The University of Amsterdam at TREC 2010
Session, Entity, and Relevance Feedback

Marc Bron Jiyin He Katja Hofmann Edgar Meij
Maarten de Rijke Manos Tsagkias Wouter Weerkamp

ISLA, University of Amsterdam
`http://ilps.science.uva.nl/`

Abstract: We describe the participation of the University of Amsterdam's ILPS group in the session, entity, and relevance feedback track at TREC 2010. In the Session Track we explore the use of blind relevance feedback to bias a follow-up query towards or against the topics covered in documents returned to the user in response to the original query. In the Entity Track REF task we experiment with a window size parameter to limit the amount of context considered by the entity co-occurrence models and explore the use of Freebase for type filtering, entity normalization and homepage finding. In the ELC task we use an approach that uses the number of links shared between candidate and example entities to rank candidates. In the Relevance Feedback Track we experiment with a novel model that uses Wikipedia to expand the query language model.

1 Introduction

This year the Information and Language Processing Systems (ILPS) group of the University of Amsterdam participated in the session, entity and relevance feedback tracks. In this paper, we describe our participation for each of these tracks, in three largely independent sections: Section **??** on our session track participation, Section **??** on our participation in the entity track, and Section **??** on our work in the relevance feedback track. We detail the runs we submitted, present the results of the submitted runs, and, where possible, provide an initial analysis of these results. Before doing so, we describe the shared retrieval approach in Section **??**. We conclude in Section **??**.

2 Retrieval Framework

In this section we describe our general approach for each of the tracks in which we participated this year. We employ a language modeling approach to IR and rank documents by their log-likelihood of being relevant given a query. Without presenting details here, we only provide our final formula for ranking documents, and refer the reader to (**?**) for the steps of deriving this equation:

$$\log P(D|Q) \propto \log P(D) + \textstyle\sum_{t \in Q} P(t|\theta_Q) \cdot \log P(t|\theta_D). \quad (1)$$

Here, both documents and queries are represented as multinomial distributions over terms in the vocabulary, and are referred to as *document model* (θ_D) and *query model* (θ_Q), respectively. The third component of our ranking model is the *document prior* ($P(D)$), which is assumed to be uniform, unless stated otherwise. Note that by using uniform priors, Eq. **??** gives the same ranking as scoring documents by measuring the KL-divergence between the query model θ_Q and each document model θ_D, in which the divergence is negated for ranking purposes (**?**).

2.1 Modeling

Unless indicated otherwise, we smooth each document model using a Dirichlet prior:

$$P(t|\theta_D) = \frac{n(t,D) + \mu P(t)}{\sum_t n(t,D) + \mu}, \quad (2)$$

where $n(t,D)$ indicates the count of term t in D and $P(t)$ indicates the probability of observing t in a large background model such as the collection:

$$P(t) = P(t|C) = \frac{\sum_D n(t,D)}{|C|}. \quad (3)$$

μ is a hyperparameter that controls the influence of the background corpus which we set to the average document length.

As to the query model θ_Q, we adopt the common approach to linearly interpolate the initial query with an expanded part (**??**):

$$P(t|\theta_Q) = \lambda_Q P(t|\hat{\theta}_Q) + (1 - \lambda_Q) P(t|Q), \quad (4)$$

where $P(t|Q)$ indicates the MLE on the initial query, $P(t|\hat{\theta}_Q)$ indicates the MLE of the expanded part, and the parameter λ_Q controls the amount of interpolation.

2.2 Significance testing

Throughout the paper we use the Wilcoxon signed-rank test to test for significant differences between runs. We report on significant increases (or drops) for $p < .01$ using $^\blacktriangle$ (and $^\blacktriangledown$) and for $p < .05$ using $^\vartriangle$ (and $^\triangledown$).

2.3 Clueweb

All the tracks we participated in this year make use of the Clueweb document collection. We do not use any form of stemming and remove a conservative list of 588 stopwords. We index the headings, titles, and contents as searchable fields and do not remove any HTML tags.

3 Session Track

The goal of the TREC Session track is to find out how retrieval systems perform over the course of a session and whether taking a previous query into account can help improve retrieval performance for a follow-up query. The current setting considers sessions consisting of an original query and one follow-up query that constitutes either a generalization, specialization, or a parallel move.

