Content-Based Weak Supervision for Ad-Hoc Re-Ranking

Sean MacAvaney
IRLab, Georgetown University
sean@ir.cs.georgetown.edu

Andrew Yates
Max Planck Institute for Informatics
ayates@mpi-inf.mpg.de

Kai Hui*
Amazon
kaihuibj@amazon.com

Ophir Frieder
IRLab, Georgetown University
ophir@ir.cs.georgetown.edu

ABSTRACT

One challenge with neural ranking is the need for a large amount of manually-labeled relevance judgments for training. In contrast with prior work, we examine the use of weak supervision sources for training that yield pseudo query-document *pairs* that already exhibit relevance (e.g., newswire headline-content pairs and encyclopedic heading-paragraph pairs). We also propose filtering techniques to eliminate training samples that are too far out of domain using two techniques: a heuristic-based approach and novel supervised filter that re-purposes a neural ranker. Using several leading neural ranking architectures and multiple weak supervision datasets, we show that these sources of training pairs are effective on their own (outperforming prior weak supervision techniques), and that filtering can further improve performance.

ACM Reference Format:
Sean MacAvaney, Andrew Yates, Kai Hui, and Ophir Frieder. 2019. Content-Based Weak Supervision for Ad-Hoc Re-Ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331316

1 INTRODUCTION

A lack of manual training data is a perennial problem in information retrieval [18]. To enable training supervised rankers for new domains, we propose a weak supervision approach based on *pairs* of text to train neural ranking models and a filtering technique to adapt the dataset to a given domain. Our approach eliminates the need for a query log or large amounts of manually-labeled in-domain relevance judgments to train neural rankers, and exhibits stronger and more varied positive relevance signals than prior weak supervision work (which relies on BM25 for these signals).

Others have experimented with weak supervision for neural ranking (see Section 2.2). Our weak supervision approach differs from these approaches in a crucial way: we train neural rankers using datasets of text *pairs* that exhibit relevance, rather than using a heuristic to find pseudo-relevant documents for queries. For instance, the text pair from a newswire dataset consisting of an article's headline and its content exhibits an inherent sense of relevance because a headline often provides a concise representation of an article's content. To overcome possible domain differences between the training data and the target domain, we propose an approach to filter the training data using a small set of queries (templates) from the target domain. We evaluate two filters: an unsupervised heuristic and using the neural ranker itself as a discriminator.

We evaluate our approaches by training several leading neural ranking architectures on two sources of weak supervision text pairs. We show that our methods can significantly outperform various neural rankers when trained using a query log source (as proposed by [5]), the ranker when trained on a limited amount of manually-labeled in-domain data (as one would encounter in a new domain), and well-tuned conventional baselines. In summary, we (1) address existing shortcomings of weak supervision to train neural rankers by using training sources from text pairs, (2) address limitations related to domain differences when training rankers on these sources using novel filtering techniques, and (3) demonstrate the effectiveness of our methods for ad-hoc retrieval when limited in-domain training data is available. Our code is public for validation and further comparisons.[1]

2 BACKGROUND AND RELATED WORK

2.1 Neural IR models

Ad-hoc retrieval systems rank documents according to their relevance to a given query. A neural IR model (*nir*) aims to measure the interaction between a query-document pair (q, d) with a real-value relevance score $rel = nir(q, d)$. The model *nir* is trained to minimize pairwise loss between training triples consisting of a query q, relevant document d^+, and non-relevant document d^-. Neural retrieval models can be categorized as *semantic matching* models (which create dense query/document representations) or as *relevance matching* models (which compare query and document terms directly, often through a query-document similarity matrix). We focus on relevance matching models because they generally show better performance than semantic matching models. We test our approach on three leading neural rankers:

KNRM [16] uses Gaussian kernels applied to each individual similarity score and log-summed across the document dimension. A final dense learning-to-rank phase combines these features into a relevance score.

