Actively Learning to Rank Semantic Associations for Personalized Contextual Exploration of Knowledge Graphs

Federico Bianchi(✉), Matteo Palmonari, Marco Cremaschi, and Elisabetta Fersini

University of Milan - Bicocca, Viale Sarca 336, Milan, Italy
{federico.bianchi,palmonari,cremaschi,fersiniel}@disco.unimib.it

Abstract. Knowledge Graphs (KG) represent a large amount of Semantic Associations (SAs), i.e., chains of relations that may reveal interesting and unknown connections between different types of entities. Applications for the contextual exploration of KGs help users explore information extracted from a KG, including SAs, while they are reading an input text. Because of the large number of SAs that can be extracted from a text, a first challenge in these applications is to effectively determine which SAs are most interesting to the users, defining a suitable ranking function over SAs. However, since different users may have different interests, an additional challenge is to personalize this ranking function to match individual users' preferences. In this paper we introduce a novel active learning to rank model to let a user rate small samples of SAs, which are used to iteratively learn a personalized ranking function. Experiments conducted with two data sets show that the approach is able to improve the quality of the ranking function with a limited number of user interactions.

1 Introduction

Knowledge Graphs (KG) represent entities of different types, their properties and binary relations that interconnect these entities. KGs are today frequently used to support interoperability among applications also in the industry, while languages like RDF support the publication of KGs as open linked data. A problem that has recently gained attention is how to exploit the vast amount of knowledge available in proprietary or open KGs to deliver useful information to the users. While query answering is aimed at satisfying specific information needs, knowledge exploration provides mechanisms to deliver information that is estimated to be interesting for the users in a proactive fashion [1].

One approach to support knowledge exploration is to push content from KGs while users are carrying out familiar tasks, such as querying a search engine, watching media content [2], or reading a text of interest [3,4]. We refer to the latter approaches as *contextual KG exploration*, where an input text (possibly a description of media content) is used as a entry point to let users explore information extracted from the KG. By using well-known entity linking techniques [3],

E. Blomqvist et al. (Eds.): ESWC 2017, Part I, LNCS 10249, pp. 120–135, 2017.
DOI: 10.1007/978-3-319-58068-5_8

entities mentioned in the text are linked to a KG. If more than one entity is found, semi-walks in the KG that connect the two entities, i.e., **Semantic Associations** (SAs) of finite length between the two entities [4,5] reveal connections between entities, which may provide new and interesting insights into the topic of the input text. For example, a SA found in DBpedia revealing that Clinton and Trump have been both members of the Democratic Party has been found interesting by many Italian students who read about US Election 2016.

The main problem arising in contextual KG exploration is that a very large amount of SAs can be found between a set of entities extracted from even relatively short texts. For example, from an input article[1], as many as 40.107 SAs are found in DBpedia with DaCENA[2], a data journalism prototype for contextual KG exploration [4]. The crucial research challenge to exploit such a large amount of SAs represented in KGs is to provide effective methods to identify those few SAs that are more interesting for the users. Several approaches have been proposed that use measures based on graph analytics to rank and filter SAs [5,6]. However, *different users may be interested in different kinds of SAs*, which suggests that the ranking function should adapt to the preferences of individual users. One approach proposes to personalize a ranking function by learning from explicit user preferences [7], but does not address the problem of minimizing the labels collected from the users, which is crucial when exploring very large sets of SAs. Starting from these observations in this paper we address the following research questions: (RQ1) Can we learn to rank SAs by iteratively collecting a small number of labels from a user, so that we can personalize the content delivered to her based on her preferences? (RQ2) Do we need personalization in contextual exploration of KGs with SAs, or can we assume that different users are interested in the same content?

To answer to Q1, we propose an active learning to rank model to reduce the number of SAs that need to be labeled by the user. The model comprises: (1) a workflow to iteratively collect labels from a user and learn to rank the SAs based on her preferences using the RankSVM algorithm; (2) algorithms to actively select the SAs that the user has to label; (3) different approaches to select the first set of SAs that the user has to label, thus solving a cold start problem affecting the above mentioned active sampling algorithms, (4) a set of features based on KG analytics to represent SAs and support the model. To evaluate the effectiveness of the proposed model under different configurations and against different baselines, we have built two data sets consisting of ratings given by different users on a complete set of SAs extracted for different pieces of news articles. Results show that the proposed approach is feasible and provides a consistent improvement of the ranking quality with a limited number of interactions. To answer to Q2, we measure the agreement among ratings given by different users to SAs found for the same articles. Results clearly show that different users are interested in different content, thus confirming the need for personalization methods in contextual KG exploration.