Our submission for the TREC Session track explores the use of blind relevance feedback to bias a follow-up query towards or against the topics covered in documents that were returned to the user in response to the original query. Blind relevance feedback takes the most discriminative terms from a set of documents retrieved for a query, and uses these to build a query model that incorporates information about the topic underlying the documents. We apply this method to an initial, diverse result list. Below we explain our approach in detail, and give an overview of our results.

3.1 Approach

Currently, little is known about users' expectations about retrieval systems' behavior throughout a session. Therefore, we based our submission on the following intuitions. Without contextual information about the users' preferred interpretation of a query, a retrieval system can return a standard retrieval run (cf. *Retrieval approach*, below). If the query is ambiguous, it may be better to provide a diverse result list to increase the likelihood of providing at least some relevant documents for different possible interpretations of the query (cf. *Diversification*). When additional information from a previous query can be taken into account, search results can be more focused. In our submission we use pseudo relevance feedback to combine information about the topics covered by the two queries (cf. *Pseudo Relevance Feedback*).

Retrieval approach Our retrieval system uses the framework explained above (cf. §**??**). We use a Dirichlet prior with $\mu = 1600$. Queries are constructed to emphasize phrases, as these are often found in web queries. Phrases and individual terms are combined with equal weights. An example query is shown below.

```
<query>
  <number>1</number>
  <text>#weight( 0.5 #1(legal advice) 0.5
        #combine(legal advice) )</text>
</query>
```

Figure 1: Example query, combining individual terms and phrases.

Retrieval runs are post-processed to filter out category and redirection pages from wikipedia, and to place the top result from wikipedia at the top of the result list.

Diversification We diversify runs following a *topic model-based approach*. It models documents as a mixture of topics and constructs a final result list by re-ranking an initial list so that as many topics as possible are represented in the top ranked documents.

Our approach is inspired by previous work on diversifying a ranked list with Maximal Marginal Relevance (MMR) by **?** and based on a topic modeling approach, i.e., LDA (**?**). It treats the re-ranking problem as a procedure of selecting a sequence of documents, where a document is selected depending on both its relevance with respect to the query and the documents that have already been selected before it, so as to have a set of documents that (i) are most relevant to the query and (ii) represent most if not all topical aspects.

We proceed as follows. First, we use LDA to extract 10 topics from the top 500 documents of the baseline retrieval run described above, so that each document is represented as a mixture of these 10 topics. We then start the re-ranking procedure by selecting the top relevant document in the initial list as the first document in the new ranked list. Then, we select a next document that can maximize the expected joint probability of presence of all topics in the selected result set. Since the sum of topic proportions within a document equals 1, the maximum joint probability (i.e., product of the probabilities of presence of each topic) occurs when the topics have equal proportion in the selected set. On the other hand, we use the retrieval score from the initial run as a prior probability that a document is selected as the next one, so as to take into account the relevance relation between the document and the original query.

Formally, given a query Q, a set of candidate documents $Ca = \{D_j\}_{j=1}^n$ and a set of latent topics $T = \{t_i\}_{i=1}^m$, a document is selected from Ca for inclusion in the ranked list S such that

$$\arg\max_{D \in Ca} \quad P(Q|D) \prod_{i=1}^{m} P(t_i \in S \cup \{D\}), \qquad (5)$$

where $P(Q|D)$ is the query likelihood between the query Q and document D calculated as in a standard language modeling framework. The term $P(t_i \in S \cup \{D\})$ denotes the probability of a topic being present in the set $S' = S \cup \{D\}$, which is estimated by

$$P(t_i \in S') = \sum_{D_j \in S'} P(t_i \in D_j) P(D_j). \tag{6}$$

Pseudo Relevance Feedback RL3 runs are generated using blind relevance feedback as follows. First, retrieval runs for the original and the follow-up query individually are generated, using the *baseline* method and *diversification* as described above. From the top-10 documents of these runs, the 10 most discriminative relevance feedback terms are extracted to form the sets of expansion terms E_O (expansion terms extracted from results for the original query) and E_F (expansion terms extracted from results for the follow-up query). These are combined to form the query expansion E as follows:

RF1 $E = E_O$ - only use feedback terms extracted from the top-ranked results of the original query.

RF2 $E = E_F \setminus E_F$ - take feedback terms generated from results for the follow-up query and remove terms that were also extracted from results for the original query.

RF3 $E = E_O \cup E_F$ - combine relevance feedback terms of both queries.

These three approaches implement the following intuitions. First, we assume that results returned for the original query were helpful and can be used to focus or disambiguate results for the follow-up query. Second, we cover the assumption that results for the original query were not helpful. Finally, we consider the possibility that the underlying topic may best be represented by both queries.

