Attentive Neural Architecture for Ad-hoc Structured Document Retrieval

Saeid Balaneshinkordan
saeid@wayne.edu
Wayne State University
Detroit, MI, USA

Alexander Kotov
kotov@wayne.edu
Wayne State University
Detroit, MI, USA

Fedor Nikolaev
fedor@wayne.edu
Wayne State University
Kazan Federal University
Kazan, Russian Federation

ABSTRACT

The problem of ad-hoc structured document retrieval arises in many information access scenarios, from Web to product search. Yet neither deep neural networks, which have been successfully applied to ad-hoc information retrieval and Web search, nor the attention mechanism, which has been shown to significantly improve the performance of deep neural networks on natural language processing tasks, have been explored in the context of this problem. In this paper, we propose a deep neural architecture for ad-hoc structured document retrieval, which utilizes attention mechanism to determine important phrases in keyword queries as well as the relative importance of matching those phrases in different fields of structured documents. Experimental evaluation on publicly available collections for Web document, product and entity retrieval from knowledge graphs indicates superior retrieval accuracy of the proposed neural architecture relative to both state-of-the-art neural architectures for ad-hoc document retrieval and probabilistic models for ad-hoc structured document retrieval.

CCS CONCEPTS

• **Information systems** → *Retrieval models and ranking*;

KEYWORDS

Deep Neural Networks, Attention, Structured Document Retrieval

ACM Reference Format:
Saeid Balaneshinkordan, Alexander Kotov, and Fedor Nikolaev. 2018. Attentive Neural Architecture for Ad-hoc Structured Document Retrieval. In *2018 ACM Conference on Information and Knowledge Management (CIKM'18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271801

1 INTRODUCTION

The problem of structured (or multi-field) document retrieval (SDR) arises in many information access scenarios, from Web search to entity retrieval from a knowledge graph. However, previous information retrieval (IR) research has largely viewed documents as holistic and homogeneous fragments of text. Retrieval models that adopt this holistic view aim at quantifying relevance through aggregation of a small number of relevance heuristics, such as the number and proximity of occurrences of query terms, their collection statistics and document length. Accounting for document structure requires additional strategies for aggregating these heuristics, which are calculated at the level of document fields, into the matching score of an entire document. Such aggregation is informed by a relative importance of document fields, which can be inferred from either the properties of document fields (e.g. a query term matched in a field of a Web page with the terms highlighted in larger font should have a different importance than a query term matched in other fields) or query intent (e.g. in the query "attractive outdoor light with security features", "attractive" refers to the product description field, "outdoor light" to the product name field and "security features" to the product attributes field). The ability to take into account document structure makes SDR methods particularly effective for retrieving documents with lexically similar, but semantically diverse fields (e.g. descriptions of products or entities in a knowledge graph).

SDR methods face three major challenges: i) identifying the key phrases in keyword queries ii) semantic matching of the key query phrases in different fields of structured documents iii) aggregating the scores of the matched query phrases into the overall score of a structured document. Different probabilistic and language modeling based models for structured document retrieval [14, 23, 24, 28, 45] have been proposed over the past decade to address some of these challenges. However, all these methods are based on direct matching of terms and phrases in queries and documents, which may lead to the issue of *lexical gap*, when semantically similar concepts are communicated using different words and phrases in queries and relevant documents. To address this issue, several neural architectures have been recently proposed for *ad-hoc document retrieval* [7–9, 30, 38, 40] and *Web search* [6, 12, 21, 32, 44]. These architectures typically take dense continuous vector representations (i.e. embeddings) of words in queries and documents as input and quantify relevance by applying a series of non-linear transformations to those representations to obtain the document retrieval score. A combination of word embeddings and representation learning for semantic matching endows neural architectures with the ability to effectively address the issue of lexical gap without relying on traditional IR approaches, such as dimensionality reduction [16], query [3] or document [2] expansion. Incorporation of additional retrieval heuristics, such as inverse document frequency (IDF) of query terms, and features, such as retrieval scores of bag-of-words retrieval models and word overlap, have been shown

CIKM'18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271801

to improve the retrieval accuracy of neural architectures even further [9]. However, *deep neural architectures for retrieval of structured documents are only starting to be explored.* The only such architecture that we are aware of was proposed by Zamani et al. [44], which is the most relevant paper to our work. However, their model is tailored to artifacts specific to Web search, such as varying number of document fields, and utilizes user behavior data, such as clicks, in document representation and for training.

This paper takes a different route and focuses on exploring the utility of the attention mechanism, which has been previously shown to be effective in natural language processing [1, 29] and computer vision [39] tasks, in modeling important aspects of relevance in structured document retrieval, such as dynamic importance of query phrases and document fields. In particular, we propose **A**ttention-based **N**eural Architecture for Ad-hoc **S**tructured Document **R**etrieval (ANSR)[1], which attends to the most important query phrases and fields of structured documents in order to quantify their relevance to a keyword query. Based on the interaction matrices between phrases in a query and document fields, our proposed architecture computes the matching score of document fields with respect to a query at multiple levels of granularity and aggregates those scores into the matching score of a structured document. Our proposed architecture also incorporates collection statistics to dynamically attend to different query phrases and fields of a document based on a given query.

This work provides the following contributions:

- To the best of our knowledge, we present the first deep neural architecture for ad-hoc SDR, which can be used in the absence of user interaction data for training. The proposed architecture addresses the issue of lexical mismatch, which plagues traditional probabilistic and language modeling-based ad-hoc SDR models.
- We experimentally demonstrate the effectiveness of attention mechanism within neural ranking models for structured documents at identifying important query phrases and document fields.

Roadmap. The rest of this paper is organized as follows. Section 2 provides an overview of the prior work on structured document retrieval and neural networks for IR. Our proposed deep neural architecture for structured document retrieval is presented in Section 3. Results of experimental evaluation of the proposed architecture on multiple datasets are presented in Section 4. Section 5 concludes the paper.

2 RELATED WORK

In this section, we provide an overview of previous work on probabilistic and machine learning-based models for SDR, deep learning for IR and attention mechanism in neural architectures.

Probabilistic models for SDR. Numerous probabilistic and language modeling-based models, such as BM25F [28], Mixture of Language Models (MLM) [24], Probabilistic Retrieval Model for Semistructured Data (PRMS) [14], as well as machine learning-based methods [23, 26, 34, 36, 37, 45] have been proposed for structured document retrieval over the past decade. While early methods [24, 26, 28, 34, 36, 37] assume bag-of-words document representation, more recent methods, such as Fielded Sequential Dependence Model (FSDM) [45] and Parameterized Fielded Sequential Dependence Model (PFSDM) [23], additionally take into account sequential dependencies between the query terms. These methods compute the matching score of structured documents in two ways: (1) as a weighted linear combination of field scores computed by a standard retrieval model, such as BM25 [28], query likelihood [24] or sequential dependence model [45] (2) directly based on the weighted linear combination of retrieval heuristics, such as query term frequency in each document field [28].

Neural Architectures for Text Matching and Document Ranking. The recent success of deep learning has revitalized research on semantic text matching and document ranking. The pioneering work on neural architectures for semantic text matching, such as DeepMatch [17], ARC-I/ARC-II [11], DCNN [13], MultiGranCNN [42] and MatchPyramid [25] laid a foundation for subsequent efforts aimed at developing neural architectures for document ranking in Web search [6, 12, 21, 32, 44] and ad-hoc IR [7–9, 30, 38, 40]. The two major classes of these architectures are *representation-based* and *interaction-based.*

DSSM [12] is an example of a representation-based architecture and one of the first neural architectures for Web search. DSSM takes word hashing based representations of a query and documents as input and employs a Siamese architecture consisting of multiple fully connected layers to learn semantic representations of queries and documents, which are then used to measure relevance between them with cosine similarity. Similar to DSSM, ARC-I [11] finds representations of texts with a Siamese architecture consisting of multiple convolutional layers and uses these representations as input to a multi-layer perceptron (MLP), which determines the degree of semantic match. Other representation-based neural architectures for text matching or document ranking include CDSSM [32] and DCNN [13] .

