Collection Ranking and Selection for Federated Entity Search

Krisztian Balog, Robert Neumayer, and Kjetil Nørvåg

Norwegian University of Science and Technology, Trondheim, Norway,
{krisztian.balog,robert.neumayer,kjetil.norvag}@idi.ntnu.no

Abstract. Entity search has emerged as an important research topic over the past years, but so far has only been addressed in a centralized setting. In this paper we present an attempt to solve the task of ad-hoc entity retrieval in a cooperative distributed environment. We propose a new collection ranking and selection method for entity search, called AENN. The key underlying idea is that a lean, name-based representation of entities can efficiently be stored at the central broker, which, therefore, does not have to rely on sampling. This representation can then be utilized for collection ranking and selection in a way that the number of collections selected and the number of results requested from each collection is dynamically adjusted on a per-query basis. Using a collection of structured datasets in RDF and a sample of real web search queries targeting entities, we demonstrate that our approach outperforms state-of-the-art distributed document retrieval methods in terms of both effectiveness and efficiency.

1 Introduction

The increasing popularity of the Web of Data (WoD) has lead to increasing amounts of data exposed in knowledge bases, like DBPedia or Freebase. Typically, such knowledge repositories contain data about entities (persons, locations, organizations, products, etc.) and the relations between them (such as birthPlace, parentCompany). Entity queries account for a significant portion of web searches [10], therefore, utilizing these structured data sources for retrieval is a fertile and growing area of research.

All existing work on entity search, however, assume that a centralized index, encompassing the contents of all individual data sources, is available. Instead of expending effort to crawl all Web of Data sources—some of which may not be crawleable at all—*distributed information retrieval (DIR)* (or *federated search*) techniques directly pass the query to the search interface of multiple, suitable collections that are usually distributed across several locations [12]. For example, the query "entity retrieval" may be passed to a related collection, such as a bibliographical database for research articles dealing with information retrieval topics, while for the query "San Antonio" collections containing information about the city, such as geonames or DBpedia, might be more appropriate. There are also queries for which multiple databases can contain answers.

We focus on queries that target specific entities, mentioned by their name. While this is a rather specific scenario, Pound et al. [10] estimate that over 40% of web search queries are like this. Therefore, we study a significant problem with practical utility.

We consider a cooperative distributed environment and focus on two sub-problems: collection ranking and collection selection. In Section 3 we discuss state-of-the-art distributed document retrieval techniques that can be applied to the case of entities in a straightforward manner. For collection ranking, we formulate two main families of approaches (lexicon-based and document-surrogate methods) in a unified language modeling framework. This allows for a fair comparison between approaches. For collection selection, we use top-K selection, where K is a fixed rank-based cutoff.

Next, in Section 4, we introduce our novel approach, AENN. The key underlying idea is that instead of relying on sampling, the central broker maintains a complete dictionary of entity names and identifiers. Based on this lean, name-based representation, we generate not only a ranking of collections but also an *expected* ranked list of entities (that is, an approximation of the final results). This can then aid us in the collection selection step to dynamically adjust the number of collections selected, moreover, allows for orientating the selection towards high precision, high recall, or a balanced setting.

As no standard test collection exists for our task, in Section 5 we introduce an experimental testbed based on a collection of Linked Data, described as RDF triples, and a set of queries sampled from an actual Web search engine log. We develop three collections with different characteristics to allow for the generalization of findings.

Our experimental evaluation, reported in Section 6, demonstrates that AENN has merit and provides a viable alternative. On collections where names are available for entities—a reasonable precondition for our approach—AENN's effectiveness (measured in terms of precision and recall) is comparable to that of an idealized centralized approach that has full knowledge of the contents of all collections, while achieving gains in efficiency (i.e,. selecting fewer collections).

2 Related work

The present work lies in the intersection of entity retrieval and distributed information retrieval. In this section we review related work on these two research areas.