[1] https://goo.gl/RFvqZh.
[2] http://www.dacena.org/article/84.

To the best of our knowledge this is the first attempt to use active sampling to learn to rank SAs, thus improving on state of the art approaches that require a large number of labels to learn a ranking function over SAs. The paper is organized as follows: in Sect. 2, we further motivate the proposed approach by discussing contextual KG exploration, with an example of application; in Sect. 3, we explain our active learning to rank model; in Sect. 4 we describe the experiments conducted to evaluate our model; in Sect. 5 we discuss related work, while in Sect. 6 we draw some conclusions and discuss future work.

2 Contextual KG Exploration

Applications for the contextual exploration of KGs enrich the experience of a user who is accessing textual or multimedia content by delivering information extracted from a KG [2–4]. The input content tell us something about current interests of the user, thus providing a starting point to select pieces of valuable information, because helpful to expand her knowledge or to better understand the content itself. Named Entity Recognition and Linking (NEEL) techniques [3] can be applied to an input text to extract a first a set of entities from the KG, which can be subsequently used to retrieve additional information, e.g., SAs. In the following, we refer to SAs as *semi-walks in a KG* [5]*, which do not contain loops and whose nodes are entities*. We discuss how SAs can be used as source of additional information in these application by referring to DaCENA [4] (Data Context for News Articles), an application that supports exploration of KGs to address one of the missions of data journalism, i.e., "*[...] to provide context, clarity and, [...] find truth in the expanding amount of digital content*" [8]

DaCENA presents a set of SAs extracted from a KGs as additional information (a data context) to a user who is reading a news article. End users can read the article and explore the extracted SAs from an interactive interface. Figure 1 shows a screenshot of the interface, and readers can play with the same example online[3]. The graph shows the k-most interesting SAs (40.107 for this article), where k can be set by the user. When the user clicks on an entity node, e.g., *Separatism*, SAs from/to such node are shown in the lower panel and ordered by estimated interest. SAs are extracted with a process controlled through a back-end. In DaCENA we are currently using TextRazor[4] as NEEL tool and DBpedia[5] as reference KG. Once entities are extracted we make several queries to the DBpedia SPARQL endpoint to extract the SAs that connect all these entities (we use a SPARQL endpoint to ensure that we retrieve up-to-date information). While we started with extracting SAs of maximum length equal to three [4], now we consider only SAs of maximum length equal to two, because we found - with preliminary user studies - that SAs of length greater than two are seldom considered interesting by the users. Otherwise, while we previously found every SA from a principal entity to every other entity, we now consider

[3] http://www.dacena.org/article/84.
[4] https://www.textrazor.com/.
[5] https://dbpedia.org/.