Interaction-based architectures take a matrix of similarities between embeddings of all pairs of words in a document and a query or in a pair of texts as input. DeepMatch [17] is an example of interaction-based architecture, which models the matching score between a pair of texts as a hierarchy of semantic similarity scores between all pairs of words in them. ARC-II [11], similar to DeepMatch, computes the matching score based on the interaction space between a pair of texts. It was shown in [11] that ARC-II, an interaction-based architecture, has a superior retrieval accuracy than ARC-I, a representation-based architecture, which was attributed to the effectiveness of local matching patterns between a pair of texts. Other interaction-based neural architectures for text matching or document ranking include MatchPyramid [25], MultiGranCNN [42], DRMM [9], K-NRM [38] and Conv-KNRM [7].

ANSR, the proposed architecture, can also be categorized as interaction-based, however, unlike other interaction- or representa-tion-based neural architectures for IR [7, 9, 12, 32, 38], ANSR takes into account document structure.

Attention and Gating in Neural Networks. The attention mechanism, which allows neural architectures to focus on the most important parts of contextual information, was first proposed for machine translation [1] and later successfully utilized in neural architectures for other tasks, such as image [39] and video captioning [43], text summarization [29] as well as question answering [40].

[1] source code available at https://github.com/teanalab/ANSR

Yang et al. [41] proposed a two-level Hierarchical Attention Network, which allows to attend differently to words and sentences. The attention-based neural architecture for matching questions with answers proposed by Yang et al. [40] combines interaction matrix between question and answer with the attention scheme based on a softmax gating function to model the importance of question terms.

Inspired by the gating mechanism in Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures, neural networks in [4, 9, 31, 33, 35] include gating units to regulate information flow. Gated neural networks have also been utilized in a variety of information retrieval scenarios. For example, Sheldon et al. [31] introduced a gated neural network based score combination function that weighs different query reformulations when merging them into the final ranked list. Guo et al. [9] proposed a neural ranking model, which utilized a gating function to weigh the contributions of matching scores of query terms towards the matching score of an entire document based on their IDF, and demonstrated that such gating results in significant improvement of retrieval accuracy.

3 METHOD

3.1 Overview

ANSR quantifies the relevance of a structured document to a given keyword query by aggregating semantic matching scores at different resolution levels. Semantic matching scores between query phrases and document fields at a lower resolution level are transformed into the matching scores at a higher resolution level by fully connected attention and aggregation layers until the final document relevance score is obtained. First, ANSR computes semantic matching scores of two types of query phrases in each of the document fields. Query phrases of the first type consist only of query terms (i.e., unigrams), while query phrases of the second type consist of two-word (i.e. sequential) bigrams. Second, the matching scores of each unigram- and bigram-based query phrase in different document fields are aggregated into a single matching score based on the relative importance weights of different document fields, which are determined by the *document field attention layer*. Third, semantic matching scores of all unigram- and bigram-based query phrases are combined into two aggregate matching scores for these types of query phrases. Finally, the aggregate matching scores of unigram and bigram-based query phrases are combined into the final relevance score of an entire document based on the relative importance weights of these two types of query phrases, which are determined by the *query phrase attention layer*. Accounting for both unigram and bigram query concepts has been previously shown to improve retrieval accuracy for both ad-hoc [19] and structured document retrieval [45].

The input to the matching score aggregation component of ANSR shown in Figures 1 is a set of similarity matrices representing interactions between distributed representations of a query and each one of the n^f document fields. There are several options for representing queries and document fields and computing similarity matrices, which differ in computational complexity and the number of parameters in the resulting neural architecture. The simplest way is to represent queries and document fields with embeddings of their constituent words and compute one similarity matrix per each document field, in which the entries are cosine similarities between embeddings of all pairs of words in a query and a document field. However, in this case, the dimensions of similarity matrices and the number of trainable parameters in the resulting neural architecture will vary with the size of the corresponding document fields.

Table 1: Table of Notations

Notation	Definition
n^f	number of fields in document d
n^q	number of n-grams in query q
n^l	number of n-grams in the lth document field
$\hat{\mathbf{r}}^q$	compressed distributed representation of q
$\hat{\mathbf{r}}^q_i$	ith embedding vector in $\hat{\mathbf{r}}^q$
$\hat{\mathbf{r}}^l$	compressed distributed representation of the lth document field
$\hat{\mathbf{r}}^l_j$	jth embedding vector in $\hat{\mathbf{r}}^l$
$\hat{\mathbf{M}}^l$	compressed similarity matrix between a query and the lth document field
$\hat{\mathbf{M}}^l_{i,:}$	ith row of $\hat{\mathbf{M}}^l$
$\hat{n}^f$	number of compressed similarity matrices
$\hat{n}^q$	number of rows in compressed similarity matrices
$\hat{n}^l$	number of columns in compressed similarity matrices
ξ	number of dimensions in word embedding vectors
$m(\cdot,\cdot)$	the matching score between two objects

To address the issue of varying sizes of query-document field similarity matrices and decrease the number of trainable parameters, ANSR includes the pooling component, which creates *compressed similarity matrices* of the same fixed dimensions for each document field, as described in Section 3.2. The entries in each compressed similarity matrix are pair-wise similarity metrics between embedding vectors in *compressed distributed representations* of a query and a document field. Each embedding vector in these compressed representations is created by combining the embeddings of several adjacent unigrams and bigrams in a query or a document field and can be viewed as corresponding to a phrase in a query or a document field.

Compressed similarity matrices constructed by the pooling component of ANSR are used as input to the matching score aggregation and attention components, which will be described in more detail in Section 3.3. By considering the matching scores between compact representations of a query and document fields and employing weight sharing, ANSR minimizes the number of trainable parameters to enable effective training with a limited amount of training data, which is typical for ad-hoc IR scenarios. Next, we discuss pooling, aggregation and attention components of ANSR in more detail.

3.2 Pooling component

The output of the pooling component consists of two compressed similarity matrices $\hat{\mathbf{M}}^l_u \in \mathbb{R}^{\hat{n}^q \times \hat{n}^l}$ and $\hat{\mathbf{M}}^l_b \in \mathbb{R}^{\hat{n}^q \times \hat{n}^l}$, where $\hat{n}^q \leq n^q$ and $\hat{n}^l \leq n^l$, for each document field l. These matrices are created from unigram- ($\hat{\mathbf{u}}^q$ and $\hat{\mathbf{u}}^l$) and bigram-based ($\hat{\mathbf{b}}^q$ and $\hat{\mathbf{b}}^l$)