Distributed information retrieval (DIR), also known as *federated search*, is ad-hoc search in environments containing multiple, possibly many, text databases [4]. DIR targets cases when documents cannot be copied into a single centralized database for the purpose of indexing and searching, and is concerned with retrieving documents scattered throughout different databases.[1] Based on where the indexes are kept, different architectures can be considered. Most of these, just like our work, assume a central broker that orchestrates the communication with the collections and takes care of the merging of results. Independent of the architecture used, distributed information retrieval involves three important sub-problems: (i) *acquiring resource descriptions*, that is, representing the content of each collection in some suitable form, (ii) *resource selection*, i.e., selecting the collections most relevant to the query (based on the representation built in phase (i)), and, finally, (iii) *result merging*, i.e., combining the results from all selected collections into a single ranked list. Our focus throughout this paper is on (i) and (ii); we discuss relevant DIR literature in relation to our approach in Section 3. For an excellent survey on federated search we refer the reader to [12].

[1] In this paper, we use databases, collections, and resources interchangeably.

Entity retrieval or entity-oriented search is now supported by a range of commercial providers. It has been shown that over 40% of queries in web search target entities [10]. Major web search engines try to cater for such requests by using structured data to generate enhanced result snippets [8]. A plethora of vertical search engines exist to deal with specific entity types: people, companies, services, locations, and so on. Entity search has been gaining increasing attention in the research community too, as recognized by various world-wide evaluation campaigns. The TREC Question Answering track focused on entities with factoid questions and list questions (asking for entities that meet certain constraints) [16]. The TREC 2005–2008 Enterprise track [1] featured an expert finding task: given a topic, return a ranked list of experts on the topic. The TREC Entity search track ran from 2009 to 2011 [2], with the goal of finding entity-related information on the web, and introduced the related entity finding (REF) task: return a ranked list of entities (of a specified type) that engage in a given relationship with a given source entity. Between 2007 and 2009, INEX too featured an Entity Ranking track [6]. There, entities are represented by their Wikipedia page, and queries ask for typed entities (that is, entities that belong to certain Wikipedia categories) and may come with examples. Most recently, the Semantic Search Challenge (SemSearch) ran a campaign in 2010 [9] and 2011 [3] to evaluate the ad-hoc entity search task over structured data. Our experimental setup is based on the SemSearch data set, queries, and relevance judgments, as we explain in Section 5.

3 Baseline Methods

We start by presenting a high-level overview of the distributed approach we use for our entity retrieval task. We assume a cooperative environment, in which the retrieval process is coordinated by a central broker. Figure 1 shows the typical architecture of such a system. When the broker receives an incoming query (Q) from the user (1), it ranks collections based on how likely each would contain results relevant to this query. This is done by comparing the query against summaries of the collections (often referred to as representation sets [12]), kept locally at the broker. Next (2), the broker selects a few of the top ranked collections and requests them to generate results for the input query. In the final step (3), after all selected collections returned their answers, the broker merges the results and presents them, as a single result set, to the user. These three steps are depicted as numbers in circles in Figure 1.

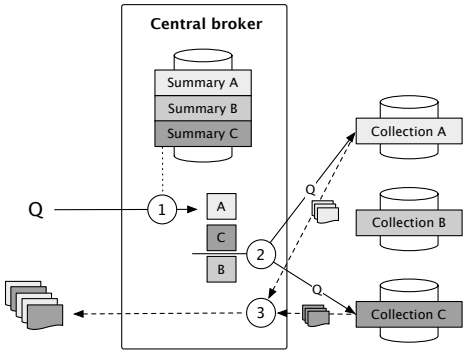

Fig. 1. Schematic overview of a typical broker-based distributed information retrieval system.

In this paper, we focus on the first two steps of this pipeline, as these are the components where our contributions take place. Results merging is a research topic on its

own; to stay focused (and also due to space considerations) we do not perform that step. We note, however, that—assuming a reasonable results merging mechanism—improved collection selection leads to better overall results on the end-to-end task too.

