# Appendix for Reproducability: Retrieving Entities and Support Passages for Wikipedia Construction

## Abstract

This appendix accompanies the paper under submission to ICTIR 2016. It describes all features used in technical detail with the goal of ensuring reproducibility of the work.

## 1. Problem Statement

We view knowledge portfolio construction process as a retrieval/ranking task.

**Problem Statement.**
We view the knowledge portfolio construction process as a nested retrieval/ranking task. Given a free text query $Q$, a knowledge base of entities, and a text corpus with entity links to the knowledge base, we want to retrieve:

i) entities $E$ relevant for the query;

ii) for each relevant entity $E$, passages $S$ that explain how it is relevant for the query $Q$.

A related, yet latent task is the identification of which entity aspects are relevant for the input query.

**Example.**
Let us focus, for instance, on the TREC Query 201 raspberry pi[1] (i.e., the small ARM-based computer). Here, the goal is to answer with the entities describing the product (`Raspberry_Pi`), the company (`Raspberry_Pi_Foundation`), the inventor (`Eben_Upton`), the `United_Kingdom` as its place of invention and early adoption, and so on. Furthermore, for each of these entities appropriate text fragments, which capture the relevance for the query, should be given. For instance, "founder and former trustee of the Raspberry Pi Foundation, and now CEO of the Raspberry PI trading company" for entity `Eben_Upton`.[2]

## 2. Approach

**Method overview.**
We address the problem by means of a pipeline, summarized in Figure 1, which consists of the following four steps:

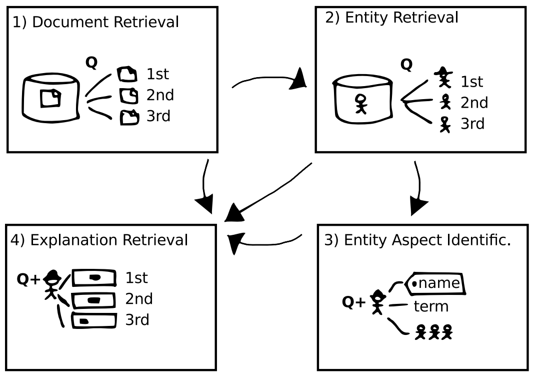

Figure 1: Overview of the knowledge portfolio construction process.

1. **Document Retrieval**: Retrieve relevant documents $D$ from the corpus using the union of three document retrieval models.
2. **Entity Retrieval**: Retrieve relevant entities $E$ from the knowledge base leveraging information from documents using indicators from documents.
3. **Entity Aspect Identification**: For each entity $E$, infer a distribution over neighboring entities $E'$, terms (i.e., words) $W$, and names $A$ using indicators from documents and knowledge base.
4. **Explanation Retrieval**: For each relevant entity $E$, use the inferred aspects to identify text passages $S$ from the documents that are relevant explanations for the entity-query-relevance.

Steps 1, 2, and 4 each consist of a candidate pool building phase and a re-ranking phase. In between these two phases, entity aspects (step 3) and other indicators for query-relevance are extracted by analyzing documents, entities, and entity links. Thus, entity aspects and relevance indicators are used to derive features to separately re-rank entities and, for each entity, re-rank passage candidates. To rank the entities w.r.t. the query, and the passages w.r.t. entities in query-context, we make use of established learning-to-rank methods [Li, 2011], which have been shown to provide robust performance across a wide range of NLP and IR tasks. Moreover, the supervised setting provides us with an intuitive framework to evaluate features and knowledge sources, in order to benchmark their effectiveness for the task at hand.

### 2.1 Feature encoding of scores

In a learning-to-rank setting, each instance (i.e., entity or passage) is represented as a feature vector. At test time, the feature vector is assigned a score by a ranking function, which was learned during training from labeled data. Here, we make extensive use of features that are based on rank scores. Since these are typically not comparable across different queries, we complement for all features (except those marked with †) the rank score with boolean features indicating the contain-

[1]In this paper, we use Sans Serif for words and terms, and `Monospace` for entities.