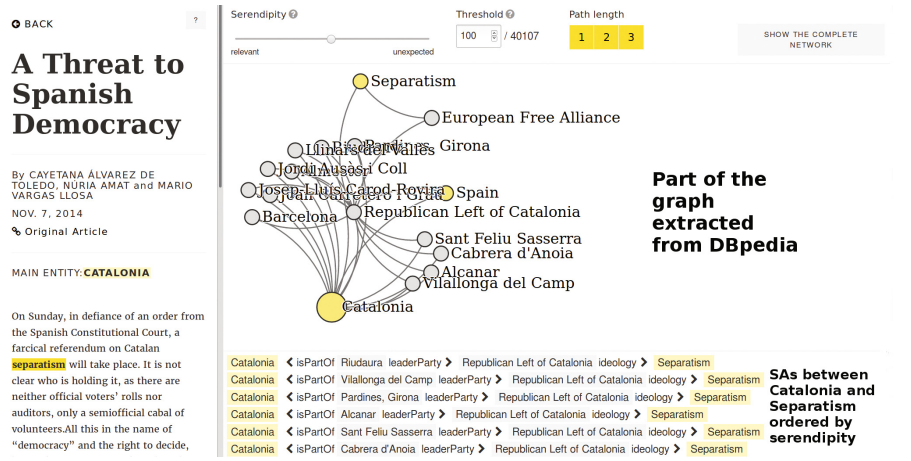

Fig. 1. DaCENA interface

shorter SAs between every entity extracted from the text. Processing an article may require significant amount of time (up to thirty minutes) if semantic data are fetched by querying a SPARQL endpoint as we currently do. Therefore, texts and data are processed off-line so as to make the interactive visualization features as much fluid as possible. DaCENA currently uses a measure to evaluate *interestingness* of SAs named **Serendipity** [4]. Serendipity is defined as a parametric linear combination of **Relevance**, a measure that evaluates the relevance of a SA with respect to a text, and **Rarity**, a measure that evaluates how much a SA may be unexpected for the users. A SA is relevant if the virtual document built by concatenating the abstracts of each entity occurring in the SA is similar to the given text (we compute the cosine similarity between word vectors weighted using TF-IDF). Instead, a SA is unexpected, or rare, when it is composed by properties that are not frequently used in the KG (see [4] for the formula). Let α be a parameter used for balancing the weight of each measure, and *text* be the input text; the serendipity $S(\pi)$ of an SA π, is computed by $S(\pi, text) = \alpha relevance(\pi, text) + (1 - \alpha)\ rarity(\pi)$. In the interface shown in Fig. 1 the user can adjust the serendipity parameter to favor relatedness or unexpectedness.

By analyzing several articles with DaCENA, we could observe that thousands or even dozens of thousands of SAs can be extracted from an article (see, e.g., the example shown in Fig. 1), while preliminary user studies suggest that users do not want to look at more than 100 SAs. Thus, the ranking function used to push the most interesting SAs upfront and filter out other SAs is crucial to help users effectively explore the KG content. Moreover, while some user may be interested in finding out information about small municipalities associated with separatism, other users may be more interested in information about more important cities. If different users have different interests (an hypothesis validated in our experiments), mechanisms to personalize KG exploration are needed.

3 Active Learning to Rank for Semantic Associations

The Active Learning to Rank (ALR) model proposed to personalize the exploration of KG is based on a learning loop: at each iteration, ratings given by the user to few SAs are used to update the ranking over the whole set of SAs. The workflow implemented by our model is described in Fig. 2 and explained with an example depicted in Fig. 3. The example is taken from a run of the best performing configuration of the ALR model (according to experiments discussed in Sect. 4) and shows ratings given by one user to few associations (red circles) as well as the 3 most interesting SAs according to the learned ranking (blue clouds represent the ratings eventually given by the user after rating all SAs). The entry point (Step 1) is a *bootstrapping* phase where we select the first SAs that the user has to label. The user labels the SAs selected in the bootstrapping step (Step 2) using ratings in a graded scale, e.g., $<1, 2, 3, 4, 5, 6>$, where higher grades represent higher interest for an SA. Then we use these labels to train a learning to rank algorithm (step 3), which ranks all the SAs by assigning them a score. If the user decides that she is satisfied with the ranking obtained so far, the loop stops. Else, we proceed to further improve the ranking by collecting more labels using active sampling (Step 5). In active sampling, observations are selected with the aim of optimizing the ranking function with as few labels as possible. To find the observations for which labels are estimated to be more informative, active sampling algorithms use the scores determined by the learned ranking function. This prerequisite, motivates the need for introducing a bootstrapping step (Step 1) where labels are not selected using active sampling. After Step 5, we close the loop by repeating Step 2. Observe that after the first iteration, the user always labels SAs selected with active sampling. In Fig. 3, it can be noticed that the ranking improves after the second iteration (Step 3). The main steps of the loop, i.e., steps 1, 3 and 5, are explained in details here below.

Step 1: Bootstrap Active Learning. Two approaches are proposed: the first one (alternative 1a) is based on clustering algorithms while the second one uses an heuristic ranking function (alternative 1b). The latter has the advantage that