*Work conducted while the author was at the Max Planck Institute for Informatics.

[1] https://github.com/Georgetown-IR-Lab/neuir-weak-supervision

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331316

Conv-KNRM [4] is a variant of KNRM which applies convolution filters of lengths 1–3 over word embeddings before building cross-matched (matching all kernel lengths with one another) similarity matrices. The rest of the ranking process is identical to KNRM.

PACRR [8] uses square convolutional kernels over the similarity matrix to capture soft n-gram matches. k−max pooling is applied to retain only the strongest signals for each query term, and signals are combined with a dense layer.

2.2 Weak supervision

In IR, weak supervision uses pseudo-relevant information to train a ranking model in place of human judgments. Early work on weak supervision for IR focused on training learning-to-rank models [2], using web anchor text [1] and microblog hashtags [3] for weak supervision. More recently, Dehghani et al. [5] proposed a weak supervision approach that makes use of the AOL query log and BM25 results as a source of training data. Aside from limitations surrounding the availability of query logs, their approach suffers from limitations of BM25 itself: it assumes that documents ranked higher by BM25 are more relevant to the query than documents ranked lower. Others have suggested using a similar approach, but using news headlines [9], also assuming relevance from BM25 rankings. Still others have employed a Generative Adversarial Network to build training samples [15], but this limits the generated data to the types of relevance found in the training samples, making it a complementary approach. In contrast, our approach uses freely-available text *pairs* that exhibit both a high quality and large size.

3 METHOD

3.1 Ranking- and content-based sources

Recall that pairwise training consists of a set of training triples, each consisting of a query q, relevant document d^+, and non-relevant document d^-. We describe two sources of weak supervision training data that replace human-generated relevance judgments: ranking-based and content-based training sources.

Ranking-based training sources, first proposed by [5], are defined by a collection of texts T, a collection of documents D, and an unsupervised ranking function $R(q, d)$ (e.g., BM25). Training triples are generated as follows. Each text is treated as a query $q \in T$. All documents in D are ranked using $R(\cdot)$, giving D^q. Relevant documents are sampled using a cutoff c^+, and non-relevant documents are sampled using cutoff c^-, such that $d^+ \in D^q[0 : c^+]$ and $d^- \in D^q[c^+ : c^-]$. This source is referred to as ranking-based because the unsupervised ranker is the source of relevance.[2]

Content-based training sources are defined as a collection of text pairs $P = \{(a_1, b_1), (a_2, b_2), ..., (a_{|P|}, b_{|P|})\}$ and an unsupervised ranking function $R(q, d)$ (e.g., BM25). The text pairs should be semantically related pairs of text, where the first element is similar to a query, and the second element is similar to a document in the target domain. For instance, they could be heading-content pairs of news articles (the headline describes the content of the article content). For a given text pair, a query and relevant document are selected $(q, d^+) \in P$. The non-relevant document is selected from the collection of documents in $B = \{b_1, b_2, ..., b_{|P|}\}$. We employ $R(\cdot)$ to select challenging negative samples from B^q. A negative cutoff c^- is employed, yielding negative document $d^- \in B^q[0 : c^-] - \{d^+\}$. We discard positive samples where d^+ is not within this range to eliminate overtly non-relevant documents. This approach can yield documents relevant to q, but we assert that d^+ is *more* relevant.

Although ranking-based and content-based training sources bear some similarities, important differences remain. Content-based sources use text pairs as a source of positive relevance, whereas ranking-based sources use the unsupervised ranking. Furthermore, content-based sources use documents from the pair's domain, not the target domain. We hypothesize that the enhanced notion of relevance that content-based sources gain from text pairs will improve ranking performance across domains, and show this in Section 4.