Before proceeding further, it is important to point out that in this section we consider an idealized scenario with a "perfect" central broker. This means that the broker has full knowledge about the contents of each collection. We are aware that this is an unrealistic assumption in practice, but do this for a twofold reason. One, our main research interest is in comparing the effectiveness of collection ranking and selection methods; when doing so, we wish to rule out all other influencing factors, such as the quality of sampling (a technique, typically used for building collection summaries [12, 13]). Two, we want to compare our proposed solution, to be presented in Section 4, against this idealized setting; as we shall show later, our novel approach can deliver competitive performance without making such unrealistic assumptions.

3.1 Collection Ranking

In the collection ranking phase (Step 1 in Figure 1), we need to score collections based on their likelihood of containing entities relevant to the input query. We present two main families of approaches for this task. *Lexicon-based* methods treat and score each collection as if it was a single, large document [5, 14]. *Document-surrogate* methods, on the other hand, model and query individual documents (in our case: entities), then aggregate (estimates) of their relevance scores to determine the collection's relevance [11, 13]. As pointed out earlier, we assume a "perfect" central broker; for lexicon-based methods it means complete term statistics from all collections; for document-surrogate methods it essentially amounts to a centralized index of all entities.

We formalize both strategies in a language modeling framework and rank collections (c) according to their probability of being relevant given a query (q), $P(c|q)$.

Collection-centric collection ranking (CC). Following Si et al. [14], the collection query-likelihood is estimated by taking a product of the collection prior, $P(c)$, and the individual term probabilities:

$$P(c|q) \propto P(c) \cdot \prod_{t \in q} P(t|\theta_c). \tag{1}$$

We set priors proportional to the collection size: $P(c) \propto |c|$. A language model θ_c is built for each collection, by collapsing all entities of c into a single large document and then smoothing it with the global language model. Here, we use Dirichlet smoothing, as we found it to perform better empirically than Jelinek-Mercer smoothing used in [14]; we set the smoothing parameter to the average collection length.

Entity-centric collection ranking (EC). Under this approach, entities are ranked by the central broker, according to their probability of relevance, and the top relevant entities contribute to the collection's query-likelihood score:

$$P(c|q) \propto \sum_{e \in c, r(e,q) < \gamma} P(e|q), \tag{2}$$

where $P(e|q)$ is the query likelihood of the entity, computed using a standard language modeling approach and Dirichlet smoothing (with the average entity representation length used as the smoothing parameter). Further, $r(e, q)$ denotes the rank position of entity e (the top ranked result has rank 0, the second in line has 1, and so on). Finally, γ is a rank threshold, set to 50 based on preliminary experiments; this value has also been commonly used in the literature, see, e.g. [11, 15]. It is worth mentioning that although collection priors are not explicitly included in Eq. 2, larger collections are implicitly favored, as they are more likely to have more relevant results among the top γ.

3.2 Collection Selection

In the previous subsection we established mechanisms for ranking collections in order of relevance to a query. Next, we need to identify a set of collections that are likely to contain most relevant entities; this corresponds to Step 2 in Figure 1. The problem is generally addressed by choosing a fixed cutoff ahead of time; for example, Si and Callan [13] use 5 to 20. We refer to this method as *top-K collection selection*. (SUSHI [15] offers an alternative selection strategy; we will briefly discuss it in Section 4.2.)

Formally, let $r(c, q)$ be the rank of collection c for query q according to the collection ranking component (where the highest ranked collection has rank value 0). The set of selected collections, $S_c(q)$, is then defined as follows: $S_c(q) = \{c|r(c, q) < K\}$, where K is a fixed cutoff value.

4 The AENN Method for Federated Entity Search

In this section we introduce a novel approach to collection ranking and collection selection for federated entity search. We focus on queries that target a particular entity, mentioned by its name. A significant portion of queries in web search are formulated that way [10]. Blanco et al. [3] explain this phenomena as follows: *"users have learned that search engine relevance decreases with longer queries and have grown accustomed to reducing their query (at least initially) to the name of an entity"* [3]. Therefore, the problem we study is a significant one, with practical utility. While traditional collection ranking and selection techniques can immediately be applied, the question arises, whether we can do better by tailoring representations and models to entities.