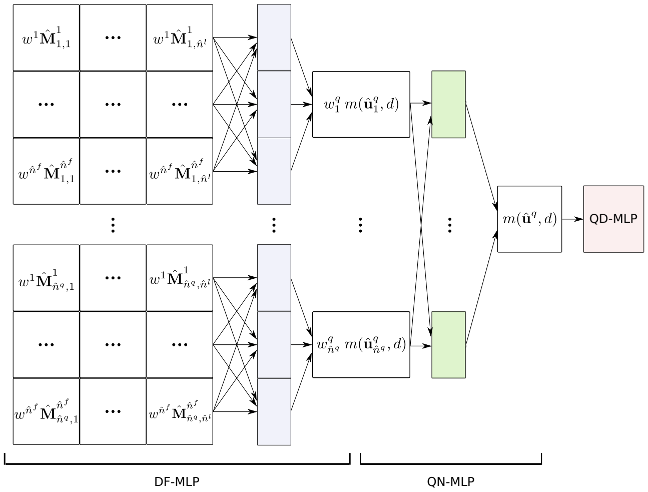

$w^1\hat{\mathbf{M}}^1_{1,1}$
$w^1\hat{\mathbf{M}}^1_{1,\hat{n}^l}$
$w^{\hat{n}^f}\hat{\mathbf{M}}^{\hat{n}^f}_{1,1}$
$w^{\hat{n}^f}\hat{\mathbf{M}}^{\hat{n}^f}_{1,\hat{n}^l}$
$w^q_1\, m(\hat{\mathbf{u}}^q_1, d)$
$m(\hat{\mathbf{u}}^q, d)$
QD-MLP
$w^1\hat{\mathbf{M}}^1_{\hat{n}^q,1}$
$w^1\hat{\mathbf{M}}^1_{\hat{n}^q,\hat{n}^l}$
$w^{\hat{n}^f}\hat{\mathbf{M}}^{\hat{n}^f}_{\hat{n}^q,1}$
$w^{\hat{n}^f}\hat{\mathbf{M}}^{\hat{n}^f}_{\hat{n}^q,\hat{n}^l}$
$w^q_{\hat{n}^q}\, m(\hat{\mathbf{u}}^q_{\hat{n}^q}, d)$
DF-MLP
QN-MLP

Figure 1: Sub-network of the matching score aggregation component of ANSR for unigram-based compressed distributed query representation. Sub-network for calculating the matching score of bigram-based query representation has similar structure and is not shown.

compressed distributed representations of a query and the lth document field, respectively. The matching score aggregation and attention components of ANSR that process either of these two similarity matrices are identical. Therefore, we will henceforth use the notation $\hat{\mathbf{r}}^q$ to refer to the compressed n-gram (unigram or bigram) based distributed query representation, $\hat{\mathbf{r}}^l$ to refer to the compressed n-gram based distributed representation of the lth document field and $\hat{\mathbf{M}}^l$ to refer to the compressed similarity matrices for the lth document field.

Each compressed similarity matrix $\hat{\mathbf{M}}^l$ is created in two steps. In the first step, the pooling functions f_q and f_d create fixed-size compressed distributed representations for a query and the lth document field, respectively. The functions f_d and f_q replace a set of embeddings of consecutive n-grams (unigrams or bigrams) within the windows of size $\lceil n^l/\hat{n}^l \rceil$ and $\lceil n^q/\hat{n}^q \rceil$ in a query and the lth document field, respectively, with a single embedding vector obtained by averaging the embeddings of n-grams within these windows. Specifically, function f_d:

$$f_d : \mathbb{R}^{n^l \times \xi} \mapsto \mathbb{R}^{\hat{n}^l \times \xi}, \quad \forall l \in \{1, \dots, n^f\}, \tag{1}$$

compresses the embeddings of n^l consecutive n-grams in the lth document field into $\hat{n}^l$ embeddings $\hat{\mathbf{r}}^l_j$, such that each $\hat{\mathbf{r}}^l_j$ is an average over $\lceil n^l/\hat{n}^l \rceil$ consecutive n-grams. Similarly, function f_q:

$$f_q : \mathbb{R}^{n^q \times \xi} \mapsto \mathbb{R}^{\hat{n}^q \times \xi}, \tag{2}$$

compresses the embeddings of n^q consecutive query n-grams into $\hat{n}^q$ embedding vectors $\hat{\mathbf{r}}^q_i$, such that each $\hat{\mathbf{r}}^q_i$ is an average over $\lceil n^q/\hat{n}^q \rceil$ consecutive n-grams.

$\hat{\mathbf{M}}^l_{i,j}$, the (i, j)th entry of $\hat{\mathbf{M}}^l$ corresponds to cosine similarity between $\hat{\mathbf{r}}^q_i$ and $\hat{\mathbf{r}}^l_j$:

$$\hat{\mathbf{M}}^l_{i,j} = \frac{\hat{\mathbf{r}}^q_i\ \hat{\mathbf{r}}^l_j}{\left\|\hat{\mathbf{r}}^q_i\right\|_2 \left\|\hat{\mathbf{r}}^l_j\right\|_2} \tag{3}$$

Alternatively, the entries of $\hat{\mathbf{M}}^l$ can be viewed as semantic similarities between phrases in a query and a document field. The ability to account for interactions between phrases in queries and documents sets ANSR apart from the traditional methods for document [19] and structured document [23, 45] retrieval, which consider only unigrams and bigrams.

If $n^l < \hat{n}^l$, zero-padding is performed by adding $\hat{n}^l - n^l$ zero columns to $\hat{\mathbf{M}}^l$. If a document d is missing field l, $\hat{\mathbf{M}}^l$ with zero entries is created. In the end, the pooling component of ANSR creates $\hat{n}^f$ compressed similarity matrices with $\hat{n}^q$ rows and $\hat{n}^l$ columns.

3.3 Matching score aggregation and attention components

The relevance score $s(q, d)$ of query q to document d based on compressed similarity matrices $\{\hat{\mathbf{M}}^1, \dots, \hat{\mathbf{M}}^{\hat{n}^f}\}$ for each document field is computed by matching score aggregation as well as document field and query phrase attention components of ANSR. Each aggregation or attention component consists of one fully-connected hidden layer and one output layer. tanh is used as a non-linear function.

3.3.1 *Matching score aggregation components.*

ANSR includes the following matching score aggregation components:

- **DF-MLP**: computes $m(\hat{\mathbf{u}}^q_i, d)$ and $m(\hat{\mathbf{b}}^q_i, d)$ by aggregating the matching scores of $\hat{\mathbf{u}}^q_i$ and $\hat{\mathbf{b}}^q_i$, respectively, in different fields of d;
- **QN-MLP**: computes $m(\hat{\mathbf{u}}^q, d)$ and $m(\hat{\mathbf{b}}^q, d)$ by aggregating $m(\hat{\mathbf{u}}^q_i, d)$ and $m(\hat{\mathbf{b}}^q_i, d)$ for all i, respectively;
- **QD-MLP**: computes $s(q, d)$ by aggregating $m(\hat{\mathbf{u}}^q, d)$ and $m(\hat{\mathbf{b}}^q, d)$.

Since after pooling each query is represented with $\hat{n}^q$ embedding vectors, the network has $\hat{n}^q$ DF-MLP components. Given the ith rows of each $\hat{\mathbf{M}}^l$ as input, the ith DF-MLP component computes $m(\hat{\mathbf{u}}^q_i, d)$ by taking into account the importance weights of document fields determined by the *document field attention component*. The outputs of DF-MLP components are used as inputs to the QN-MLP components to compute $m(\hat{\mathbf{u}}^q, d)$ and $m(\hat{\mathbf{b}}^q, d)$ by taking into account the importance weights of embedding vectors in query representation determined by the *query phrase attention component*. The two outputs of the QN-MLP components are used as input to QD-MLP to compute $s(q, d)$.

3.3.2 *Attention components.*

ANSR includes the following attention components:

- **DF-ATT**: computes the importance weights of document fields for aggregating the matching scores of $\hat{\mathbf{u}}^q_i$ and $\hat{\mathbf{b}}^q_i$ in different document fields;
- **QP-ATT**: computes the importance weights of embedding vectors $\hat{\mathbf{r}}^q_i$ in $\hat{\mathbf{r}}^q$ for computing the matching score of a document with respect to the unigram- or bigram-based compressed distributed query representation.

