PACRR Gated Expansion for TREC CAR 2018

Sean MacAvaney[1], Andrew Yates[2], Nazli Goharian[1], and Ophir Frieder[1]

[1]Information Retrieval Lab, Georgetown University,
{firstname}@ir.cs.georgetown.edu
[2]Max Planck Institute for Informatics, ayates@mpi-inf.mpg.de

Abstract

In this work, we present our approach to the 2018 TREC Complex Answer Retrieval (CAR) task. We submitted two passage retrieval runs. The first uses the state-of-the-art technique from TREC CAR 2017: a modified neural ranker modified to incorporate query heading frequency information while performing term matching on each heading independently. The second run incorporates a novel gated technique for incorporating query expansion terms in a neural ranker. Our TREC runs indicate significant performance improvements can be achieved when using the expansion approach.

1 Introduction

Complex Answer Retrieval (CAR) is the task of finding answers to questions that have complex, multi-paragraph answers. The TREC CAR 2018 track is a continuation of the TREC CAR 2017 track, which first introduced CAR [1]. TREC CAR 2018 uses a similar task formulation, with the addition of questions from AI2's Text book question answering dataset (TQA) for task evaluation.

In this work, we describe our passage retrieval submissions to TREC CAR 2018. Our main contribution is the addition of a query expansion technique, which we find significantly improves CAR performance from a baseline neural method.

2 Background and Related Work

TREC CAR 2017 first introduced the complex answer retrieval task. It makes use of a dump of Wikipedia as both a source of answers (paragraphs), complex queries (article heading hierarchies), and an automatic source of relevance judgments (paragraphs under a given heading are assumed relevant to the query associated with that heading). Various baseline approaches were proposed on this dataset. Early baseline approaches for CAR passage retrieval include BM25, cosine similarity (both with TF-IDF vectors and word embeddings), a baseline learning-to-rank approach, query expansion, and the Duet deep neural model [10, 11]. Submissions to TREC CAR 2017 included various approaches, including neural ranking architectures [7, 8, 12], and query expansion approaches using established unsupervised rankers [5].

The leading approaches for CAR incorporate two concepts into existing neural ranking methods: heading frequency and heading independence [7, 6]. Heading frequencies are incorporated by calculating the frequency that CAR headings appear in the training set, and using this as a signal when learning to rank. The intuition is that headings that occur frequently in training data (e.g., "History") are less likely to match in the text than headings that occur less frequently (e.g., "Green Sea Turtle"). Heading independence is achieved by limiting the scope of soft n-gram matching in in target documents by only matching n-grams within a given heading. This approach is similar to modifications to the Sequential Dependence Model (SDM [9]), and also performed well at TREC CAR 2017 [5].

3 Methods

3.1 Baseline CAR-PACRR

In this section we describe the details of our first run (`guir`), the CAR-PACRR model. This was an implementation of the prior state-of-the-art CAR approach [6], and was based on the PACRR information retrieval architecture [3]. Let the general-domain neural ranking model $R(\mathbf{q},\mathbf{d}) = C(M(\mathbf{q},\mathbf{d}))$. $M(\mathbf{q},\mathbf{d})$ represents a matching function, yielding a relevance scores for each query term in query $\mathbf{q} = \{\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_{|\mathbf{q}|}\}$ from document $\mathbf{d} = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_{|\mathbf{d}|}\}$. In PACRR, this function represents convolutions over the query-document similarity matrix. $C(\cdot)$ represents a combination function, which aggregates individual query term scores into a final relevance score. In PACRR, this is accomplished using a simple dense feedforward network. One naïve solution for CAR simply concatenates all heading terms in a CAR query, and uses the PACRR as-is.