As a final step in generating results when taking an original query into account, we remove documents that have been displayed in the top-10 of the response to the original query from the result list shown for the follow-up query.

3.2 Runs

All submitted runs were automatic category A runs.

uvaExt*.RL1 standard retrieval run using the original query + diversification using LDA

uvaExt1.RL2 standard retrieval run using the follow-up query

uvaExt1.RL3 combines uvaExt1.RL1 and uvaExt1.RL2 using RF1.

uvaExt2.RL2 standard retrieval run using the follow-up query + blind relevance feedback using the follow-up query

uvaExt2.RL3 combines uvaExt1.RL1 and uvaExt1.RL2 using RF2.

uvaExt3.RL2 standard retrieval run using the follow-up query + diversification using LDA

uvaExt3.RL3 combines uvaExt1.RL1 and uvaExt1.RL2 using RF3.

3.3 Results

Results for our submissions are listed in Table **??**. Listed is $nsDCG@10.RL_{12,13}$ with and without taking duplicate documents into account. $nsDCG@10.RL_{12}$ measures session performance when the original query was not taken into account. $nsDCG@10.RL_{13}$ measures session performance when the original query was taken into consideration.

Table 1: Results. nsDCG@10 for RL_{12} and RL_{13}.

	with duplicates		*w/o duplicates*	
runID	n..RL_{12}	n..RL_{13}	n..RL_{12}	n..RL_{13}
uvaExt1	0.1356	0.1320	0.1416	0.1398
uvaExt2	0.1260	0.1297	0.1317	$0.1373^{\vartriangle}$
uvaExt3	0.1262	0.1279	0.1311	0.1356

We find that the overall best performance is achieved by run *uvaExt1* when duplicates are removed and the original query was *not* taken into account. This run retrieves document lists for each query. In all other cases, the follow-up run was diversified and in these cases, performance improves when taking the original query into account. In one case (*uvaExt2*, no duplicates), this improvement is statistically significant.

Performance when measured after removing duplicate documents improves in all cases. This is expected, as we remove duplicate documents that were previously displayed to the user at high ranks.

4 Entity Track

The Entity Track consists of two tasks this year. The main task is the Related Entity Finding (REF) task introduced last year, where the goal is to find homepages of entities given a source entity, relation and target type. New this year is the second task: Entity List Completion (ELC). In the ELC task the goal is to find entities in structured data given a source entity, relation, target type and example entities.

4.1 Related Entity Finding Approach

In the REF task, we continue our experiments with co-occurrence models. This year we use a generative model to rank candidate entities that combines the co-occurrence between the source entity and candidate entities with evidence for relevance to the relation from the snippets in which they co-occur (**?**). To the ranking provided by this model we apply type filtering based on Freebase and homepage finding using candidate entity names as queries to a web search engine. We briefly recall the derivation of our co-occurrence model below, followed by a description of each of the components.

We formulate the entity ranking problem as follows: rank candidate entities (e) according to $P(e|E,T,R)$, where E is the source entity, T is the target type, and R is the relation described in the narrative.

Instead of estimating this probability directly, we use Bayes' rule and reformulate it into:

$$P(e|E,T,R) \quad = \quad \frac{P(E,T,R|e)\cdot P(e)}{P(E,T,R)}. \tag{7}$$

Next, we drop the denominator as it does not influence the ranking of entities, and derive our final ranking formula as follows:

$$\begin{aligned}
&P(E,T,R|e)\cdot P(e) \\
&\quad = P(E,R|e)\cdot P(T|e)\cdot P(e) && (8)\\
&\quad = P(E,R,e)\cdot P(T|e) \\
&\quad = P(R|E,e)\cdot P(E,e)\cdot P(T|e) \\
&\quad = P(R|E,e)\cdot P(e|E)\cdot P(E)\cdot P(T|e) && (9)\\
&\quad \overset{\text{rank}}{=} P(R|E,e)\cdot P(e|E)\cdot P(T|e) && (10)
\end{aligned}$$

In (**??**) we assume that the type is independent of the source entity E and the relation R. Next, we rewrite $P(E,R|e)$ to $P(R|E,e)$ so that it expresses the probability that relation R is generated by the two (co-occurring) entities (e and E). Finally, we rewrite $P(E,e)$ to $P(e|E)\cdot P(E)$ in (**??**) as the latter is a more convenient form for estimation, and we drop $P(E)$ in (**??**) as it does not influence the ranking (for a fixed source entity E). Given equation (**??**) we are left with the following components:

- $P(e|E)$: pure co-occurrence model,
- $P(R|E,e)$: context dependent model, and
- $P(T|e)$: type filtering.