The central idea of our approach is aptly captured in the acronym AENN: "All that an Entity Needs is a Name." Instead of building traditional collection summaries based on (full) textual representations of entities, as was done in the previous section, we only use their names and maintain a complete dictionary of entity names and identifiers at the central broker. This is a viable alternative as it requires only limited cooperation from the distributed collections (i.e., we need to be able to request a list of entities, with name and ID, they contain) and features minimal network traffic. Based on this lean, name-based representation, we can generate not only a ranking of collections but also an "expected" ranked list of entities (that is, a prediction of the final result list, that we would see after the merging step). This expected ranking serves as a basis of an entity-centric collection ranking, which, in turn, we utilize to aid us in the collection selection step by dynamically adjusting the number of collections selected. It is important to note that AENN is only used for collection ranking and selection; the next step in the

processing pipeline (that we do not perform here) is to request the selected collections to generate a ranked list of entities given the input query. The collections may use the retrieval method of their choosing to perform this local ranking.

4.1 Collection Ranking

Our initial experiments with collection-centric (CC) and entity-centric (EC) collection ranking strategies suggest the two different approaches work best with different queries. Therefore, we expect to maximize performance, by taking a linear combination of the two methods:

$$AENN(c, q) = (1 - \lambda) \cdot CC(c, q) + \lambda \cdot EC(c, q), \tag{3}$$

where $CC(c, q)$ and $EC(c, q)$ are normalized collection scores generated by the corresponding models from Section 3.1. In the lack of training material, we combine the two with equal weights, i.e., set $\lambda = 0.5$. Note that under the AENN approach the central broker only contains the names of entities.

4.2 Collection Selection

A good collection selection method balances between effectiveness and efficiency. That is, select as few servers as possible, to minimize communication costs and latency. On the other hand, avoid being too restrictive, since only the selected collections can contribute to the final result set. The central ranking of entities (ER) plays a vital role in our collection selection mechanism; it may be viewed as a prediction of the final ranked list of results that we expect to see at the end of the merging step. However, it is to be decided how much confidence we wish to assign to this prediction. Next, we define three different selection strategies; one favors precision, another prefers recall, and the final one attempts to balance between the two.

Precision-oriented selection (AENN(p)) We put our trust in the entity-centric (EC) collection ranking and believe that it would yield the highest precision. Subsequently, we only select collections that EC contains. Unlike the following two methods, this selection strategy may decide to "skip" certain collections that otherwise have a high AENN score, but did not contribute any results to the top γ of the ER ranking. This strategy shares similarities with SUSHI [15] in the sense that it only selects collections that are expected to contribute results to the final merged list.

Recall-oriented selection (AENN(r)) The most conservative strategy falls back to the collection-centric (CC) collection ranking, as that has a higher recall than EC. We start at the top of the CC ranking and include collections for selection until we have all from the EC ranking covered. Formally, this method selects the top ρ collections from the AENN ranking, where

$$\rho = \arg\min_{x} \forall c \in EC : r_{CC}(c, q) < x. \tag{4}$$

Balanced selection (AENN(b)) This strategy attempts to balance between precision and recall, by selecting the top collections based on the AENN ranking until all

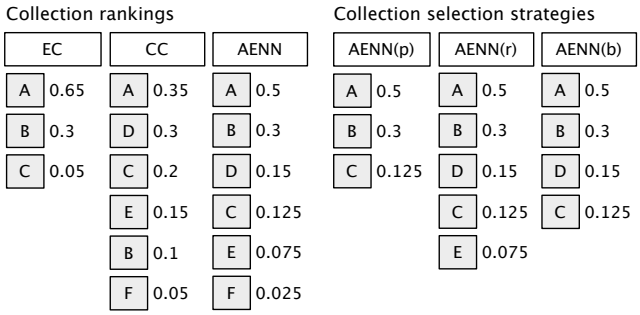

Fig. 2. Illustration of collection selection strategies. The squared letters A,..,F represent collections, the numbers next to them are the collection ranking scores.

collections from EC are covered. Formally, we select the top ρ from AENN such that

$$\rho = \arg\min_{x} \forall c \in EC : r_{AENN}(c, q) < x. \quad (5)$$

Figure 2 illustrates the three strategies on a toy-sized example.