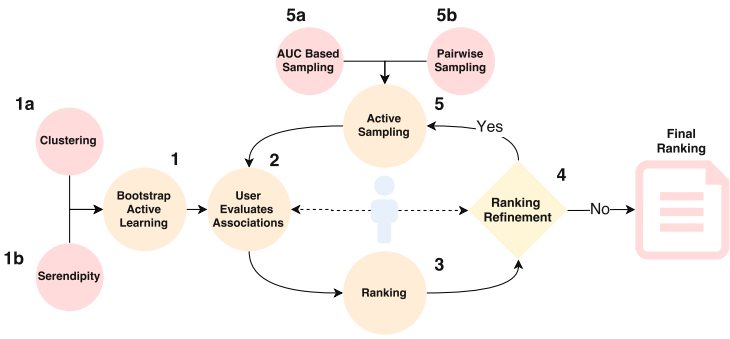

Fig. 2. Workflow of the active learning to rank model

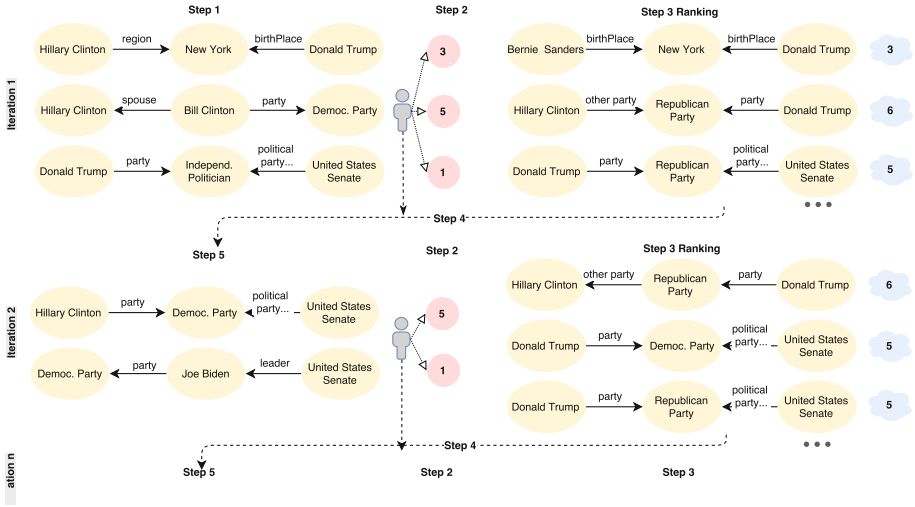

Fig. 3. Example of iterative ranking refinement with the ALR model (Color figure online)

we have an ordered set of SAs (hence, a small set of interesting SAs) to present to the user even before she provides any label.

Alternative 1a. The assumption at the basis of this approach is the following: observations that would be rated in a similar way by a user are spatially near in the feature space used to represent the SAs, while observations rated in different ways should be distant in this space. Based on this assumption, the best way to quickly collect the training data is to cluster the data set and take the most representative observation for each cluster (the observation nearest to the mean of the cluster). Approaches similar to this have been already considered in active learning settings [9,10], in which clustering is used to find the first observations for machine learning models. We test two clustering algorithms: the first one is the Dirichlet Process Gaussian Mixture Model [11] (Dirichlet) that has been chosen for its ability to automatically find the best number of cluster inside the data set; this is useful because we cannot know a-priori which could be the correct number of cluster for a given set of SAs. We also evaluate a second clustering algorithm, the Gaussian Mixture Model (Gaussian) [12]. It is important to notice that these algorithms select SAs that are representative of a data set, but that does not mean that these SAs are also meaningful or interesting for a user.

Alternative 1b. Another approach that we propose to use for bootstrapping the model (also in consideration of the latter remark) is to use an heuristic ranking measure. In this way, not only we can show to a user a set of SAs even before she provides any rating, but we can also ask their ratings on a

set of SAs that are heuristically believed to be interesting. In the context of contextual KG exploration, this may be desirable to improve the user experience, when compared to asking ratings on a set of uninteresting SAs. In particular, as heuristic ranking function, we use the *serendipity* measure defined in Sect. 2.

Step 3: Ranking. In this phase we train a learning to rank algorithm that can help us ordering SAs. In our approach we use the RankSVM [13] algorithm, where the ranking problem is transformed into a pair-wise classification problem [14]. RankSVM is a variant of SVM created for learning to rank tasks, it takes ratings (based on a graded scale) over a set of the domain items as input and use these ratings to infer labels for a set of item pairs. An item pair is assigned a label equal to 1, if the first item of the pair should be ranked higher than the second one, and equal to -1, otherwise. This is the binary input of the inner algorithm used to learn the ranking function. This function assigns a score (a real number) to each item by generalizing the binary input.

Step 5: Active Sampling. We implemented two supervised active sampling algorithms proposed in the document retrieval field. Both the algorithms use a pairwise approach, meaning that they can be directly used on pairwise learning to rank algorithms, like RankSVM.