Pure co-occurrence model We use χ^2 to express the strength of associations between the source entity and candidates, without considering the nature of their relation:

$$\text{cooc}_{\chi^2}(e,E) = \frac{N\cdot(c(e,E)\cdot c(\overline{e},\overline{E}) - c(e,\overline{E})\cdot c(\overline{e},E))^2}{c(e)\cdot c(E)\cdot(N-c(e))\cdot(N-c(E))},$$

where N is the total number of documents, and $\overline{e}$, $\overline{E}$ indicate that e, E do not occur, respectively (i.e., $c(\overline{e},\overline{E})$ is the number of documents in which neither e or E occurs).

Context-dependent model We take the context surrounding a source entity and candidate entity into account by constructing a co-occurrence language model (θ_{Ee}) from the contexts in which a source entity and candidate co-occur. By assuming independence between the terms in the relation R we arrive at the following estimate:

$$P(R|E,e) = P(R|\theta_{Ee}) = \prod_{t\in R} P(t|\theta_{Ee})^{n(t,R)}, \tag{11}$$

where $n(t,R)$ is the number of times t occurs in R. To estimate the co-occurrence language model θ_{Ee}, we collect the snippets of a certain window size (w) in which the two entities co-occur and obtain term probabilities as follows:

$$P(t|\theta_{Ee}) = \frac{n(t,E,e)+\mu\cdot P(t)}{\sum_t' n(t',E,e)+\mu}, \tag{12}$$

where μ is set to the average length of all the snippets in which E and e co-occur. Finally we define $n(t,E,e)$ as:

$$n(t,E,e) = \sum_d n(t,E,e,d,w), \tag{13}$$

where $n(t,E,e,d,w)$ is the number of times term t co-occurs with both entities E and e in document d within a distance of at most w term positions from either E or e. The window size w parameter will be determined empirically.

Type filtering In last year's Entity Track DBpedia was used heavily to obtain type information for filtering. One issue when using type information from a closed vocabulary is handling entities outside the vocabulary. We experimented with Freebase[1] as alternative knowledge source. Freebase is a collection of entities gathered from multiple structured data sources, i.e., DBpedia and WordNet, and therefore it covers more entities than DBpedia. Freebase contains information about properties of entities and relations between entities encoded in RDF (**?**). RDF data is structured as subject, predicate, and object triples. A relation between two entities is represented by having the entities as subject and object and the predicate specifies the type of relation. A property is represented by having a text string (literal) as object instead of an entity. One property of Freebase is that it maintains a link to the original data source an entity originates from, i.e., the DBpedia URI of an entity.

To perform filtering we first created a manual mapping of the Freebase category labels to the target types T_{fb}. Then for each candidate entity with an entry in Freebase we obtain the category labels associated with the candidate (e_{fb}). Given this mapping we estimate $P(T|e)$ as

[1] http://wiki.freebase.com

follows:

$$P(T|e) = \begin{cases} 1 & \text{if } e_{fb} \cap T_{fb} \neq \emptyset \\ 0 & \text{otherwise,} \end{cases}$$

Hompage finding Another property associated with some of the entities in Freebase is the URL of the entity homepage. To obtain additional URLs we submit entity strings to a web search engine and collect the top 10 URLs. If we find a match for the Freebase URL in ClueWeb then we use it as the entity homepage, otherwise we use the highest ranked URL returned by the search engine that is found in ClueWeb. If no homepage is found we remove the entity from the candidates.

Entity normalization We perform a preliminary experiment with entity normalization and again turn to Freebase to provide a mapping of variants to a canonical entity. We consider all strings that are linked to the same Freebase ID with a name predicate as variants of the same entity. Normalization requires the following changes to the pure co-occurrence component: $\mathrm{cooc}_{\chi^2}(V_e, V_E) =$

$$\frac{N \cdot (c(V_e, V_E) \cdot c(\overline{e}, \overline{E}) - c(V_e, \overline{E}) \cdot c(\overline{e}, V_E))^2}{c(V_e) \cdot c(V_E) \cdot (N - c(V_e)) \cdot (N - c(V_E))},$$

where V_E is the set of variants association with E and $c(V_E, V_e)$ is defined as: $\sum_{v_E \in V_E} \sum_{v_e \in V_e} c(v_e, v_E)$.