5 Experimental Setup

As no standard test collection exists for our task, in this section we introduce the experimental testbed we have developed for evaluation purposes.

5.1 Distributed Environment

Our setup is based on the test suites of the 2010 and 2011 editions of the Semantic Search (SemSearch) Challenge [3, 9]. The data collection used there is the Billion Triple Challenge 2009 (BTC-2009) dataset. It comprises about 1.14 billion RDF statements and describes entities from domains like *dbpedia.org*, *livejournal.com*, or *geonames.org*.[2] The task addressed at SemSearch is ad-hoc entity search: given a keyword query, targeting a particular entity, provide a ranked list of relevant entities, identified by their URIs. There are two topic sets, consisting of 92 and 50 keyword queries for years 2010 and 2011, respectively. The queries were sampled from web search engine logs. Relevance judgments are provided on a 3-point scale (excellent, fair, and irrelevant) and were collected using crowdsourcing.

To create a distributed environment, we have chosen the top 100 largest second-level domains from BTC-2009, in terms of the number of entities they contain, and indexed them as 100 separate collections. As with typical federated search testbeds, the collections are disjoint and do not overlap [12]. The number of partitions we use follows standard practice, see, e.g., [11, 17], and is considered sufficiently large. We use all SemSearch queries (that is, from both 2010 and 2011) that contain at least one relevant result from one of the top 100 domains; this amounts to a total of 136 queries. We restricted the corresponding relevance judgments to our set of selected collections, but apart from the filtering we use the SemSearch assessments unchanged. We will refer to this test set throughout the paper as *BTC*.

[2] http://vmlion25.deri.ie/

The distribution of collection sizes for the top 100 domains, shown in Figure 3, resembles a Zipfian distribution, where the three largest domains account for almost 30% of the data set. While this is not at all unexpected or unusual, a somewhat unique property of this test set is that relevant entities do not follow the same distribution, but are very highly biased towards the biggest collection, DBpedia. In fact, 73% of all relevant results originate from DBpedia (note that this is not due to our selection of top 100 collections, as DBpedia holds 59% of all known relevant results, without any domain restrictions). To ensure that our findings are not misguided because of this anomaly, we created two more test sets, representing distributed environments with different characteristics.

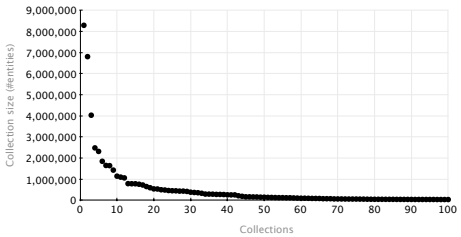

Fig. 3. Distribution of collection sizes in BTC-2009.

BTC\DBpedia is the same as the BTC test set, but the DBpedia collection is excluded. This set, therefore, contains 99 separate collections. Consequently, DBpedia results have also been removed. There are 20 queries for which all relevant results come from DBpedia; these have been omitted from the query set, leaving 116 queries in total. Here, the distribution of relevant results is more evenly distributed—this collection represents a typical linked data collection.

We also look at the *DBpedia* subset, on its own. Instead of using the version that is part of BTC, we considered the full version. Specifically, we used its most recent dump in version 3.7 and indexed all infobox predicates as well as labels and short abstracts of its 8.8M entities. The reason for doing so is that the BTC collection is based on a Web crawl, including some degree of noise. DBpedia, on the other hand, can be considered a more "clean" and uniform collection. We randomly distributed the DBpedia data set into 100 individual collections of equal size. The resulting collections are by no means organized (like topically or temporally) which implies that the relevant documents are also randomly distributed across all collections. Due to its random distribution, we expect it to be the most difficult setup in this context with respect to collection selection (with all sub-collections being very similar to each other in terms of term statistics).