Alternative 5a: the first algorithm [15] (denominated *AUC-Based Sampling*, or, shortly, *AS*, in the next sections) tries to optimize the Area Under the Curve (AUC), by selecting individual observations for labeling, without explicitly comparing every pair of domain items. The algorithm is thus known to be sub-optimal, runs efficiently. It is essentially based on the computation of the estimated probability of a binary class for an observation (thus, it was used in a binary setting). The algorithm uses a parameter λ to calibrate the weight of two different probability estimations.

Alternative 5b: the second algorithm [16] (denominated *Pairwise Sampling*, or, shortly, *PS*, in the next sections) explicitly compares pairs of SAs to select the most informative pairs. The most informative pairs are the ones that maximize two measures: Local Uncertainty (LU), which estimates the uncertainty of the relative order within the pair, and Global Uncertainty (GU), which estimates the uncertainty of the position of each element of the pair within the global ranking. A parameter p is used to tune the weight assigned to the LU measure. In this case, users are then asked to rate each of the most informative pairs of SAs. With this approach, we can evaluate if the uncertainty score used to select the pair is incoherent with user ratings, thus providing more informative labels to RankSVM. The explicit generation of the observation-pairs makes this algorithms less efficient than AUC-Based Sampling, which may prevent its application to the exploration of a large number of SAs.

3.1 Features

To represent the SAs inside our platform we used different measures. In this way, we define feature vectors for the active learning to rank algorithms. We normalize

data extracted with these measures using a standard normalization techniques by removing the mean and scaling to unit variance.

Global PageRank. We use the data in [17] to collect a global score of the PageRank inside DBpedia. In this, way we are able to get an overall value of the importance of an entity inside the KG. The Global Pagerank of a SA is computed as the average global pagerank of every entity occurring in that SA.

Local PageRank. We compute PageRank on the sub-graph, defined by the SAs extracted from an input text, to measure the *centrality importance* of each entity.

Local HITS. We ran the HITS (Hyperlink-Induced Topic Search) [18] algorithm to compute two scores for each node of the local graph. Authority score, indicates how much a node is *important*, while hub score indicates nodes that point to nodes with a high authority score. The algorithm gives two scores for each SA: one for the average of the authority values and one for the average of the hub values.

Temporal Relevance. Using the Wikimedia API we extract the number of times a Wikipedia entity (page) as been accessed in a specific date (date of the publication of a given text, for example). In this way we are able to measure the value of importance related to timing. For example, if we consider Wikipedia access[6] on the page Paris, we see that the entity has been accessed 8.331 times on 12-11-2015 and 171.988 times on 14-11-2015, when on 13-11-2015 there have been terrorist attacks in Paris. The temporal relevance for a SA is given by the average temporal relevance of all the entities in the SA.

Path Informativeness. We use a measure defined in [5], which is based on the concept of Predicate Frequency Inverse Triple Frequency (PF-ITF).

Path Pattern Informativeness We use a measure on path patterns, defined in [5], to get the informativeness of patterns extracted from paths.

Relevance and Rarity. We use the two measures explained in Sect. 2 as components of our Serendipity measure [4].

4 Experiments

The purpose of the experimental evaluation is to validate the hypothesis that personalization is important in KG exploration, to evaluate the performance of the proposed model, and to compare alternative approaches proposed for different steps of the model. The targets of the application are fairly educated users familiar with IT technologies. So far, we used in our experiments master students from Computer Science, Mechanical Engineering and Communication Sciences with good English reading skills. All the data sets used in our expriments are available online[7]. The experiments were run on a machine with a Intel Core i5 (4th Gen, 1.6 Ghz).

[6] http://tools.wmflabs.org/pageviews/.
[7] https://github.com/vinid/semantic-associations-survey.

4.1 Experimental Settings

To test our model we built two different datasets, each one consisting of triples $<text_i, A_i, ratings_{u,i}>$, where $text_i$ is a text extracted from an article retrieved from online news platforms like NYT and The Guardian, A_i is the set of all SAs extracted from $text_i$ with our tool DaCENA, and $ratings_{u,i}$ contains the labels assigned by a user u to every SA in A_i. From each triple in a dataset, we can derive a complete ranking of the retrieved SAs for one user, i.e., a personal ideal ranking. We describe the creation of each dataset here below.