[2]We focus our example on an entity query. However, our method is able to address *any* open domain query including, e.g., TREC Query 299 pink slime in ground beef, for which we can retrieve entities `Ammonia`, `Beef_Products`, and `United_States_Department_of_Agriculture`.

ment in the top $k$ ranks, with cutoffs $k \in [1, 5, 10, 20, 50]$. This makes it possible to use labeled data to learn the optimal values for the $k$s together with other features.

## 2.2 Document Retrieval

We use the union of three state-of-the art keyword-based document retrieval models: The sequential dependence model (SDM) [Metzler and Croft, 2005], SDM with corpus-based query expansion RM3 [Lavrenko and Croft, 2001], and SDM with knowledge base-query expansion method WikiRM1 [Bendersky et al., 2012]. Frequent terms are down-weighted through Dirichlet smoothing of all retrieval models.

For each retrieval system, we build a document pool from the top 1000 documents. All documents are then tagged with an entity linker, which identifies entity mentions and disambiguates them to the knowledge base. To support easier replicability of our experiments, we use in this work the FACC1 entity link collection [Gabrilovich et al., 2013]. We make use of the contexts surrounding entity links as a source for relevance indicators for the entity in the context of the query. Scanning relevant documents for entity links that refer to a particular entity $E$, we refer to the context surrounding each link with up to 50 (resp. eight) terms as entity-context-50 (entity-context-8).

## 2.3 Entity Ranking

**Candidate Generation.**
We build a candidate set of potentially relevant entities through the union of the top-1000 entities from each of two sources, namely the *knowledge base* as well as the *document collection*. To identify entities from the knowledge base, we first perceive the knowledge base as text. We build a knowledge base index, which associates each entity with text that includes the Wikipedia article, as well as anchor text, names and type labels. We use the sequential dependence model (SDM) to retrieve entities-as-articles from the knowledge base index as query-relevant entities. Next, we add the 1000 most frequently linked entities from the retrieved document pool, using the document retrieval step from Section 2.2.

**Entity Features.**
To rank each entity $E$ against the query we make use of a variety of different features encoding heterogeneous information from the knowledge base and the document collection.

**i) wiki** The rank score of retrieving the entity $E$ from the knowledge base (we use Wikipedia as entity vocabulary, see Section 3). The ranking of articles can be interpreted as a distribution over entities which is obtained by exponentiating and renormalizing the retrieval score $s_Q(E)$ of the entity under the query using a sequential dependence retrieval model.

$$p(E \mid Q) = \frac{1}{Z} \exp s_Q(E)$$

Here, $Z$ ensures normalization across the feedback entities.

**ii) docs** The rank score proportional to the frequency of entity links to $E$ in the top-25 documents retrieved from the corpus using $Q$.

$$p(E \mid Q) \propto \sum_D count(\text{entity link to } E)$$

We can also apply a variation on pseudo-relevance feedback which we extend to document annotations. The unaltered relevance model provides a distribution over expansion termd $p(W|Q)$ by integrating over retrieval probability $p(D|Q)$ and language model $p(W|D)$ of documents in the feedback set.

In a similar fashion we can derive a distribution over entities by only considering targets $E$ of entity links instead of words of each document. This leads to computing the features:

$$p(E \mid Q) = \sum_D \left( \sum_E p(E|D) \right) p(D|Q)$$

Few entity linking tools provide uncertain entity targets (with weighted alternatives). This disambiguation confidence distribution can be incorporated into this equation.

**iii) entity-50, entity-8** A pseudo document $D_{50}(E)$ is build for each entity $E$ by concatenating contexts of 50 (resp. 8) terms surrounding entity links to $E$ in the document pool. Alternative rank scores for the entity $E$ are derived under the SDM retrieval score of the pseudo document under the query $Q$.

$$p(E \mid Q) = \frac{1}{Z} \exp s_Q(D_{50}(E))$$

Here, $Z$ ensures normalization across the feedback entities.