3.2 Filter framework

We propose a filtering framework to overcome domain mismatch that can exist between data found in a weak supervision training source and data found in the target dataset. The framework consists of a filter function $F_D(q, d)$ that determines the suitability of a given weak supervision query-document pair (q, d) to the domain D. All relevant training pairs $(q, d^+) \in S$ for a weak supervision source S are ranked using $F_D(q, d^+)$ and the c_{max} maximum pairs are chosen: $S_D = \max^{c_{max}}_{(q,d^+)\in S} F_D(q, d^+)$. To tune $F_D(\cdot)$ to domain D, a set of *template pairs* from the target domain are employed. The set of pairs T_D is assumed to be relevant in the given domain.[3] We assert that these filters are easy to design and can have broad coverage of ranking architectures. We present two implementations of the filter framework: the $k$max filter, and the Discriminator filter.

***k*-Maximum Similarity (*k*max) filter.** This heuristic-based filter consists of two components: a *representation function* $rep(q, d)$ and a *distance function* $dist(r_1, r_2)$. The representation function captures some matching signal between query q and document d as a vector. Since many neural ranking models consider similarity scores between terms in the query and document to perform soft term matching [4, 7, 8, 16], this filter selects the k maximum cosine similarity scores between the word vectors of each query term and all terms in the document: $\max^k_{d_j \in d} sim(q_i, d_j) : \forall q_i \in q$.

Since neural models can capture local patterns (e.g., n-grams), we use an aligned mean square error. The aligned MSE iterates over possible configurations of elements in the representation by shifting the position to find the alignment that yields the smallest distance. In other words, it represents the minimum mean squared error given all rotated configurations of the query. Based on the shift operation and given two interaction representation matrices r_1 and r_2, the aligned $dist_{kmax}(r_1, r_2)$ is defined as the minimum distance when shifting r_1 for $s \in [1, |r_1|)$. More formally: $dist_{kmax}(r_1, r_2) = \min^{|r_1|}_{s=1} MSE\big(shift(r_1, s), r_2\big)$.

Using these two functions, the filter is simply defined as the minimum distance between the representations of it and any template pair from the target domain:

$$F_D(q, d) = \min_{(q', d') \in T_D} dist(rep(q, d), rep(q', d')) \quad (1)$$

[2] Our formulation of ranking-based sources is slightly different than what was proposed by Dehghani et al. [5]: we use cutoff thresholds for positive and negative training samples, whereas they suggest using random pairs. Pilot studies we conducted showed that the threshold technique usually performs better.

[3] Templates do not require human judgments. We use sample queries and an unsupervised ranker to generate T_D. Manual judgments can be used when available.

Discriminator filter. A second approach to interaction filtering is to use the ranking architecture R itself. Rather than training R to distinguish different degrees of relevance, here we use R to train a model to distinguish between samples found in the weak supervision source and T_D. This technique employs the same pairwise loss approach used for relevance training and is akin to the discriminator found in generative adversarial networks. Pairs are sampled uniformly from both templates and the weak supervision source. Once R_D is trained, all weak supervision training samples are ranked with this model acting as $F_D(\cdot) = R_D(\cdot)$.

The intuition behind this approach is that the model should learn characteristics that distinguish in-domain pairs from out-of-domain pairs, but it will have difficulty distinguishing between cases where the two are similar. One advantage of this approach is that it allows for training an interaction filter for any arbitrary ranking architecture, although it requires a sufficiently large T_D to avoid overfitting.

4 EVALUATION

4.1 Experimental setup

Training sources. We use the following four sources of training data to verify the effectiveness of our methods:

- **Query Log (AOL, ranking-based, $100k$ queries).** This source uses the AOL query log [12] as the basis for a ranking-based source, following the approach of [5].[4] We retrieve ClueWeb09 documents for each query using the Indri[5] query likelihood (QL) model. We fix $c^+ = 1$ and $c^- = 10$ due to the expense of sampling documents from ClueWeb.
- **Newswire (NYT, content-based, $1.8m$ pairs).** We use the New York Times corpus [13] as a content-based source, using headlines as pseudo queries and the corresponding content as pseudo relevant documents. We use BM25 to select the negative articles, retaining top $c^- = 100$ articles for individual headlines.
- **Wikipedia (Wiki, content-based, $1.1m$ pairs).** Wikipedia article heading hierarchies and their corresponding paragraphs have been employed as a training set for the Trec Complex Answer Retrieval (CAR) task [10, 11]. We use these pairs as a content-based source, assuming that the hierarchy of headings is a relevant query for the paragraphs under the given heading. Heading-paragraph pairs from train fold 1 of the Trec CAR dataset [6] (v1.5) are used. We generate negative heading-paragraph pairs for each heading using BM25 ($c^- = 100$).
- **Manual relevance judgments (WT10).** We compare the ranking-based and content-based sources with a data source that consists of relevance judgments generated by human assessors. In particular, manual judgments from 2010 Trec Web Track ad-hoc task (WT10) are employed, which includes $25k$ manual relevance judgments ($5.2k$ relevant) for 50 queries (topics + descriptions, in line with [7, 8]). This setting represents a new target domain, with limited (yet still substantial) manually-labeled data.

[4] Distinct non-navigational queries from the AOL query log from March 1, 2006 to May 31, 2006 are selected. We randomly sample $100k$ of queries with length of at least 4. While Dehghani et al. [5] used a larger number of queries to train their model, the state-of-the-art relevance matching models we evaluate do not learn term embeddings (as [5] does) and thus converge with fewer than $100k$ training samples.

[5] https://www.lemurproject.org/indri/

Training neural IR models. We test our method using several state-of-the-art neural IR models (introduced in Section 2.1): PACRR [8], Conv-KNRM [4], and KNRM [16].[6] We use the model architectures and hyper-parameters (e.g., kernel sizes) from the best-performing configurations presented in the original papers for all models. All models are trained using pairwise loss for 200 iterations with 512 training samples each iteration. We use Web Track 2011 (WT11) manual relevance judgments as validation data to select the best iteration via nDCG@20. This acts as a way of fine-tuning the model to the particular domain, and is the only place that manual relevance judgments are used during the weak supervision training process. At test time, we re-rank the top 100 Indri QL results for each query.

Interaction filters. We use the 2-maximum and discriminator filters for each ranking architecture to evaluate the effectiveness of the interaction filters. We use queries from the target domain (Trec Web Track 2009–14) to generate the template pair set for the target domain T_D. To generate pairs for T_D, the top 20 results from query likelihood (QL) for individual queries on ClueWeb09 and ClueWeb12[7] are used to construct query-document pairs. Note that this approach makes no use of manual relevance judgments because only query-document pairs from the QL search results are used (without regard for relevance). We do not use query-document pairs from the target year to avoid any latent query signals from the test set. The supervised discriminator filter is validated using a held-out set of 1000 pairs. To prevent overfitting the training data, we reduce the convolutional filter sizes of PACRR and ConvKNRM to 4 and 32, respectively. We tune c_{max} with the validation dataset (WT11) for each model ($100k$ to $900k$, $100k$ intervals).

Baselines and benchmarks. As baselines, we use the AOL ranking-based source as a weakly supervised baseline [5], WT10 as a manual relevance judgment baseline, and BM25 as an unsupervised baseline. The two supervised baselines are trained using the same conditions as our approach, and the BM25 baselines is tuned on each testing set with Anserini [17], representing the best-case performance of BM25.[8] We measure the performance of the models using the Trec Web Track 2012–2014 (WT12–14) queries (topics + descriptions) and manual relevance judgments. These cover two target collections: ClueWeb09 and ClueWeb12. Akin to [5], the trained models are used to re-rank the top 100 results from a query-likelihood model (QL, Indri [14] version). Following the Trec Web Track, we use nDCG@20 and ERR@20 for evaluation.