Short Articles Many Users (SAMU). We collected user ratings for this data set using an online form. For ratings we choose a graded scale from 1 to 6, following guidelines suggested in a recent study [19]. Differently from a five-valued ordinal scale, this scale provides a symmetric range that clusters scores in two sets: scores with a negative tendency (1, 2 and 3) and scores with a positive tendency (4, 5 and 6). Each user had to evaluate the complete set of associations extracted from one article, thus we had to choose articles small enough to let users perform their task without being subject to fatigue bias [19]. We thus selected the first self-contained paragraphs of articles from NYT and Guardian with the following features: articles topic concern politics, is reasonably well-known and engaging for foreign (Italian) educated users; the number of associations extracted by DaCENA is comprised between 50 and 100 SAs. The average task completion time resulted in 12 min - little below the fatigue bias threshold mentioned in [19]. We also wanted to have preferences of different users on a same article to measure inter-user agreement and validate the "personalization hypothesis": we needed a number of articles small enough to collect at least 3 evaluations from different users. Articles were assigned randomly to each user to avoid any bias. After evaluating the first article, a user could stop or evaluate more articles. We stopped searching users for the evaluation when we collected evaluations by at least 3 users on each article, which resulted in a total of 14 different users, and 25 gold standards (personal rankings).

Long Articles Few Users (LAFU). We wanted also to evaluate if results obtained over small SA sets are comparable with results obtained with (and thus generalizable to) large SA sets. To this end, two users were asked to rate thousands of SAs extracted for two full-length articles, with the goal of evaluating heuristic functions used in an early version of DaCENA. In this case, we used a three-valued scale for ratings, from 1 to 3. The two users involved in the evaluation of the longer articles were Communication Sciences students with no background in Computer Science. They were granted several days for completing the task, and asked to complement their task with a qualitative analysis.

Using the ideal rankings in the two gold standards, we measure the quality of the rankings returned by our model at different iterations using Normalized Discounted Cumulative Gain (nDCG) computed over the top-10 ranked SAs, denoted by nDCG@10. In addition, we compute the Area Under the nDCG@10 Curve (AUNC) as an aggregate performance measure, the curve is based on the nDCG@10 values at each iteration. We carry out experiments in two

different settings. In *Contextual Exploration Settings*, we consider the workflow as implemented in a system that supports contextual exploration: the set from which we select the observations to label is the same set used to evaluate the performance of the model. In these settings, we make sure that observations labeled during previous iterations are not labeled a second time by the user. In *Cross Validation Settings*, which was used also in previous work [15], active sampling always picks SAs from the training set. Although not amenable in contextual exploration, this approach is helpful to evaluate the robustness of the model. In fact, we can use 2Fold-Stratified Cross Validation to make sure that results can be reasonably generalized and do not depend on specific data.

4.2 Configurations and Baselines

We evaluate different configurations of the model, based on the alternative algorithms proposed in the two steps of the loop. Direct comparison with other state-of-the-art approaches is difficult because we could not find an active learning to rank approach for SAs. For Bootstrap Active Learning, we consider three approaches: two clustering algorithms (Gaussian vs. Dirichlet), and the Serendipity heuristic function, for which we set $\alpha = 0,5$. For Active Sampling, we consider two algorithms: AUC Based Sampling [15] (AS) and Pairwise Sampling [16] (PS). Parameters of these two algorithms have been determined experimentally, and set to $\lambda = 0.8$ and $p = 1$. The six configurations of the active learning to rank workflow described above are compared also against three different baselines:

- *Random + Random*: RankSVM is still used to learn a ranking function, but is trained using ratings assigned to SAs that are randomly selected, both in the bootstrap and active sampling steps.
- *Serendipity No-AL*: we consider the ranking determined with Serendipity, which is not based on active learning and does not change across iterations.
- *Random No-AL*: we consider random rankings of SAs, which are not based on active learning and do change across iterations.

Random algorithm are run multiple times to stabilize values (100 hundred thousands of them).

Configuration Details. In the SAMU data sets Dirichlet and Gaussian Clustering, in the first iterations, selected an average number of clusters equal to 3 (and thus, an average number of 3 SAs are selected from this two methods in the first iteration); for this reason, to feed the model with a balanced number of observations, on the average, for both Serendipity and Random Active Learning we choose to select 3 SAs when using Serendipity and Random in the bootstrapping step. Finally for this data set we select 2 SAs to be evaluated at each active sampling step. The number of observations collected at each iteration was increased in the LAFU data set. The clustering algorithms in this data set selected an average number of cluster equal to 5, leading to 5 SAs to be labeled when using Serendipity and Random for bootstrapping. In the active sampling phase, for the LAFU data set, we selected 6 observation to be labeled (since we

have more data) at each iteration. In this data set we could not use the *Temporal Relevance* as a feature because articles in this data set are not recent. We used a RankSVM with polynomial kernel on the LAFU data set that was able to output the results of a single iteration in what we considered interactive time (less then 2 s).