**iv) types** The rank score of the semantic types $T$ associated with the query. This is computed on the basis of the distribution of types over the different entities found in the top-25 documents retrieved using $Q$. For that we inspect all entity links and count how often an entity with the given type $T$ is present.

$$p(T \mid Q) \propto \sum_D \sum_{E \in D} count(E \text{ has type } T)$$

Alternatively, we can also apply the relevance model assuming a uniform type distribution per entity $p(T|E) \propto (E \text{ has type } T)$ to arrive at:

$$p(T \mid Q) = \sum_D \left( \sum_E p(T|E) p(E|D) \right) p(D|Q)$$

$$p(T|E) \propto \begin{cases} 1 & E \text{ has type } T \\ 0 & \text{otherwise.} \end{cases}$$

To account for different levels of concept granularity across different vocabularies, we use Wikipedia categories as well as Freebase types.

## 2.4 Entity Aspect Properties

For the task of entity-passage ranking it is essential to identify which aspects of an entity $E$ are most relevant in the context of the query $Q$. Here, we model different aspects of a entity through distributions over three kinds of 'vocabularies', namely related entities $E'$, terms (i.e., words) $W$, and names $A$.

Aside from the entity identifier itself, we can derive the following prominently known aspects from the knowledge base alone.

**i) Terms from the KB** Normalized term frequencies from the knowledge base entry of $E$.

$$p(W \mid E) \propto count(W \in E\text{'s article})$$

**ii) Names from the KB** Probability distribution of different surface forms being used as anchor text within links to a specific entity. We compute this using Wikipedia's internal hyperlinks, as well as those found on the Web (using an external resource [Spitkovsky and Chang, 2012]).

$$p(M \mid E) \propto \frac{count(M \text{ is anchor text of link to } E)}{\sum_{E'} count(M \text{ is anchor text of link to } E')}$$

This quantity is known as link propability in the entity linking literature and indicates the certainty that this name alias refers to the entity $E$. It is motivated through Bayes' rule (hence the denominator).

Depending on the query, some aspects of a single entity are more important than others. Potentially these are considered side-aspects from the perspective of the entity and are potentially not found in the given knowledge base. Therefore, we include entity aspects which are derived from the contexts in which the entity is mentioned within query-relevant documents.

**iii) Entity-context Neighbors** Probability distribution over entities $E'$ that are co-mentioned in the 50/8-term entity context $c$ of $E$.

$$p(E' \mid E, Q) \propto \sum_{\forall c: E, E' \in c} count(E')$$

**iv) Entity-context Co-Mentions** Probability distribution over surface forms (i.e., names) $M_{E'}$ of any entity $E'$ that is co-mentioned in the 50/8-term entity context $c$ of $E$.

$$p(M_{E'} \mid E, Q) \propto \sum_{\forall c: E, E' \in c} count(M_{E'})$$

**v) Entity-context terms** Normalized term frequencies across the entity contexts of $E$ using 50 (resp 8) terms. [3]

$$p(W \mid E, Q) \propto \sum_{\forall c: E \in c} count(W)$$

## 2.5 Explanation Ranking

**Candidate Generation.**
Given the query $Q$ and an entity $E$, we collect a set of passages as candidate explanations for the entity's relevance. To this end, we use $Q$ to retrieve documents from the collection, and inspect entity links therein that link to $E$. The context surrounding each link is used as a candidate (we choose a passage size of 50 terms).

**Passage Features.**
For each candidate passage we compute features to capture good explanations for how entity $E$ is relevant for query $Q$. These features are derived from all other steps of the pipeline (Figure 1): entity aspects, entity ranking, and document retrieval. We distinguish features that are computed with the knowledge base alone (**Wiki**) from those that arise through previously retrieved documents and entity passages (**Psg**). Features are additionally presented according to the distribution vocabulary ($X$) they use (Section 2.4), i.e. **entities**, **names**, and **terms**.

These features are derived through distributions of the aspect distribution using the following process:

1. Compute aspect distribution $P(X|E,Q)$.
2. Compute a query expansion by taking the 20 entries $X$ with the highest probability under the distribution. This yields a weighted query with terms $X_1, X_2, \dots$ which are weighted according to $p(X_1|E,Q), p(X_2|E,Q) \dots$.
3. Using the query expansion to retrieve/rank candidate passages using a weighted query likelihood model with Dirichlet smoothing. In the case of names, the SDM model is used instead of query likelihood.
4. The score $s_X(P)$ of the passage $P$ under the expansion gives rise to a set of features as detailed above: the score, the reciprocal rank, and boolean features according to cutoffs.