The context dependent model becomes:

$$P(R|V_E, V_e) = P(R|\theta_{V_E V_e}) = \prod_{t \in R} P(t|\theta_{V_E V_e})^{n(t,R)}.$$

We follow the estimation in Equation **??** but define $n(t, V_E, V_e)$ as:

$$\sum_{v_E \in V_E} \sum_{v_e \in V_e} \sum_{d} (t, v_E, v_e, d, w).$$

For the typefiltering component we only consider the type of the canonical form of the variants.

4.1.1 Pre-processing

The main challenge this year was to construct a related entity finding system that runs on the complete ClueWeb Cat A collection. For Named Entity Recognition (NER) we used the Stanford tagger (**?**) which resulted in almost 2 billion unique entities. Removing entities with frequency lower than 5, replacing diacritics and removing entities longer than 128 characters, left us with 148 million entities.[2] We then replaced entities by a unique identifier and indexed the documents using the Indri toolkit [3] resulting in 10 indexes one for each part of ClueWeb Cat A.

[2]Available at http://ilps.science.uva.nl/resources/cikm2010-entity.

[3]http://www.lemurproject.org/indri

4.1.2 REF Experiments

In our experiments we focus on the window size parameter and our entity normalization approach. We submitted the following four official runs:

ilpsA500 An automatic run with a window size of 500.

ilpsM50 A manual run with a window size of 50.

ilpsM50var A run with entity normalization.

ilpsM50agfil A run with strict type filtering.

In addition we generated the following un-official run:

ilpsA50 An automatic run with a window size of 50.

Note that in manual runs, apart from manually removing stop words and adding the base forms of verbs and singular forms of plural terms to the narrative, there was no manual interaction with the system.

4.1.3 REF Results

We first look at the affect of the window size parameter. The first two rows in Table **??** show the results for a run with a window size of 500 (ilpsA500) and a run with a window size of 50 (ilpsA50). We observe that using a smaller widow size leads to a small gain in performance. This is in accord with the intuition that entities that occur close to the source entity are more strongly associated with it and that restricting context is effective in removing irrelevant entities. We plan further experiments in which we determine the optimal value of the window size parameter.

The intuition to perform entity normalisation is that by collapsing variants we obtain more reliable co-occurrence statistics and more complete co-occurrence models. Row 4 and 5 in Table **??** show the results for a run with (ilpsM50var) and a run without (ilpsM50) entity normalisation. The drop in performance we observe when using normalisation is caused by abbreviations of entities identical to common terms, e.g., us and US (United States), which distort the co-occurrence statistics and context models.

Our best run (ilpsM50agfil) uses a more aggressive form of type filtering, i.e., entities that belong to the target type but also to other types are ranked lower proportional to the number of non-target types they belong to.

Finally, we note that although our scores approach the average of the per topic median nDCG_R (0.08301), both are low compared to the average of the per topic maximum. In part this is due to the change to the Category A corpus. Our co-occurrence models take all contexts in which a source and candidate entity co-occur into account. While this is a competitive approach when applied to a high quality corpus such as Wikipedia and to a certain degree Category B (**?**), it suffers from low quality contexts introduced by the increase in corpus size.

run	pri_ret	nDCG_R
ilpsA500	88/779	0.0460
ilpsA50	94/779	0.0684
ilpsM50	94/779	0.0692
ilpsM50var	77/779	0.0571
ilpsM50agfil	**99/779**	**0.0718**

Table 2: Results of the related entity finding runs.

4.2 Entity List Completion Approach

In our participation of the ELC task we investigate two intuitions. First, candidate entities that are more similar to the example entities in terms of the number of links they share should be ranked higher than entities that are less similar. Second, entities that link to objects that share words with the narrative, source entity and target type should be ranked higher than entities that do not.