4.2 Results

In Table 1, we present the performance of the rankers when trained using content-based sources without filtering. In terms of absolute score, we observe that the two n-gram models (PACRR and ConvKNRM) always perform better when trained on content-based sources than when trained on the limited sample of in-domain data. When trained on NYT, PACRR performs significantly better. KNRM performs worse when trained using the content-based sources, sometimes significantly. These results suggest that these content-based training sources contain relevance signals where n-grams

[6] By using these stat-of-the-art architectures, we are using stronger baselines than those used in [5, 9].

[7] https://lemurproject.org/clueweb09.php, https://lemurproject.org/clueweb12.php

[8] Grid search: $b \in [0.05, 1]$ (0.05 interval), and $k_1 \in [0.2, 4]$ (0.2 interval)

Table 1: Ranking performance when trained using content-based sources (NYT and Wiki). Significant differences compared to the baselines ([B]M25, [W]T10, [A]OL) are indicated with ↑ and ↓ (paired t-test, $p < 0.05$).

		nDCG@20		
Model	Training	WT12	WT13	WT14
BM25 (tuned w/ [17])		0.1087	0.2176	0.2646
PACRR	WT10	B↑ 0.1628	0.2513	0.2676
	AOL	0.1910	0.2608	0.2802
	NYT	**W↑ B↑ 0.2135**	**A↑ W↑ B↑ 0.2919**	**W↑ 0.3016**
	Wiki	W↑ B↑ 0.1955	A↑ B↑ 0.2881	W↑ 0.3002
Conv-KNRM	WT10	B↑ 0.1580	0.2398	B↑ 0.3197
	AOL	0.1498	0.2155	0.2889
	NYT	**A↑ B↑ 0.1792**	**A↑ W↑ B↑ 0.2904**	**B↑ 0.3215**
	Wiki	0.1536	A↑ 0.2680	B↑ 0.3206
KNRM	WT10	B↑ 0.1764	**0.2671**	0.2961
	AOL	**B↑ 0.1782**	0.2648	**0.2998**
	NYT	W↓ 0.1455	A↓ 0.2340	0.2865
	Wiki	A↓ W↓ 0.1417	0.2409	0.2959

are useful, and it is valuable for these models to see a wide variety of n-gram relevance signals when training. The n-gram models also often perform significantly better than the ranking-based AOL query log baseline. This makes sense because BM25's rankings do not consider term position, and thus cannot capture this important indicator of relevance. This provides further evidence that content-based sources do a better job providing samples that include various notions of relevance than ranking-based sources.

When comparing the performance of the content-based training sources, we observe that the NYT source usually performs better than Wiki. We suspect that this is due to the web domain being more similar to the newswire domain than the complex answer retrieval domain. For instance, the document lengths of news articles are more similar to web documents, and precise term matches are less common in the complex answer retrieval domain [10].

We present filtering performance on NYT and Wiki for each ranking architecture in Table 2. In terms of absolute score, the filters almost always improve the content-based data sources, and in many cases this difference is statistically significant. The one exception is for Conv-KNRM on NYT. One possible explanation is that the filters caused the training data to become too homogeneous, reducing the ranker's ability to generalize. We suspect that Conv-KNRM is particularly susceptible to this problem because of language-dependent convolutional filters; the other two models rely only on term similarity scores. We note that Wiki tends to do better with the 2max filter, with significant improvements seen for Conv-KNRM and KNRM. In thse models, the discriminator filter may be learning surface characteristics of the dataset, rather than more valuable notions of relevance. We also note that c_{max} is an important (yet easy) hyper-parameter to tune, as the optimal value varies considerably between systems and datasets.

5 CONCLUSION

We presented an approach for employing content-based sources of pseudo relevance for training neural IR models. We demonstrated that our approach can match (and even outperform) neural ranking models trained on manual relevance judgments and existing ranking-based weak supervision approaches using two different sources of data. We also showed that performance can be boosted using two filtering techniques: one heuristic-based and one that re-purposes a neural ranker. By using our approach, one can effectively train neural ranking models on new domains without behavioral data and with only limited in-domain data.