Table 1 presents descriptive statistics of the three test collections we developed.[3]

5.2 Entity Representation

We apply a rather straight-forward entity model: every unique subject found in the collection is an entity. From all subject-predicate-object triples with the entity as subject, we concatenate the object text values into a *content* field. An entity's *name* is given by the object values of a predefined list of predicates (e.g., *foaf:name* or *rdfs:label*).

[3] The resources (queries, relevance judgments, and DBpedia splits) used in our experimental evaluation are available at `http://bit.ly/OzfYK2` and at the first author's homepage.

Table 1. Overview of test collections.

	BTC	**BTC \DBpedia**	**DBpedia**
#Entities	68.8M	60.5M	8.8M
#Collections	100	99	100
#Queries	136	116	130
Avg. #rel. entities /query	14.9	4.8	10.1
Avg. #rel. collections /query	3.4	2.8	9.4

5.3 Ground Truth and Evaluation Metrics

To evaluate *collection ranking*, we use standard IR evaluation metrics: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG). We obtained ground truth from the original SemSearch relevance assessments as follows. For the metrics that work with binary judgments, a domain is considered relevant if it contains at least one entity that was judged relevant for the query. When computing NDCG, we set the gain for each collection to the number of relevant documents the collection contains.

For evaluating *collection selection*, we introduce two metrics, $\mathcal{P}_K$ and $\mathcal{R}_K$, which are rough analogues of the classical precision and recall measures and consider the effectiveness of the collection selection method alone. We base our definitions on the metrics proposed in [7], and use the variant by Thomas and Shokouhi [15] for $\mathcal{R}_K$. If K collections are selected, $\mathcal{P}_K$ is the fraction of collections that contain (any) relevant entities, while $\mathcal{R}_K$ is the ratio of (all) relevant entities held by these collections. Both metrics range from 0 to 1, where higher values are desired. We also measure the average number of collections selected; this serves as our metric of efficiency.

6 Experimental Evaluation

The main research question guiding us is as follows: How does the AENN method compare to traditional document-based methods on the collection ranking and collection selection tasks? We present our evaluation results in the two subsequent sections.

6.1 Collection Ranking

Table 2 reports an overview of collection ranking results. The top two blocks show our baseline methods (CC and EC). For reference, we also included retrieval results for well-known representatives of the two families of methods: CORI [5] for a lexicon-based approach, and ReDDE [13] and two variants of CRCS [11] for document-surrogate methods. For these, we use default parameter settings as suggested in the corresponding publications. All baseline methods are tested with two types of entity representations at the central broker (second column): name-only (N) and content (C). The last block presents our proposed method; recall that it always uses a name-only representation as the central broker in the AENN case maintains only the names of entities.

The first important observation from Table 2 is that our language modeling based baselines (CC and EC) outperform or are in par with existing methods from the literature. Second, as expected, the content-based representation provides better results than

Table 2. Collection ranking results. Significance is tested using a two-tailed paired t-test at the 0.01 level. †/‡denotes significant differences to the CC/EC rows, respectively.

Method	Rep.	BTC MAP	BTC MRR	BTC NDCG	BTC\DBpedia MAP	BTC\DBpedia MRR	BTC\DBpedia NDCG	DBpedia MAP	DBpedia MRR	DBpedia NDCG
Lexicon-based methods										
CC	N	.5542	.8085	.7755	.3451	.4701	.5122	.1731	.2901	.4357
	C	.5741	.8884	.8382	.3544	.4883	.5330	.1766	.2985	.4384
CORI	N	.3299	.4436	.5001	.2595	.3466	.4309	.1753	.2895	.4399
	C	.3254	.4459	.4753	.2808	.3940	.4666	.1627	.2656	.4232
Document-surrogate methods										
EC	N	.5492	.8560	.8094	.3166	.4645	.4782	.3349	.5746	.5802
	C	.6746	.8856	.8533	.5455	.6750	.6871	.3462	.5725	.5839
ReDDE	N	.4877	.7825	.7510	.2521	.3596	.4361	.2165	.3600	.4704
	C	.6292	.8975	.8637	.3827	.4933	.5597	.2532	.4443	.5064
CRCS(l)	N	.5188	.8181	.7807	.2765	.4082	.4565	.2726	.4567	.5209
	C	.6732	.9093	.8801	.4275	.5326	.5938	.3056	.5043	.5473
CRCS(e)	N	.4742	.7025	.6830	.3089	.4629	.4677	.3429	.5945	.5860
	C	.6559	.8747	.8133	.5410	.6858	.6737	.3426	.5921	.5824
Our method										
AENN	N	.6151†‡	.8774†	.8392†‡	.3746†‡	.4756	.5264‡	.3343†‡	.5825†	.5817†