Given an entity ranking $p_{\text{rank}}(E \mid Q)$ and an entity aspect distribution $p_{\text{aspect}}$ (e.g., Terms from the KB), we can aggregate them into a query-global aspect distribution, defined as:

$$p(X \mid Q) \propto \sum_{E} p_{\text{rank}}(E \mid Q) p_{\text{aspect}}(X \mid E, Q) \quad (1)$$

where $X$ represents each of the three distribution vocabularies over entities, names, and terms.

**i) Wiki-Entity** Query-relevant distribution over entities, as derived from knowledge base ranking (cf. feature **wiki** in Section 2.3, i)).

**ii) Wiki-Terms and Wiki-Names** Terms and names from the knowledge-base aspect distributions of $E$ (cf. Section 2.4, i) and ii)). Additionally, for every entity ranking we derive a query-global aspect distribution (Equation 1) as an additional feature.

**iii) Psg-Entities** The entities in the entity-context neighbors of $E$ (cf. Section 2.4, iii)). In addition, the context-centric features used for entity ranking, i.e., **doc**, **entity-50** and **entity-8** (cf. Section 2.3, ii) and iii)).

**iv) Psg-Names** Names from the entity-context co-mentions of $E$ (cf. Section 2.4, iv)). Furthermore, for every entity ranking, we also include a query-global aggregation as additional feature.

**v) Psg-Terms** Terms from the entity-context terms of $E$ for window sizes of 50 and 8 terms are used (cf. Section 2.4, v)). Additionally, the corresponding query-global aggregations are incorporated.

We include additional features which are inspired by the state-of-the-art.

**vi) Query** The text of each passage is ranked using each of our retrieval models from Section 2.2, namely SDM, SDM+RM3, and SDM+ WikiRM1. Each retrieval model provides us with a separate feature. Furthermore, since this is a common approach in passage retrieval [Bendersky and Kurland, 2008], combining all these features with supervised reranking gives us a strong baseline for evaluation, referred to as **query** $\star$.

[3]Using frequent terms of to rank different entity-contexts promotes centroid passages – cf. previous work in document summarization [Nenkova and McKeown, 2012] and pseudo-relevance feedback [Lavrenko and Croft, 2001].

**vii) Position** † The location of the begin/end of the passage within the document is computed. This promotes passages that are found at the beginning of a document, inline with previous findings in summarization [Nenkova and McKeown, 2012].

**viii) Spam** † As some passages are of generally low quality – e.g., web spam, forum headers, and enumeration of entities – we devise several spam indicators following Bendersky et al. (2011) with the extension to repetitions of digits, entities, and words. These are combined into one spam score through supervised classification in a separate step.

Each of the features is formalized as a distribution over a vocabulary. Since one passage may include more than one entry of the distribution, we aggregate multiple matches into a single feature. We use three aggregator functions (each as one feature): accumulating the scores, the fraction of passage that are matched, and the coverage of the distribution.

## 3. Experimental Setup

### 3.1 Knowledge base

We use the Wikipedia WEX[4] dump from 2012 as knowledge base, which we process to include categories, the anchor text of Wikipedia links and Web pages [Spitkovsky and Chang, 2012], redirects and disambiguation links. The dump is further merged with a Freebase dump from 2012 to include canonical names and types from the Freebase schema.

This knowledge base is indexed with Galago 3.7. Unless otherwise noted, we use SDM model with Dirichlet smoothing to retrieve from the knowledge base corpus. The used hyperparameters are tuned on the TAC KBP corpus for entity linking resulting in Dirichlet smoothing $\mu = 96400$ and SDM parameters $(0.29, 0.21, 0.50)$ not using fielded retrieval (fields flattened into one flat text).