The ELC task uses the Billion Triple Corpus (BTC) a set of structured data consisting of RDF triples. Both entities and relations are represented by URIs. The source entity and candidate entities given in the topic are represented by URIs as well, while the relation is given as a text string. In order to collect candidate entities we first obtain all objects that have one of the example entities as subject to form the set of example objects O_x. Candidates are all entities that have an object in common with the example entities:

$$C = \{s : triple(s,p,o), o \in O_x\},$$

where $triple(s,p,o)$ is a triple in the BTC with subject s, predicate p and object o.

ilpsSetOL: baseline run To rank a candidate entity (c) based its links we take the predicates and objects from the set of triples which have the candidate as subject and store them as predicate-object tuples:

$$T(c) = \{(p,o) : triple(s,p,o), s = c\}.$$

We do this analogously for each example entity and calculate the candidate ranking score as the average Jaccard coefficient between the candidate entity and the example entities:

$$avgJ(c,X) = \frac{1}{|X|}\sum_{x\in X}\frac{|T(c)\cap T(x)|}{|T(c)\cup T(x)|},$$

where x is an example entity from the set of examples X and $|X|$ is the size of the example set. Given this set we used two ranking approaches given below.

ilpsSetOLnar: baseline combined with word overlap In our second run we combine the predicate-object tuple set overlap with the word set overlap between an entities word set representation and a topics word set representation. A candidate word set is the set of unique terms obtained by parsing an entity's predicate-object tuples ($W(c)$). The set of unique terms from the source entity, narrative and target type ($W(E,R,T)$) form the topic representation. We calculate the word overlap as follows:

$$wo(W(c),W(E,R,T)) = \frac{|W(c)\cap W(E,R,T)|}{|W(E,R,T)|\cdot 2}.$$

We combine the word overlap score with the average Jaccard coefficient to obtain our ranking score:

$$score(c) = avgJ(c,Ex)\cdot wo(W(c),W(E,R,T))$$

run	rel_ret	map	Rprec
ilpsSetOL	38/81	0.1063	0.0973
ilpsSetOLnar	**43/81**	**0.1152**	**0.0899**

Table 3: Results of the entity list completion runs.

4.3 Results

Table **??** shows the results of our two ELC runs. There is only a small difference in performance between the runs. This suggests that the links of the example entities already cover the relation expressed by the topic and that word overlap of the candidate entities with the topic relation does not add additional discriminative information on top of that. We note that our approach finds the most relevant entities, which is about half of the total number of relevant entities. One of the limitations is that we only consider entities as candidates when they link to the same objects as the example entities. In future work we plan to loosen this constraint by also considering entities that are relevant to the terms in the relation without any links to the example entities.

5 Relevance Feedback Track

Typical relevance feedback algorithms consider feedback documents as generative models from which to sample terms. We find that simply applying out-of-the-box relevance feedback algorithms to the single example document is not effective; such feedback algorithms degrade retrieval performance. To address this issue, we have implemented a novel model and our focus in our TREC participation this year is to evaluate its performance.

No results have been provided to the participants at the time of writing. Because of this, we limit our discussion to describing our algorithm.

Our algorithm makes use of the moderated contents of Wikipedia as a pivot language. Wikipedia articles can be created by anyone, but they are typically moderated by a relatively small group of volunteers. Moreover, Wikipedia has extensive guidelines in place, pertaining to the correct use of grammar and style. As a consequence (and unlike common web pages), the language used in each article tends to be "clean" and to the point. It is this particular feature of Wikipedia that we use to influence the estimation of the language model of web pages. The expanded query language model is interpolated with the initial query to obtain a final representation of the user's information need.

6 Conclusion

We have described the participation of the University of Amsterdam's ILPS group in the session, entity, and relevance feedback track at TREC 2010. We arrived at the following preliminary conclusions. For the Session Track we find that using blind relevance feedback to extend a follow-up query with terms extracted from original results gives mixed results. Future work will be needed to select high-quality expansion terms more consistently, and to address differences between different (types of) queries.

For the Entity Track REF task we find that using a smaller window size for the contexts considered by our co-occurrence model leads to a small increase in performance. The inclusion of all context in which a candidate and source entity co-occur in our model leads to worse results when using a larger (more noisy) corpus. We also explore the use of Freebase for type filtering, homepage finding and entity normalization. Of which only the latter proved unsuccessful due to abbreviations identical to common terms distorting the co-occurrence counts and context models. For the ELC task we find that ranking candidate entities based on the links shared with the example entities is effective, especially in terms of recall.

No results are available for the Relevance Feedback Track at the time of writing and consequently we can provide no analyses or conclusions about our participation in the track.

7 Acknowledgments

This research was partially supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430, by the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191, by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project nr STE-09-12, by the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.066.512, 612.061.814, 612.-061.815, 640.004.802, 380-70-011, and by the Center for Creation, Content and Technology (CCCT).