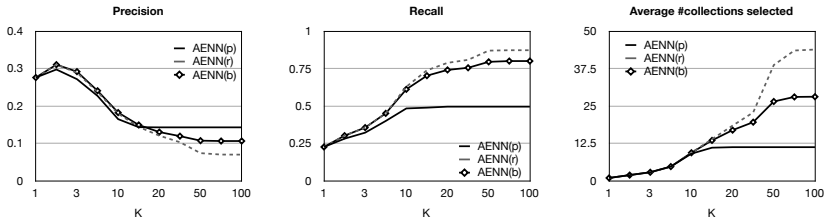

Fig. 4. Comparison of AENN collection selection strategies on the BTC\DBpedia collection.

the name-only one. Apart from a few exceptions, this holds for all methods and collections, however, the difference is rather small for the *DBPedia* collection; this is because a name is available there for each entity. Third, our AENN method successfully combines the two collection ranking strategies. It outperforms all name-only representations for the BTC and BTC\DBPedia collections by 12% and 18% in terms of MAP, respectively. On the DBpedia collection the results are virtually the same as that of the EC run. The differences in MAP are significant for all collections. In sum, the overall performance of the AENN method makes it a viable alternative to other approaches that use a full content-based representation. Moreover, it has additional benefits for collection selection, as we shall see next.

6.2 Collection Selection

First, we compare the three AENN collection selection strategies we devised against each other. In the interest of space, we only show the plots for BTC\DBpedia in Figure 4; the observed very similar behavior on the other two sets. We find that the three methods indeed work as they were originally intended: AENN(p) results in the highest precision and on average it never selects more than 11 collections. AENN(r), on the other hand, can select up to 44 collections; this leads to a high recall, especially for high K values. AENN(b) seems to be able to find the golden middle between the two;

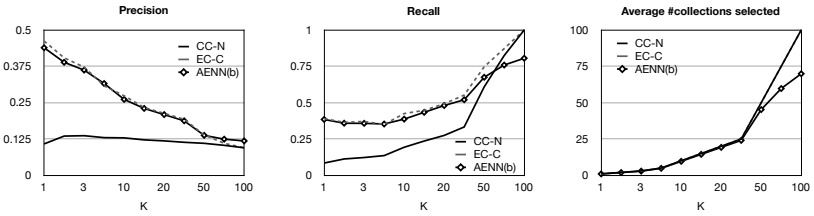

Fig. 5. Comparison of baseline and the AENN(b) collection selection strategies on DBpedia.

it performs well both in terms of precision and recall, while it keeps the number of selected collections reasonably low (28 at most).

Next, we compare the AENN against two baselines, both using top-K selection with a fixed K value: (1) collection-centric using a name-only representation (CC-N), and (2) entity-centric using a content-based representation (EC-C). The latter serves as an upper limit that could be achieved if the broker had a local copy of the full contents of all collections. Figure 5 reports the results, this time on the DBpedia collection. (As we have space to cover only one collection in detail, we chose to include the one that we consider as the most difficult setting.) We find that the balanced variant of the AENN method comes very close to the "oracle" run of EC-C, both on precision and on recall. It can also reduce the average number of selected collections, but only for high K values. This is due to the random distribution of relevant documents, a special characteristic of this setup. On the other two collections the number of selected collections is further reduced. Precision and recall on BTC are very close to the ones shown in Figure 5. Finally, on BTC\DBpedia, where names are missing for many entities, AENN(b) is closer to CC-N than to EC-C, but it always outperforms the former, nevertheless.