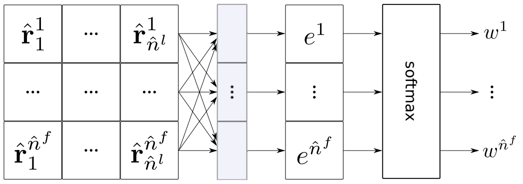

$\hat{\mathbf{r}}^1_1$
...
$\hat{\mathbf{r}}^1_{\hat{n}^l}$
...
...
...
$\hat{\mathbf{r}}^{\hat{n}^f}_1$
...
$\hat{\mathbf{r}}^{\hat{n}^f}_{\hat{n}^l}$
e^1
$e^{\hat{n}^f}$
softmax
w^1
$w^{\hat{n}^f}$

Figure 2: Document field attention component of ANSR.

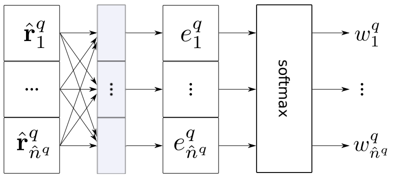


Figure 3: Query phrase attention component of ANSR.

The DF-ATT attention component shown in Figure 2 computes w^l, the importance weight of the lth document field for computing the aggregate matching score of each $\hat{\mathbf{r}}^q_i$ in the entire document as:

$$w^l = \frac{\exp(e^l)}{\sum_{k=1}^{\hat{n}^f} \exp(e^k)} \tag{4}$$

where

$$e^l = f_{df-att}(\hat{\mathbf{r}}^l, \{\hat{\mathbf{r}}^1, \dots, \hat{\mathbf{r}}^{\hat{n}^f}\}) \tag{5}$$

and $f_{df-att}(\cdot)$ represents the layers that compute e^l, intermediate weight of the lth field, based on $\hat{\mathbf{r}}^l$, the compressed representation of that field.

The input to the ith DF-MLP component consists of $\{\hat{\mathbf{M}}^1_{i,:}, \dots, \hat{\mathbf{M}}^{\hat{n}^f}_{i,:}\}$, which correspond to the similarity vectors between $\hat{\mathbf{r}}^q_i$ and each $\hat{\mathbf{r}}^l_j$ in $\hat{\mathbf{r}}^l$, weighed by w^l computed by the DF-ATT attention module.

Similarly, the QP-ATT attention component shown in Figure 3 computes w^q_i, the importance weight of $\hat{\mathbf{r}}^q_i$, the ith embedding vector in unigram- or bigram-based compressed distributed query representation $\hat{\mathbf{r}}^q$ for computing the aggregated matching score of $\hat{\mathbf{r}}^q$ in the entire document as:

$$w^q_i = \frac{\exp(e^q_i)}{\sum_{k=1}^{\hat{n}^q} \exp(e^q_k)} , \tag{6}$$

where

$$e^q_i = f_{qp-att}(\hat{\mathbf{r}}^q_i, \{\hat{\mathbf{r}}^q_1, \dots, \hat{\mathbf{r}}^q_{\hat{n}^q}\}) . \tag{7}$$

and $f_{qp-att}(\cdot)$ represents the layers that compute e^q_i, intermediate weight of the ith embedding vector, based on all embedding vectors in $\hat{\mathbf{r}}^q$.

3.3.3 Regulating the impact of pooling and zero-padding. Zero-padding and pooling operations described in Section 3.2 to generate compressed similarity matrices need to be regulated to avoid the problem of favoring longer fields since: (1) a large field is twice more likely to have k relevant n-grams than equivalently relevant short field that is half its size, which biases the pooling operation to favor longer fields and (2) zero-padding adds $\hat{n}^l - n^l$ zeros to the interaction matrix which makes it equivalent to a field of size $\hat{n}^l$ with $\hat{n}^l - n^l$ n-grams that are not relevant to a query. To compensate for the above side-effects of pooling and zero-padding for documents with the fields of different lengths, we use field length to regulate the field importance weight, so that $s(q,d)$ becomes less dependent on the lengths of document fields. To do so and to compensate for the effect of zero-padding in documents with the number of fields smaller than $\hat{n}^f$, we update the weights in the document field attention component:

$$w^l \leftarrow w^l \times \frac{\hat{n}^l}{n^l} \times \frac{\hat{n}^f}{n^f} . \tag{8}$$

3.4 Model training

Besides the weights in DF-ATT and QP-ATT attention components ($\mathbf{W}^{(DF)}_{ATT}$, $\mathbf{W}^{(QP)}_{ATT}$), three sets of weights ($\mathbf{W}^{(DF)}$, $\mathbf{W}^{(QN)}$, and $\mathbf{W}^{(QD)}$) should be estimated for the DF-MLP, QN-MLP, and QD-MLP matching score aggregation components. Therefore, the set of trainable parameters in ANSR is:

$$\mathcal{W} = \{\mathbf{W}^{(DF)}_{ATT}, \mathbf{W}^{(QP)}_{ATT}, \mathbf{W}^{(DF)}, \mathbf{W}^{(QN)}, \mathbf{W}^{(QD)}\} . \tag{9}$$

To reduce the number of trainable parameters in the network, all DF-MLP and QN-MLP components share the same weights. Therefore, without considering bias, each DF-MLP component in ANSR has $\hat{n}^f \hat{n}^l$ neurons in its input layer, $\hat{n}^f$ neurons in its hidden layer and one neuron in its output layer, which results in $\mathbf{W}^{(DF)}$ consisting of $\hat{n}^f(\hat{n}^f \hat{n}^l + 1)$ weights. Similarly, we assume that QN-MLP has $\hat{n}^q$ nodes in its input and hidden layer and one node in its output layer, which results in $\mathbf{W}^{(QN)}$ having $\hat{n}^q(\hat{n}^q + 1)$ weights. Finally, QD-MLP has two neurons in its input and hidden layers and one neuron in its output layer, therefore $\mathbf{W}^{(QD)}$ has 6 weights. In total, the weight matrices in score aggregation components have $\hat{n}^f(\hat{n}^f \hat{n}^l + 1) + \hat{n}^q(\hat{n}^q + 1) + 6$ trainable parameters. Therefore, since we have $\hat{n}^q$ DF-MLP and two QN-MLP, by applying the weight sharing strategy to components of the same type, we need to estimate $\hat{n}^f(\hat{n}^f \hat{n}^l + 1)(\hat{n}^q - 1) + \hat{n}^q(\hat{n}^q + 1)$ fewer weights.

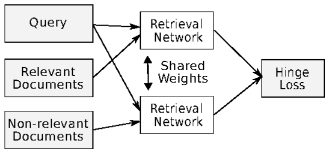


Figure 4: Given a triplet of relevant and non-relevant documents with respect to a query, we obtain their relevance scores and optimize a hinge loss function to train the network. The retrieval networks share weights and have the architecture shown in Figure 1.

As can be seen from Figure 4, we optimize a loss function that depends on the triplets of relevant and non-relevant documents given a query, i.e., $<q, d^{\mathrm{n}}, d^{\mathrm{r}}>$, where d^{n} and d^{r} are the documents in the training data that are judged as non-relevant and relevant, respectively.

Therefore, the set of triplets in our training data is as follows:

$$\mathcal{T} = \left\{ <q, d^{\mathrm{n}}_i, d^{\mathrm{r}}_i> \mid (q, d^{\mathrm{n}}_i) \in \mathcal{S}^{\mathrm{n}}; (q, d^{\mathrm{r}}_i) \in \mathcal{S}^{\mathrm{r}}, i = 1, \dots, T \right\} , \tag{10}$$

Table 2: Summary of collections, query sets and relevance judgments used for experimental evaluation. The last column provides the names of document fields in each collection.