### 3.2 Queries, corpus, entity links

We evaluate our system using queries from standard datasets for Web search such as the TREC Web track from 2013 and 2014, and ClueWeb12 Category A as background corpus.

Entity link annotations are used to connect entity mentions in the ClueWeb corpus to entities in the knowledge base. While there exists many robust entity linking toolkits [Cornolti et al., 2013], for reproducibility we opt for the publicly available annotation from the FACC1 [Gabrilovich et al., 2013] dataset for ClueWeb12/CatA.

### 3.3 Document retrieval models

We first index Clueweb12 Category A with Indri[5], and create a sub-corpus from the top 10,000 results from the query likelihood, relevance model and Terrier baseline runs provided by TREC Web organizers, with the additional augmentation of our own Indri runs with the sequential dependence model (SDM), with corpus-query expansion RM3 and KB-query expansion WikiRM1 method [Bendersky et al., 2012]. We next merge the subset of ClueWeb documents with the FACC1 annotations and index the result with Galago 3.7[6]. The SDM, RM and Dirichlet hyper-parameters were tuned on the smaller Category B subset of Clueweb12 on TREC Web 2013 queries, obtaining a Dirichlet $\mu = 4311$, original query weight $\lambda = 0.55$, and SDM parameters $(0.85, 0.05, 0.10)$ for unigrams, bigrams and window-8 skipgrams, respectively.

[4]`http://download.freebase.com/wex/`

[5]`http://www.lemurproject.org/indri.php`

[6]`http://www.lemurproject.org/galago.php`

### 3.4 Supervised ranking models

We train our supervised re-ranking models with RankLib[7], using the Coordinate Descent method to optimize for Mean Average Precision (MAP) with 20 restarts. We train separate models for re-ranking entities and passages using 5-fold cross validation with consistent, query-based folds across all re-ranking and classification steps. For training and testing, unjudged entries are treated as non-relevant.

### 3.5 Passage spam

We filter passages to remove low-quality passages before re-ranking. To generate training data for the passage-spam filter, we additionally ask human judges to provide us with spam/ham labels as part of the annotation process. The passage-spam filter is trained using scikit-learn's[8] SGD method with Hubert-loss.

### 3.6 Query splits for cross validation

We use k-fold cross validation across queries of TREC Web 2013 and 2014, following query splits consistent listed in Figure 2.

## References

Michael Bendersky and Oren Kurland. 2008. Utilizing passage-based language models for document retrieval. In *Advances in Information Retrieval*, pages 162–174. Springer.

Michael Bendersky, W Bruce Croft, and Yanlei Diao. 2011. Quality-biased ranking of web documents. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 95–104. ACM.

Michael Bendersky, Donald Metzler, and W Bruce Croft. 2012. Effective query formulation with multiple information sources. In *Proc. of WSDM-12*, pages 443–452.

Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. 2013. A framework for benchmarking entity-annotation systems. In *Proc. of WWW-13*, pages 249–260.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. Facc1: Freebase annotation of ClueWeb corpora, version 1. (Release date 2013-06-26, Format version 1, Correction level 0).

Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127. ACM.

Hang Li. 2011. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1):1–113.

[7]`http://lemurproject.org/ranklib.php`

[8]`http://scikit-learn.org`

1. 231,205,221,233,204,235,246,249,202,211,281,255,271,283,254,285,296,299,252,261
2. 230,242,238,236,219,232,225,218,209,210,280,292,288,286,269,282,275,298,259,260
3. 214,222,212,234,217,226,247,223,224,248,264,272,262,284,267,276,297,273,274,298
4. 213,229,244,227,215,206,240,228,250,237,263,279,294,277,265,256,290,278,300,287
5. 203,208,243,220,216,201,245,241,239,207,253,258,293,270,266,251,295,291,289,257

Figure 2: k-fold crossvalidation splits according to topic numbers of TREC Web 2013 and 2014.

Donald Metzler and W Bruce Croft. 2005. A markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 472–479. ACM.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer.

Valentin I. Spitkovsky and Angel X. Chang. 2012. A cross-lingual dictionary for english wikipedia concepts. In *Proc. of LREC-12*, pages 3168–3175.