7 Conclusions

In this paper, we investigated the feasibility of a federated search architecture for entity retrieval and studied two sub-problems in detail: collection ranking and collection selection. We made an argument that for queries that target a particular entity, which is very frequent in Web search, traditional document-based distributed retrieval techniques might not be the best choice. We proposed a novel method, AENN, that builds on the observation that for such queries the central broker could maintain a complete dictionary of entity names, instead of sampling full representations from each collection. This lean representation can then be utilized for collection selection and can also be used to gear results towards high precision or high recall. Further, we created three test collections from Linked Data and performed an experimental evaluation using these. Our method has shown great promise as it performed just as good as the idealized setting for some collections, in terms of precision and recall, while selecting fewer collections.

As for future work, we believe there is further milage to be gained by improving the name-based ranking of the central broker. We also wish to cover other aspects of distributed entity search that we could not deal with in this paper due to space limitations. In particular, efficiency aspects and various practical considerations (e.g., caching).

Bibliography

[1] K. Balog, I. Soboroff, P. Thomas, N. Craswell, A. P. de Vries, and P. Bailey. Overview of the TREC 2008 enterprise track. In *TREC'08*. NIST, 2009.

[2] K. Balog, A. P. de Vries, P. Serdyukov, P. Thomas, and T. Westerveld. Overview of the TREC 2009 entity track. In *TREC'09*, 2010.

[3] R. Blanco, H. Halpin, D. Herzig, P. Mika, J. Pound, H. Thompson, and T. Duc. Entity search evaluation over structured web data. In *EOS'11*, 2011.

[4] J. Callan. Distributed information retrieval. In *Advances in Information Retrieval*. Kluwer Academic Publishers, 2000.

[5] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In *SIGIR'95*. ACM, 1995.

[6] A. de Vries, A.-M. Vercoustre, J. A. Thom, N. Craswell, and M. Lalmas. Overview of the INEX 2007 entity ranking track. In N. Fuhr, J. Kamps, M. Lalmas, and A. Trotman, editors, *Focused access to XML documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2007)*, volume 4862 of *Lecture Notes in Computer Science*, pages 245–251. Springer, 2008.

[7] L. Gravano and H. Garcia-Molina. Generalizing GlOSS to vector-space databases and broker hierarchies. In *VLDB'95*, 1995.

[8] K. Haas, P. Mika, P. Tarjan, and R. Blanco. Enhanced results for web search. In *SIGIR'11*. ACM, 2011.

[9] H. Halpin, D. M. Herzig, P. Mika, R. Blanco, J. Pound, H. S. Thompson, and D. T. Tran. Evaluating ad-hoc object retrieval. In *IWEST'10*, 2010.

[10] J. Pound, P. Mika, and H. Zaragoza. Ad-hoc object retrieval in the web of data. In *WWW'10*. ACM, 2010.

[11] M. Shokouhi. Central-rank-based collection selection in uncooperative distributed information retrieval. In *ECIR'07*. Springer-Verlag, 2007.

[12] M. Shokouhi and L. Si. Federated search. *Foundations and Trends in Information Retrieval*, 5:1–102, 2011.

[13] L. Si and J. Callan. Relevant document distribution estimation method for resource selection. In *SIGIR'03*. ACM, 2003.

[14] L. Si, R. Jin, J. Callan, and P. Ogilvie. A language modeling framework for resource selection and results merging. In *CIKM'02*. ACM, 2002.

[15] P. Thomas and M. Shokouhi. SUSHI: scoring scaled samples for server selection. In *SIGIR'09*. ACM, 2009.

[16] E. Voorhees. Overview of the TREC 2004 question answering track. In *TREC'04*. NIST, 2005.

[17] J. Xu and W. B. Croft. Cluster-based language models for distributed retrieval. In *SIGIR'99*. ACM, 1999.