Name	# Docs	Queries	# Rel. Judg.	Fields
GOV2	25.2M	701–850	135K	full, inlink, metakd, doctitle, alt, largefont
DBpedia-v2	3.5M	1–467	49K	names, rel. categories, similar entity names, entity name, attributes
HomeDepot	55K	1–1000	3K	title, description, attribute

where T is number of triplets in our training data, and $\mathcal{S}^{\mathrm{n}}$ and $\mathcal{S}^{\mathrm{r}}$ are the sets of non-relevant and relevant query-document pairs.

We define the hinge loss function for each triplet $<q, d^{\mathrm{n}}, d^{\mathrm{r}}>$ in training data as:

$$l_q(d^{\mathrm{n}}, d^{\mathrm{r}}) = \max(0, \zeta - s(q, d^{\mathrm{r}}) + s(q, d^{\mathrm{n}})), \quad <q, d^{\mathrm{n}}, d^{\mathrm{r}}> \in \mathcal{T} \tag{11}$$

where ζ is a margin between the relevance scores of relevant and non-relevant documents with respect to a query. We obtain $\zeta = 1$ by using the validation set. We compute the loss function given all possible triplets in the training data as:

$$L_q(\mathcal{W}) = \sum_{<q, d^{\mathrm{n}}, d^{\mathrm{r}}> \in \mathcal{T}} l_q(d^{\mathrm{n}}, d^{\mathrm{r}}) \ . \tag{12}$$

By considering an additional regularizer term in the above objective function, we define the optimization problem in our training process as:

$$\min_{\mathcal{W}} \left(\sum_{<q, d^{\mathrm{n}}, d^{\mathrm{r}}> \in \mathcal{T}} \max(0, \zeta - s(q, d^{\mathrm{r}}) + s(q, d^{\mathrm{n}})) + \frac{\gamma}{2} ||\mathcal{W}||_2^2 \right), \tag{13}$$

where γ is a constant that corresponds to weight decay. We set $\gamma = 0.0005$ in our experiments.

4 EXPERIMENTS

We evaluated ANSR in the context of three different structured document retrieval scenarios: Web search, product search, and entity retrieval from a knowledge graph. For experiments, we used three publicly available benchmarks of different sizes: GOV2, DBpedia-v2, and HomeDepot. We compare the performance of ANSR and its variations with the state-of-the-art probabilistic models for structured document retrieval and neural architectures for structured and unstructured document retrieval.

4.1 Baselines

4.1.1 Neural and probabilistic baselines. The goal of this experimental evaluation is to investigate the performance of neural architectures for structured document retrieval when human-labeled relevance judgments are used instead of user behavior data (e.g. click-throughs) for training. Therefore, with the exception of NRM-F [44], we focus on the baselines that were designed for the scenarios in which the user behavior data is not available. Specifically, we compare ANSR with PRMS [14], MLM [24], BM25F [28] and FSDM [45]. The parameters of these baselines are trained by using the coordinate ascent [20]. We also consider DRMM [9], DESM (IN-OUT, trained on queries) [22] and NRM-F* [44], which is NRM-F [44] trained without using click-through data. Although DESM [22] uses query logs to obtain word embeddings, it does not require click data. Both DESM [22] and NRM-F* [44] were evaluated under telescoping [18] settings. It is notable that NRM-F assumes that all documents have the same number of fields, which does not allow for a pre-trained network to be adopted in the case when structured documents have different number of fields.

4.1.2 Variations of ANSR. We obtain the first variation of ANSR, called **ANSR-na**, by removing the attention network from the architecture of ANSR, i.e., by assigning the same weights to all fields and to all embedding vectors in the compressed query representation regardless of their relative importance to the retrieval task. In **ANSR-no-pooling**, instead of using our pooling strategy, we select the first $\hat{n}^l$ terms from each document field and find their matching scores with respect to a query. Inspired by the systematic comparisons in [5], we also evaluate the effectiveness of our proposed architecture relative to its *count-based* version, called **ANSR-count**. In **ANSR-count**, we utilize histogram-based (LCH) approach proposed in [9] and modify the process of creating fixed-size similarity matrices $\hat{\mathbf{M}}^l$. To do so, $\hat{\mathbf{M}}^l$, which is now a matching histogram matrix with the size $\hat{n}^q \times \hat{n}^l$, is obtained by first grouping the matching signals $\{\mathbf{M}^l_{i,j} \mid j \in \{1, \ldots, n^l\}\}$ in each row of $\mathbf{M}^l$ into a fixed number of strength levels and then applying logarithm to the counts. For example, if $\hat{n}^l = 3$ and we consider $[0.8, -0.1, -0.4, 0.75]$ as the i-th row of $\mathbf{M}^l$, the i-th row of $\hat{\mathbf{M}}^l$ becomes $[\log(1 + 1), \log(1 + 1), \log(1 + 2)]$ since we have 1, 1 and 2 matching signals with the strength levels between $(-1, -0.33)$, $(-0.33, 0.33)$ and $(0.33, 1)$, respectively, and 1 is added to all count values to avoid undefined values. For ANSR-count, we consider $\hat{n}^l = 30$ as suggested in [9]. Finally, we also consider **ANSR-unigrams**, in which the final relevance score only depends on the aggregated matching scores of embedding vectors in unigram-based compressed query representation.

4.2 Experimental setup

We used TensorFlow r1.2[2] to implement the proposed attentive neural architecture for structured document retrieval. To train ANSR, we ran Adam [15] for 150 epochs with a learning rate of 0.001 and a batch size of 100. For a fair comparison, we use the same pre-trained word embeddings obtained for Dual Embedding Space Model (DESM) model[3] by Nalisnick et al. [22] in DRMM and ANSR. To avoid over-fitting, we use drop-out with a rate of 0.2. We applied ANSR to re-rank the top 2000 documents, which were initially retrieved using BM25F [28]. We trained the parameters of the baselines on the same collections that we trained ANSR. We used ten-fold cross-validation to tune the hyper-parameters of ANSR, such as $\hat{n}^l$ and $\hat{n}^q$, as well as hyper-parameters of the baselines, separately for each collection. We used MAP, P@10, NDCG@10, and b-pref as evaluation metrics.

[2] https://www.tensorflow.org/versions/r1.2/

[3] https://www.microsoft.com/en-us/download/details.aspx?id=52597

Table 3: Performance of ANSR and the baselines on GOV2, HomeDepot, and DBpedia collections. Statistically significant improvements in terms of MAP of ANSR over FSDM and DRMM measured by the Fisher's randomization test with $\alpha = 0.05$ are indicated by "⋆" and "†". Percentage improvements over FSDM and DRMM are shown in parentheses.