Table 2: Ranking performance using filtered NYT and Wiki. Significant improvements and reductions compared to unfiltered dataset are marked with ↑ and ↓ (paired t-test, $p < 0.05$).

			WebTrack 2012–14	
Model	Training	k_{max}	nDCG@20	ERR@20
PACRR	NYT		0.2690	0.2136
	w/ 2max	$200k$	0.2716	0.2195
	w/ discriminator	$500k$	↑ **0.2875**	**0.2273**
	Wiki		0.2613	0.2038
	w/ 2max	$700k$	0.2568	0.2074
	w/ discriminator	$800k$	**0.2680**	**0.2151**
Conv-KNRM	NYT		0.2637	0.2031
	w/ 2max	$100k$	↓ 0.2338	**0.2153**
	w/ discriminator	$800k$	**0.2697**	0.1937
	Wiki		0.2474	0.1614
	w/ 2max	$400k$	**0.2609**	↑ **0.1828**
	w/ discriminator	$700k$	0.2572	0.1753
KNRM	NYT		0.2220	0.1536
	w/ 2max	$100k$	0.2235	↑ **0.1828**
	w/ discriminator	$300k$	**0.2274**	↑ 0.1671
	Wiki		0.2262	0.1635
	w/ 2max	$600k$	↑ **0.2389**	↑ **0.1916**
	w/ discriminator	$700k$	0.2366	0.1740

REFERENCES

[1] Nima Asadi, Donald Metzler, Tamer Elsayed, and Jimmy Lin. 2011. Pseudo Test Collections for Learning Web Search Ranking Functions. In *SIGIR*.

[2] Leif Azzopardi, Maarten de Rijke, and Krisztian Balog. 2007. Building Simulated Queries for Known-item Topics: An Analysis Using Six European Languages. In *SIGIR*.

[3] Richard Berendsen, Manos Tsagkias, Wouter Weerkamp, and Maarten de Rijke. 2013. Pseudo Test Collections for Training and Tuning Microblog Rankers. In *SIGIR*.

[4] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *WSDM '18*.

[5] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural Ranking Models with Weak Supervision. In *SIGIR*.

[6] Laura Dietz and Ben Gamari. 2017. TREC CAR: A Data Set for Complex Answer Retrieval. (2017). http://trec-car.cs.unh.edu Version 1.5.

[7] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *CIKM '16*.

[8] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A Position-Aware Neural IR Model for Relevance Matching. In *EMNLP*.

[9] Bo Li, Ping Cheng, and Le Jia. 2018. Joint Learning from Labeled and Unlabeled Data for Information Retrieval. In *COLING '18*.

[10] Sean MacAvaney, Andrew Yates, Arman Cohan, Luca Soldaini, Kai Hui, Nazli Goharian, and Ophir Frieder. 2018. Overcoming low-utility facets for complex answer retrieval. *Information Retrieval Journal* (2018), 1–24.

[11] Federico Nanni, Bhaskar Mitra, Matt Magnusson, and Laura Dietz. 2017. Benchmark for Complex Answer Retrieval. In *ICTIR '17*.

[12] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A Picture of Search. In *Proceedings of the 1st International Conference on Scalable Information Systems*.

[13] Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia* 6, 12 (2008), e26752.

[14] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. 2005. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, Vol. 2. Citeseer, 2–6.

[15] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *SIGIR*.

[16] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In *SIGIR*.

[17] Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the Use of Lucene for Information Retrieval Research. In *SIGIR*.

[18] Hamed Zamani, Mostafa Dehghani, Fernando Diaz, Hang Li, and Nick Craswell. 2018. Workshop on Learning from Limited or Noisy Data for IR. In *SIGIR*.