345	0.409	**0.513**	**0.683**
(B)+(D)	0.012	0.039	0.073	0.116	0.189	**0.245**	**0.349**	**0.412**	**0.517**	**0.686**
(C)+(D)	0.000	0.032	0.070	0.115	0.190	**0.246**	**0.351**	**0.413**	**0.519**	**0.688**
(A)+(D) W.	0.139	0.100	0.104	0.116	0.148	0.177	0.247	0.306	0.401	0.578
(B)+(D) W.	0.073	0.097	0.122	**0.158**	**0.199**	0.228	0.304	0.341	0.394	0.515

Table 6: MRR by training coverage on the IBM set. Coverage of 0 means documents not in training set.

	MRR	Recall@ 1	3	5
Content-based				
(A) COMBL	0.095	0.060	0.107	0.134
BM25	0.077	0.044	0.086	0.113
(B) BM25-aug	0.204	0.131	0.234	0.287
(C) IRC	0.216	0.153	0.247	0.291
Kernel KNN (ours)				
BOW	0.228	0.162	0.256	0.311
(D) LinNet	0.260	0.186	0.298	0.352
Q2R Orchestrator (ours)				
(A)+(D)	**0.266**	**0.194**	**0.306**	**0.358**
(B)+(D)	0.261	0.186	0.299	0.353
(C)+(D)	0.260	0.186	0.298	0.351
(A)+(D) W.	0.231	0.168	0.266	0.315
(B)+(D) W.	0.242	0.162	0.273	0.327

Table 7: Results on the IBM set

Finally, Tables 6 and 7 show the results on the large IBM dataset. LinNet is the best-performing model for the kernel KNN. The transformer-based models are too computationally-costly and are not considered competitive; in addition, the transformer-based model results are inferior to those presented in the table. For content-based approaches on the IBM set, we include an alternative to BM25, based on keyword enrichment, labeled COMBL.

One important observation for this dataset is that the KNN models significantly outperform the content-based approaches in terms of overall average performance. This results in a significantly unbalanced training set for the orchestrator, where most examples would favor choosing the KNN results. The Q2R orchestrator can thus be trained using a weighted loss such that examples where the content-based model should be selected are given more weight. We tag this weighted version with "W." in the tables. Observe that the unweighted hybrid methods perform the best overall, but from Table 6 we see that the weighted version gives a more balanced performance across queries with different levels of training coverage.

5 Conclusion

We presented the Q2R system aimed at providing relevant documents in response to technical queries in natural language. The key novelty in this system is its use of both content-based document retrieval techniques as well as the proposed kernel KNN approach, which taps into the available labeled data from historical queries. Our experimental results show that content-based document retrieval and the kernel KNN approach complement each other; Q2R is able to take advantage of both. The system has been deployed at IBM on various applications that involve natural language queries and has shown encouraging performance improvement over existing systems. Potential future enhancements include more sophisticated sampling procedures for the metric-learning, as well as new fusion approaches for the orchestrator.

References

Einat Amitay, Adam Darlow, David Konopnicki, and Uri Weiss. 2005. Queries as anchors: Selection by association. In *Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '05, page 193–201, New York, NY, USA. Association for Computing Machinery.

Javed A Aslam and Mark Montague. 2001. Models for metasearch. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 276–284.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 985–988, New York, NY, USA. Association for Computing Machinery.

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 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Edward A Fox and Joseph A Shaw. 1994. Combination of multiple searches. *NIST special publication SP*, 243.

Jatin Ganhotra, Haggai Roitman, Doron Cohen, Nathaniel Mills, R. Chulaka Gunasekara, Yosi Mass, Sachindra Joshi, Luis A. Lastras, and David Konopnicki. 2020. Conversational document prediction to assist customer care agents. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 349–356. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. *ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT*, page 39–48. Association for Computing Machinery, New York, NY, USA.

Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. Parade: Passage representation aggregation for document reranking. *arXiv preprint arXiv:2008.09093*.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2010.06467*.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1101–1104, New York, NY, USA. Association for Computing Machinery.

Y. A. Malkov and D. A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(04):824–836.

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

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823. IEEE Computer Society.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867. Curran Associates, Inc.

Christopher C Vogt and Garrison W Cottrell. 1999. Fusion via a linear combination of scores. *Information retrieval*, 1(3):151–173.

Ellen M. Voorhees. 2004. Overview of the trec 2004 robust retrieval track. In *In Proceedings of the Thirteenth Text REtrieval Conference (TREC2004)*, page 13.

Kilian Q. Weinberger and Lawrence K. Saul. 2009. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

A Appendix: Privacy

Since our training data includes past queries, it is important to remove all sensitive or personal information from the raw text before using them for training. We employ both automated masking and filtering followed by manual human tagging (for truly sensitive queries) as a preprocessing step in our data preparation pipeline.

B Appendix: Additional Experiment Details

All our models are trained using single-GPU (NVidia V100) 16-core machines with 128GB RAM. The total training time for each model varies from a few minutes (Twitter set, BM25) to a few days (larger BERT-based models).

For bag-of-words models, we trim the vocabulary by removing rare words and stop words, to 2000, 5600 and 54,000 words respectively for Twitter, Telco and IBM data sets.

Our BERT-based models use pretrained DistilBERT ('distilbert-base-nli-mean-tokens') (model size 253MB) and MPNET ('all-mpnet-base-v2') (model size 418MB).

For most results, the variance due to approximate-NN is small so we omitted them. For results based on neural-network training, we report averages based on at least 3 runs.

C Appendix: Additional Results

In Table 8 we present the Telco set breakdown results by training coverage.

Training coverage	0-59	63-213	216-1557	2009-2388	4165
# Test queries	757	759	887	786	543
Content-based					
BM25	0.07	0.05	0.03	0.01	0.00
BM25-aug	0.17	0.31	0.40	0.37	0.47
IRC (A)	**0.24**	**0.44**	0.56	0.41	0.58
ESIM	0.17	0.31	0.54	0.59	0.75
IRC+E (B)	0.17	0.36	0.58	0.61	0.75
SB-F	0.22	0.30	0.47	0.50	0.72
SB-M	0.23	0.30	0.48	0.56	0.64
SB-aug	0.21	0.31	0.48	0.52	0.63
Kernel KNN (ours)					
BOW	0.15	0.28	0.52	**0.70**	0.87
LinNet (C)	0.15	0.35	0.56	0.63	**0.89**
KNN-DB	0.11	0.29	0.52	0.60	0.83
KNN-MP (D)	0.15	0.35	0.61	0.65	0.77
Q2R Orchestrator (ours)					
(A)+(C)	0.17	0.37	0.55	0.60	**0.88**
(A)+(D)	0.21	0.42	0.62	0.60	0.75
(B)+(D)	0.15	0.37	**0.63**	0.68	0.80

Table 8: MRR by training coverage on Telco set. Training coverage of 0 means documents not in training set.