	GOV2				
		MAP	**P@10**	**NDCG@10**	**b-pref**
LM-based Methods	**PRMS [14]**	0.1964	0.4058	0.3448	0.2489
	MLM [24]	0.2908	0.5648	0.4729	0.3539
	BM25F [28]	0.2954	0.5478	0.4556	0.3493
	FSDM [45]	0.3012	0.5817	0.4789	0.3572
Neural Methods	**DESM [22]**	0.2968 (-1.46%)	0.5714 (-1.77%)	0.4575 (-4.47%)	0.3505 (-1.88%)
	DRMM [9]	0.3113 (3.35%)	0.5880 (1.08%)	0.4722 (-1.40%)	0.3591 (0.53%) g
	NRM-F* [44]	0.1491 (-50.50%)	0.2903 (-50.09%)	0.2132 (-55.48%)	0.1792 (-49.83%)
Proposed Method Variations	**ANSR-na**	0.3074 (2.06%/-1.25%)	0.5755 (-1.07%/-2.13%)	0.4794 (0.10%/1.52%)	0.3682 (3.08%/2.53%)
	ANSR-static	0.2934 (-2.59%/-5.75%)	0.5691 (-2.17%/-3.21%)	0.4647 (-2.97%/-1.59%)	0.3556 (-0.45%/-0.97%)
	ANSR-count	$0.3183^{\star}$ (5.68%/2.25%)	0.5952 (2.32%/1.22%)	0.4894 (2.19%/3.64%)	0.362 (1.34%/0.81%)
	ANSR-unigrams	$0.3166^{\star}$ (5.11%/1.70%)	0.5949 (2.27%/1.17%)	0.4907 (2.46%/3.92%)	0.3711 (3.89%/3.34%)
	ANSR	$0.3246^{\star\dagger}$ (7.77%/4.27%)	0.6023 (3.54%/2.43%)	$0.4937^{\dagger}$ (3.09%/4.55%)	$0.3756^{\star\dagger}$ (5.15%/4.59%)
	Home-depot				
		MAP	**P@10**	**NDCG@10**	**b-pref**
LM-based Methods	**PRMS [14]**	0.2287	0.108	0.2641	0.877
	MLM [24]	0.2476	0.1183	0.2893	0.9161
	BM25F [28]	0.2537	0.1201	0.2952	0.9231
	FSDM [45]	0.2591	0.1206	0.3024	0.9206
Neural Methods	**DESM [22]**	0.2349 (-9.34%)	0.1107 (-8.21%)	0.2769 (-8.43%)	0.8943 (-2.86%)
	DRMM [9]	0.2484 (-4.13%)	0.1131 (-6.22%)	0.2952 (-2.38%)	0.9034 (-1.87%)
	NRM-F* [44]	0.1536 (-40.72%)	0.0723 (-40.05%)	0.1832 (-39.42%)	0.4272 (-53.60%)
Proposed Method Variations	**ANSR-na**	0.2511 (-3.09%/1.09%)	0.1154 (-4.31%/2.03%)	0.2946 (-2.58%/-0.20%)	0.8935 (-2.94%/-1.10%)
	ANSR-static	0.2489 (-3.94%/0.20%)	0.1072 (-11.11%/-5.22%)	0.2858 (-5.49%/-3.18%)	0.8935 (-2.94%/-1.10%)
	ANSR-count	$0.2821^{\star\dagger}$ (8.88%/13.57%)	$0.1247^{\dagger}$ (3.40%/10.26%)	$0.3174^{\star\dagger}$ (4.96%/7.52%)	0.9352 (1.59%/3.52%)
	ANSR-unigrams	$0.2726^{\star\dagger}$ (5.21%/9.74%)	$0.1278^{\star\dagger}$ (5.97%/13.00%)	$0.3168^{\star\dagger}$ (4.76%/7.32%)	0.9342 (1.48%/3.41%)
	ANSR	$0.2846^{\star\dagger}$ (9.84%/14.57%)	$0.1377^{\star\dagger}$ (14.18%/21.75%)	$0.3204^{\star\dagger}$ (5.95%/8.54%)	$0.9585^{\dagger}$ (4.12%/6.10%)
	DbPedia-v2				
		MAP	**P@10**	**NDCG@10**	**b-pref**
LM-based Methods	**PRMS [14]**	0.2934	0.3594	0.4126	0.3142
	MLM [24]	0.3467	0.3887	0.4365	0.3547
	BM25F [28]	0.3799	0.4077	0.4605	0.3902
	FSDM [45]	0.3679	0.4073	0.4524	0.3748
Neural Methods	**DESM [22]**	0.3523 (-7.27%)	0.3894 (-4.49%)	0.4527 (-1.69%)	0.3621 (-7.20%)
	DRMM [9]	0.3682 (-3.08%)	0.4012 (-1.59%)	0.4515 (-1.95%)	0.3895 (-0.18%)
	NRM-F* [44]	0.1878 (-50.57%)	0.2092 (-48.69%)	0.2402 (-47.84%)	0.1802 (-53.82%)
Proposed Method Variations	**ANSR-na**	0.3824 (0.66%/3.86%)	0.4057 (-0.49%/1.12%)	0.4447 (-3.43%/-1.51%)	0.3792 (-2.82%/-2.64%)
	ANSR-static	0.3765 (-0.89%/2.25%)	0.3964 (-2.77%/-1.20%)	0.4339 (-5.78%/-3.90%)	0.3635 (-6.84%/-6.68%)
	ANSR-count	$0.3921^{\dagger}$ (3.21%/6.49%)	$0.4194^{\dagger}$ (2.87%/4.54%)	0.4632 (0.59%/2.59%)	0.3932 (0.77%/0.95%)
	ANSR-unigrams	$0.3912^{\dagger}$ (2.97%/6.25%)	0.4134 (1.40%/3.04%)	$0.4703^{\dagger}$ (2.13%/4.16%)	$0.4103^{\dagger}$ (3.31%/3.49%)
	ANSR	$0.3992^{\star\dagger}$ (5.08%/8.42%)	$0.4256^{\dagger}$ (4.39%/6.08%)	$0.4812^{\dagger}$ (4.50%/6.58%)	$0.4118^{\star\dagger}$ (5.54%/5.73%)

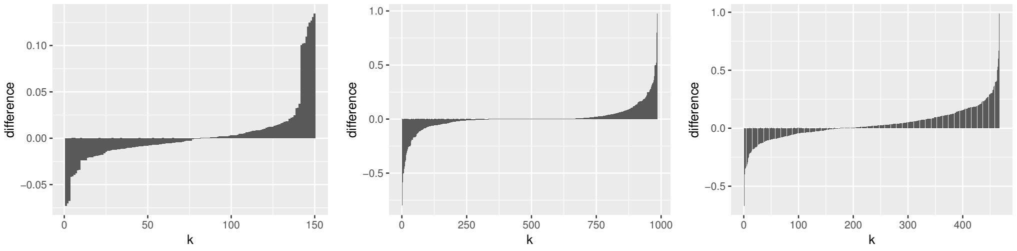
4.3 Collections

We evaluated the retrieval accuracy of ANSR on three different tasks. We adopted GOV2[4] as a Web search collection, HomeDepot[5] as a product search collection, and DBpedia-v2[6] as an entity retrieval collection. These collections are summarized in Table 2.

[4] http://ir.dcs.gla.ac.uk/test_collections/

[5] https://www.kaggle.com/c/home-depot-product-search-relevance/data

[6] https://github.com/iai-group/DBpedia-Entity

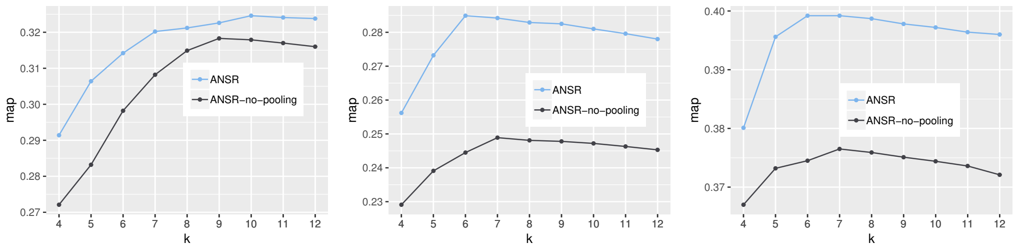


(a) GOV2

(b) HomeDepot

(c) DBpedia-v2

Figure 5: Topic-level difference in retrieval accuracy between ANSR and FSDM on three different collections.