CAR-PACRR aims to tailor the PACRR architecture for CAR in two ways. First, it breaks the headings into three categories: title, intermediate, and target headings, a technique referred to as heading independence. This differs from prior work which simply concatenates all headings, treating them as a single, long query. The title ($\mathbf{q}^{title}$) represents terms in the root of the query tree, and corresponds to the title of an article. The target heading $\mathbf{q}^{target}$ represents the bottom-most heading in a path in the query tree, and is the main topic of the query. Intermediate headings $\mathbf{q}^{int}$ are a concatenation of headings along the path between $\mathbf{q}^{title}$ and $\mathbf{q}^{target}$, which provide context for $\mathbf{q}^{target}$. CAR-PACRR performs matching separately on each of these categories, and concatenates the results prior to combination:

$$R(\mathbf{q},\mathbf{d}) = C(M(\mathbf{q}^{title},\mathbf{d}) \| M(\mathbf{q}^{int},\mathbf{d}) \| M(\mathbf{q}^{target},\mathbf{d}))$$

By performing matching separately for each heading position, the model is able to isolate and prioritize the information need of each position separately. For instance, precise matching on $\mathbf{q}^{title}$ may be more important than matching on $\mathbf{q}^{int}$.

The second change that CAR-PACRR makes to the model is incorporating heading frequency statistics. The intuition is that frequent headings (e.g., "History") are less important to match precisely than infrequent headings (e.g., "After the Acts of Union of 1707"). This is similar to the intuition that terms with a high IDF value are more important to match in a query. However, these frequencies are calculated among *query headings* found in the training corpus, not the document collection itself. Heading frequency values are incorporated into the CAR-PACRR model by simply concatenating an additional vector $\mathbf{hf} = \{\mathbf{hf}^{title}, \mathbf{hf}^{int}, \mathbf{hf}^{target}\}$ for score combination, i.e.,

$$R(\mathbf{q},\mathbf{d}) = C(M(\mathbf{q}^{title},\mathbf{d}) \| M(\mathbf{q}^{int},\mathbf{d}) \| M(\mathbf{q}^{target},\mathbf{d}) \| \mathbf{hf})$$

Note that these network modifications are generally applicable to interaction-focused neural architectures that follow the $R(\mathbf{q},\mathbf{d}) = C(M(\mathbf{q},\mathbf{d}))$ paradigm (e.g., K-NRM [14], MatchPyramid [13], and DRMM [2]).

3.2 CAR-PACRR with Expansion

Prior work has shown that expanding queries using pseudo-relevance feedback can be an effective technique for boosting ad-hoc ranking performance [4]. Thus, we test the following approach for incorporating query expansion terms into the CAR-PACRR model to improve CAR performance.

Let $\mathbf{e}^h = \{\mathbf{e}^h_1, \mathbf{e}^h_2, ..., \mathbf{e}^h_{|e|}\}$ be the sequence of expansion terms from some query expansion technique (e.g., RM3) for heading h. We augment the function $M(\cdot)$ to accept the sequence of expansion terms, such that:

$$M'(\mathbf{q}^h,\mathbf{d},\mathbf{e}^h) = \mathit{match}(\mathbf{q}^h,\mathbf{d}) \| E(\mathbf{e}^h,\mathbf{d})$$

We define a simple gating technique to perform $E(\mathbf{e}^h,\mathbf{d})$. First, an expansion relevance matrix $\mathbf{R} \in \mathbb{R}^{|\mathbf{e}^h|\times 2}$ is defined, such that $\mathbf{R}_{i,0} = \mathit{idf}(\mathbf{e}^h_i)$ and $\mathbf{R}_{i,1} = \mathit{expscore}(\mathbf{e}^h_i)$, where $\mathit{idf}(\cdot)$ and $\mathit{expscore}(\cdot)$ are the inverse document frequency and expansion score of of term $\mathbf{e}^h_i$, respectively. Then, to calculate the gate weights, a 1×2 convolution is applied to $\mathbf{R}$ with relu activation yielding $\mathbf{G} \in \mathbb{R}^{|\mathbf{e}^h|}$. Meanwhile, an expansion-document similarity score $\mathbf{S} \in \mathbb{R}^{|\mathbf{e}^h|}$ is calculated via the maximum cosine similarity between the word embedding of each expansion term and each document term $\mathbf{S}_i = \max_j \mathit{cosine}(\mathit{embed}(\mathbf{e}^h_i), \mathit{embed}(\mathbf{d}_j))$. The final score for each expansion term

Run	Rprec	MAP	nDCG
guir	0.2479	0.2475	0.4644
guir-exp	*0.2849	*0.2918	*0.5098
QL (unofficial)	0.3154	0.3080	0.5253

Table 1: TREC CAR 2018 performance results of our official runs (guir and guir-exp), and an unofficial query likelihood (QL) baseline run. In all three metrics, the guir-exp run outperforms the guir run (marked with *, paired Student's t-test, $p < 0.01$).