(a) GOV2

(b) HomeDepot

(c) DBpedia-v2

Figure 6: The effect of the pooling size (k) on the performance of ANSR and ANSR-no-pooling.

Table 4: Performance of ANSR and best-performing baselines (in terms of MAP) on difficult queries in GOV2, Home-Depot and DBpedia-v2. Statistically significant improvements of ANSR measured by the Fisher's randomization test with $\alpha = 0.05$ over these baselines are indicated by by "$\star$" and "$\dagger$". Percentage improvements are shown in parentheses.

GOV2				
	MAP	**P@10**	**NDCG@10**	**b-pref**
FSDM	0.0381	0.0533	0.052	0.0293
DRMM	0.03	0.0466	0.0552	0.0244
ANSR	0.0448*†	0.0626*†	0.0782*†	0.0429*†
	(17%/49%)	(17%/34%)	(50%/41%)	(46%/75%)
HomeDepot				
	MAP	**P@10**	**NDCG@10**	**b-pref**
FSDM	0.0344	0.015	0.0372	0.7801
DRMM	0.0289	0.0131	0.0384	0.6923
ANSR	0.0402*†	0.0194*†	0.0575*†	0.8031†
	(16%/39%)	(29%/48%)	(54%/49%)	(2%/16%)
DBpedia-v2				
	MAP	**P@10**	**NDCG@10**	**b-pref**
BM25F	0.1032	0.1583	0.1832	0.115
DRMM	0.0935	0.252	0.0923	0.2498
ANSR	0.1631 *†	0.2609*†	0.2196*†	0.2642*
	(58%/74%)	(64%/3%)	(19%/137%)	(129%/5%)

The publicly available benchmark for entity retrieval from knowledge graph, described in detail in [10], is composed of 113 queries aiming at specific entities, 99 keyword queries, 115 queries aiming at the list of entities and 140 natural language questions. In this dataset, 7K documents are judged as highly relevant, 10K as relevant, and 33K as non-relevant. We use DBpedia 2015−10 as the knowledge graph [10] for the entity retrieval task. There are 135K available relevance judgments, 19.89% of which are relevant or somewhat relevant in GOV2. Finally, the HomeDepot collection is more balanced with 1.4K relevant, 3K somewhat relevant, and 1.5K non-relevant judgments. We convert the relevance judgments in all training datasets into binary (relevant or non-relevant) ones.

We created six-field documents in the case of GOV2 collection, five-field document for each entity in DBpedia-v2 collection, and three-field document for each product in HomeDepot collection. As described in Table 2, in GOV2, the field "full" contains all the content of a document, "doctitle" field contains the title of a document, "largefonts" field contains all the text in a font larger than normal, "metakd" field contains document meta-data , "inlink" field contains all the incoming hyper-links to a document, and "alt" field contains the alternative texts for all images in a document. The contents of the fields in DBpedia-v2 entity description documents are obtained using the method in [45]. Finally, documents in HomeDepot collection have three fields, with the "attribute" field obtained by concatenating all attributes of each product.

4.4 Comparison with baselines

In Table 3, we compare the performance of our method with probabilistic baselines on three ad-hoc structured document retrieval tasks. Depending on the collection and the retrieval task, we observe that FSDM, a state-of-the-art probabilistic structured document retrieval model, may not outperform BM25F. However, ANSR demonstrates statistically significant improvement over all baselines with respect to the majority of evaluation metrics in all retrieval tasks. We mainly attribute the improvement of ANSR over FSDM and BM25F to the fact that ANSR uses neural networks and embedding vectors of unigrams and bigrams in query and collection documents to compute the relevance score, which helps in bridging the lexical gap.

Table 3 indicates that, regardless of the collection and retrieval task, ANSR performs significantly better than other neural baselines and the best performing neural baseline is DRMM. As shown in [44], NRM-F outperforms probabilistic retrieval models, such as BM25, in scenarios that involve abundant click-through data. However, due to the sensitivity of NRM-F to the training data size, we observe in Table 3 that NRM-F* does not attain its superior performance over probabilistic methods when a limited number of relevance judgments is available for ad-hoc retrieval tasks. As follows from Table 3, interaction-based architectures, such as ANSR and DRMM, can be adequately trained even with limited training data, partially due to their smaller size compared to representation-based architectures, such as NRM-F . The improvement of DRMM and ANSR over DESM, shown in Table 3, can be attributed to utilization of the collection-dependent training data by DRMM and ANSR.

Figure 5 demonstrates topic-level differences between ANSR and the best performing structured document retrieval baseline (FSDM) in terms of the average precision for all three retrieval tasks. As follows from this figure, ANSR has higher average precision than FSDM for 45.33% of the queries in GOV2. In this collection, the magnitude of improvements in average precision is 1.66 times greater than the magnitude of reductions. This result highlights superior ability of ANSR to deal with long field documents, due to utilization of compressed representations and explicit correction of the pooling bias. Similar observations can be made with respect to the topic-level differences in average precision between ANSR and FSDM in the case of the other two collections. Specifically, ANSR improves the average precision of 44.00% and 58.88% of queries in the case of HomeDepot and DBpedia-v2 collections, respectively. In these collections, the magnitude of improvements in average precision over all queries is 1.26 and 1.51 higher, than the magnitude of reductions. Since HomeDepot and DBpedia-v2 have significantly longer queries than GOV2, this result highlights the effectiveness of ANSR in the case of verbose and descriptive queries.

4.4.1 *Difficult queries.*

In this work, we consider a query as difficult if the average precision of MLM on this query is less than 0.05. According to this definition, 7.38% of queries in GOV2, 7.7% of queries in HomeDepot, and 29.3% of queries in DBpedia-v2 are difficult. Table 4 illustrates the performance of ANSR in comparison with the best performing baselines in Table 3 (according to MAP) only on difficult queries. This table indicates the superior performance of ANSR over the baselines on all the collections (particularly on DBpedia-v2), which is as a result of considering unigram- and bigram- based query phrases and using the attention mechanism to focus on the most important query parts.

4.4.2 *Best and worst performing queries.*

Due to space constraints, we highlight the best and worst performing queries for ANSR in comparison to FSDM according to the average precision only in the case of HomeDepot collection. The best performing query for ANSR is *"single lever hole bathroom sink faucet"*, which has only one relevant document with the title *"Belle Foret Single Hole 1-Handle High Arc Bathroom Vessel Faucet in Chrome with Metal Lever Handles"* in relevance judgments. This document has longer fields than the average field length in this collection, which can be an indication that the superior performance of ANSR over FSDM is due to its ability to better handle documents that have longer fields.

The worst performing query for ANSR is *"popular"*, which also has only one relevant document with the title *"Bloomsz Most Popular Water Plant Collection (8-Pack)"* in relevance judgments. ANSR was not able to place this document among the top-ranked ones. Instead, it ranked the document with the title *"South Shore Furniture Popular Twin Mates Bed in Mocha"* as the top-ranked document, since it has more words that are semantically similar to the query term *"popular"*. This can be a consequence of using word embeddings by ANSR, which can cause topic drift for very short queries [27].

4.5 Ablation study

In Table 3, we also provided experimental results for ANSR-na, which does not use the attention mechanism (or equivalently the attention module provides the same weights for all document fields and embedding vectors in query representation). This table indicates superior performance of ANSR over ANSR-na, from which we can conclude that the proposed attention mechanism plays a critical role in computing the relevance score of a structured document with respect to a query.

Table 3 shows that ANSR has a slightly better retrieval performance than its count-based variant (ANSR-count). In other words, it shows that combining the proposed pooling strategy with the proposed attention mechanism that weighs the rows in the compressed similarity matrices according to field importance results in higher retrieval accuracy than using the histogram-based method.