is calculated via element-wise multiplication: $\mathbf{S}'_i = \mathbf{S}_i\mathbf{G}_i$. Finally, the maximum k_{exp} scores are selected $E(\mathbf{e}^h, \mathbf{d}) = \max_i^{k_{exp}} \mathbf{S}'_i$. In the end, query-document relevance is defined as:

$$R'(\mathbf{q}, \mathbf{d}, \mathbf{e}) = C(M'(\mathbf{q}^{title}, \mathbf{d}, \mathbf{e}^{title}) \| M'(\mathbf{q}^{int}, \mathbf{d}, \mathbf{q}^{int}) \| M'(\mathbf{q}^{target}, \mathbf{d}, \mathbf{q}^{target}) \| \mathbf{hf})$$

4 Evaluation

We trained two variants of the CAR-PACRR model: one with the expansion technique (`guir-exp`), and one without (`guir`). Training was conducted on fold 0 of the TREC CAR 2018 training corpus using automatic relevance judgments. We used the performance on the TREC CAR 2017 test set using manual relevance judgments as our validation dataset. That is, we tuned hyper-parameter performance this dataset, such as the optimal training epoch. For our `guir-exp` run, we use the top 100 expansion terms from Relevance Model 3 [4], and set $k_{exp} = 10$.

Our results for TREC CAR 2018 are given in Table 1. The results show a significant improvement in Rprec, MAP, and nDCG when using the expansion approach (guir-exp), as compared to the unmodified neural baseline (guir). However, the baseline approach itself significantly underperforms compared to the baseline QL method (which it re-ranks). This may be due to one of several reasons. Perhaps validating on hierarchical relevance judgments while training on tree hierarchical relevance judgments might have resulted in this performance mismatch. Or, perhaps the techniques may be over-tuned for the Wikiepdia setting, and are under-performing on the TQA queries.

5 Conclusion

In this work, we presented a query expansion approach for CAR. When evaluating with manual relevance judgments, the expansion approach significantly outperforms the unmodified baseline. However, the expansion approach was not able to outperform query likelihood. Further work is necessary to diagnose the issue.

References

[1] L. Dietz, M. Verma, F. Radlinski, and N. Craswell. Trec complex answer retrieval overview. In *TREC*, 2017.

[2] J. Guo, Y. Fan, Q. Ai, and W. B. Croft. A deep relevance matching model for ad-hoc retrieval. In *CIKM*, 2016.

[3] K. Hui, A. Yates, K. Berberich, and G. de Melo. A position-aware deep model for relevance matching in information retrieval. In *EMNLP*, 2017.

[4] V. Lavrenko and W. B. Croft. Relevance-based language models. In *SIGIR*, volume 51, pages 260–267, 2001.

[5] X. Lin and W. Lam. CUIS team for TREC 2017 CAR track. In *TREC*, 2017.

[6] S. MacAvaney, A. Yates, A. Cohan, L. Soldaini, K. Hui, N. Goharian, and O. Frieder. Characterizing question facets for complex answer retrieval. In *SIGIR*, 2018.

[7] S. MacAvaney, A. Yates, and K. Hui. Contextualized pacrr for complex answer retrieval. In *TREC*, 2017.

[8] R. Maldonado, S. Taylor, and S. M. Harabagiu. UTD HLTRI at TREC 2017: Complex answer retrieval track. In *TREC*, 2017.

[9] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR*, pages 472–479, 2005.

[10] B. Mitra, F. Diaz, and N. Craswell. Learning to match using local and distributed representations of text for web search. In *WWW*, pages 1291–1299, 2017.

[11] F. Nanni, B. Mitra, M. Magnusson, and L. Dietz. Benchmark for complex answer retrieval. In *ICTIR*, pages 293–296, 2017.

[12] R. Nogueira, K. Cho, and U. Patel. New york university submission to TREC-CAR 2017. In *TREC*, 2017.

[13] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, and X. Cheng. Text matching as image recognition. In *AAAI*, 2016.

[14] C. Xiong, Z. Dai, J. P. Callan, Z. Liu, and R. Power. End-to-end neural ad-hoc ranking with kernel pooling. In *SIGIR*, 2017.