Figure 6 illustrates the effect of the size of the pooling window ($k = \lceil n^l / \hat{n}^l \rceil$) on performance of ANSR and ANSR-no-pooling. Based on Figure 6 and Table 3, we can conclude that ANSR has substantially better retrieval accuracy in terms of MAP than ANSR-no-pooling. This can be attributed to the fact that, unlike ANSR, ANSR-no-pooling selects only the first terms in each document field. We can also observe from Figure 6 that the optimal value of k depends on the collection and the retrieval task. Specifically, ANSR in conjunction with our proposed pooling strategy has the best performance on GOV2, DBpedia-v2 and HomeDepot collections when $k = 10$, $k = 6$ and $k = 6$, respectively.

A distinctive feature of ANSR is that it takes into account the matching signals between both unigram- as well as bigram- based distributed representations of a query and document fields. In Table 3, we quantify the contribution of this feature to the overall performance of ANSR. Specifically, we evaluate the retrieval accuracy of ANSR-unigrams, which only considers that matching signals between unigram-based distributed representations of a query and document fields. Comparison of performance of ANSR with ANSR-unigrams in Table 3 indicates that going beyond the matching signals between only unigram-based query representation and accounting for sequential dependencies between the terms in a query and document fields results in improved retrieval accuracy.

5 CONCLUSIONS

In this paper, we propose an attentive neural architecture for ad-hoc structured document retrieval, which can be effectively applied to different tasks, such as Web search, product search and entity retrieval from a knowledge graph. The proposed architecture has modular structure and includes the components to compute the relevance score of a structured document given a keyword query from the similarity matrices between *n*-gram based compressed distributed representations of a query and document fields. The proposed architecture also leverages a pooling strategy to generate fixed-size local interactions between *n*-gram based distributed representations of a query and document fields and employs an attention mechanism to focus on the most important document fields and query phrases . Extensive experimental evaluation of the proposed architecture and its variations on different retrieval tasks indicates that the pooling, relevance matching and attention strategies in the proposed neural architecture result in significant improvements of retrieval accuracy over state-of-the-art probabilistic models for ad-hoc structured document retrieval and neural architectures for document retrieval.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

[2] Saeid Balaneshin-kordan and Alexander Kotov. 2016. A study of document expansion using translation models and dimensionality reduction methods. In *Proceedings of ACM ICTIR*. 233–236.

[3] Saeid Balaneshin-kordan and Alexander Kotov. 2017. Embedding-based Query Expansion for Weighted Sequential Dependence Retrieval Model. In *Proceedings of ACM SIGIR*. 1213–1216.

[4] Saeid Balaneshin-kordan and Alexander Kotov. 2018. Deep Neural Architecture for Multi-Modal Retrieval based on Joint Embedding Space for Text and Images. In *Proceedings of ACM WSDM*. 28–36.

[5] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors.. In *Proceedings of ACL*. 238–247.

[6] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *Proceedings of WWW*. 531–541.

[7] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *Proceedings of ACM WSDM*. 126–134.

[8] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. *arXiv preprint arXiv:1704.08803* (2017).

[9] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *Proceedings of ACM CIKM*. 55–64.

[10] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. DBpedia-Entity v2: A Test Collection for Entity Search. In *Proceedings of ACM SIGIR*. 1265–1268.

[11] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of NIPS*. 2042–2050.

[12] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of ACM CIKM*. 2333–2338.

[13] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of ACL*. 655–665.

[14] Jinyoung Kim, Xiaobing Xue, and W Bruce Croft. 2009. A probabilistic retrieval model for semistructured data. In *Proceedings of ECIR*. 228–239.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Alexander Kotov, Vineeth Rakesh, Eugene Agichtein, and Chandan K Reddy. 2015. Geographical latent variable models for microblog retrieval. In *Proceedings of ECIR*. 635–647.

[17] Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. In *Proceedings of NIPS*. 1367–1375.

[18] Irina Matveeva, Chris Burges, Timo Burkard, Andy Laucius, and Leon Wong. 2006. High accuracy retrieval with multiple nested ranker. In *Proceedings of ACM SIGIR*. 437–444.

[19] Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of ACM SIGIR*. 472–479.

[20] Donald Metzler and W Bruce Croft. 2007. Linear feature-based models for information retrieval. *Information Retrieval* (2007), 257–274.

[21] Bhaskar Mitra and Nick Craswell. 2017. Neural Models for Information Retrieval. *arXiv preprint arXiv:1705.01509* (2017).

[22] Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving Document Ranking with Dual Word Embeddings. In *Proceedings of WWW*. 83–84.

[23] Fedor Nikolaev, Alexander Kotov, and Nikita Zhiltsov. 2016. Parameterized fielded term dependence models for ad-hoc entity retrieval from knowledge graph. In *Proceedings of ACM SIGIR*. 435–444.

[24] Paul Ogilvie and Jamie Callan. 2003. Combining document representations for known-item search. In *Proceedings of ACM SIGIR*. 143–150.

[25] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of AAAI*. 12–17.

[26] Benjamin Piwowarski and Patrick Gallinari. 2003. A machine learning model for information retrieval with structured documents. *Machine Learning and Data Mining in Pattern Recognition* (2003), 425–438.

[27] Navid Rekabsaz, Mihai Lupu, Allan Hanbury, and Hamed Zamani. 2017. Word Embedding Causes Topic Shifting; Exploit Global Context!. In *Proceedings of ACM SIGIR*. 1105–1108.

[28] Stephen Robertson, Hugo Zaragoza, and Michael Taylor. 2004. Simple BM25 extension to multiple weighted fields. In *Proceedings of ACM CIKM*. 42–49.

[29] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of EMNLP*. 379–389.

[30] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of ACM SIGIR*. 373–382.

[31] Daniel Sheldon, Milad Shokouhi, Martin Szummer, and Nick Craswell. 2011. LambdaMerge: merging the results of query reformulations. In *Proceedings of ACM WSDM*. 795–804.

[32] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of ACM CIKM*. 101–110.

[33] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Proceedings of NIPS*. 2377–2385.

[34] Krysta M Svore and Christopher JC Burges. 2009. A machine learning approach for improved BM25 retrieval. In *Proceedings of ACM CIKM*. 1811–1814.

[35] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of ACL*. 1556–1566.

[36] Andrew Trotman. 2004. Optimal Structure Weighted Retrieval. In *Proceedings of ADCS*.

[37] Andrew Trotman. 2004. Searching structured documents. *Information Processing & Management* 40, 4 (2004), 619–632.

[38] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of ACM SIGIR*. 55–64.

[39] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of ICML*. 2048–2057.

[40] Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. 2016. aNMM: Ranking short answer texts with attention-based neural matching model. In *Proceedings of ACM CIKM*. 287–296.

[41] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. 2016. Hierarchical Attention Networks for Document Classification.. In *HLT-NAACL*. 1480–1489.

[42] Wenpeng Yin and Hinrich Schütze. 2015. MultiGranCNN: An Architecture for General Matching of Text Chunks on Multiple Levels of Granularity.. In *Proceedings of ACL*. 63–73.

[43] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of IEEE CVPR*. 4584–4593.

[44] Hamed Zamani, Bhaskar Mitra, Xia Song, Nick Craswell, and Saurabh Tiwary. 2018. Neural Ranking Models with Multiple Document Fields. In *Proceedings of ACM WSDM*. 700–708.

[45] Nikita Zhiltsov, Alexander Kotov, and Fedor Nikolaev. 2015. Fielded sequential dependence model for Ad-Hoc entity retrieval in the web of data. In *Proceedings of ACM SIGIR*. 253–262.