4.3 Results and Discussion

User interests and personalization. We have measured Inter-Rater Reliability [20] (IRR) to assess the usefulness of personalization within this context. Our idea is based on the assumption that different users are interested in different things. IRR was computed on the SAMU, which had the same SAs rated by different users. We used two measures: Krippendorff's alpha, which gave in output a value of 0.06154, weighted using an ordinal matrix, and Kendall's W, which gave a score of 0.2608. We can see that for all the five texts used in this experiment, IRR is low and distant from 1, the value that usually represents unanimity between the raters. We also show the distribution of the ordinal ratings for the data sets in Table 1.

Table 1. Rating distribution in the two data sets

Rating	1	2	3	4	5	6
SAMU	23.7%	14.5%	22.3%	20.9%	10.1%	5.5%
LAFU	67.4%	30.1%	2.5%	NaN	NaN	NaN

Contextual Exploration Settings. In this setting the active learning algorithms Fig. 4 were able to perform better than the baseline considered (we can notice that active learning approaches completely outperform the non active learning ones). AUNC values can be seen in Table 3, the algorithm that performs better is the one that uses serendipity for the bootstrap step and AS for the active sampling step (Fig. 2).

Cross Validation Settings. The results we obtained in the cross validation settings provide further evidence that the best result is obtained by the Serendipity heuristic, combined with the use of AS, However, in this setting we achieve a worse performance, due to the fact that the active learning algorithms are not able to access to the test data. The plots can be found in Fig. 4 while the computed areas are in Table 3.

Bootstrap Active Learning Analysis. To construct a ranking model we need at least two examples with different labels, is not always possible in the first iteration (the user, in that step, can assign to each observation the same degree of interest). We evaluated the *time for first iteration* value, that corresponds to the average number of iterations needed for each method to have a training useful

Table 2. AUNC in cross validation

Configurations	SAMU	LAFU
Gaussian AS	3.0168	2.6455
Dirichlet AS	3.0011	2.6872
Gaussian PS	2.9975	NaN
Dirichlet PS	3.0009	NaN
Serendipity AS	**3.0742**	**2.711**
Serendipity PS	3.0302	NaN
Random Random	2.976	2.6013

Table 3. AUNC in contextual exploration

Configurations	SAMU	LAFU
Gaussian AS	3.0242	2.747
Dirichlet AS	3.0711	2.7174
Gaussian PS	2.9629	NaN
Dirichlet PS	3.019	NaN
Serendipity AS	**3.2018**	**3.0817**
Serendipity PS	3.1399	NaN
Random Random	2.9359	2.673
Serendipity No-AL	2.7199	2.734
Random No-AL	2.3199	1.7971

for training the learning to rank algorithm. The worst case of these algorithms is represented by a user who gives the same score to every observation. The results are visible in Table 4. We show data for both Cross Validation (CV) and Contextual Exploration Setting (CE). We can see that with the LAFU data set finding the first observations needed to train the model becomes more difficult for methods with the except of Dirichlet that can probably adapt itself to the dataset in an easier way.

Table 4. #Iterations to find the first training set for the ranking model

Algorithm	#Iter. SAMU CV	#Iter. LAFU CV	#Iter. SAMU CE	#Iter. LAFU CE
Dirichlet	1.16	1.11	1.010	1.063
Gaussian	1.08	1.16	1.066	1.381
Serendipity	1.08	1.5	1.346	1.1363
Random	1.25	1.5	1.1866	1.229

Discussion. Based on the evidence collected through the above experiments, we can provide answers to research questions introduced in Sect. 1. In relation to RQ2, we observed that different users have different interests, which motivates the need for personalization in KG exploration approaches. In relation to RQ1, the ALR model introduced in this paper supports shows remarkable improvement over ranking methods that do not use active sampling. In this context and for the selected set of features, we found that AUC-based sampling performs better than pairwise sampling, both in terms of effectiveness and efficiency, and that the combination of Serendipity (for bootstrapping) and AUC-based active sampling outperforms every alternative configurations on both data sets.

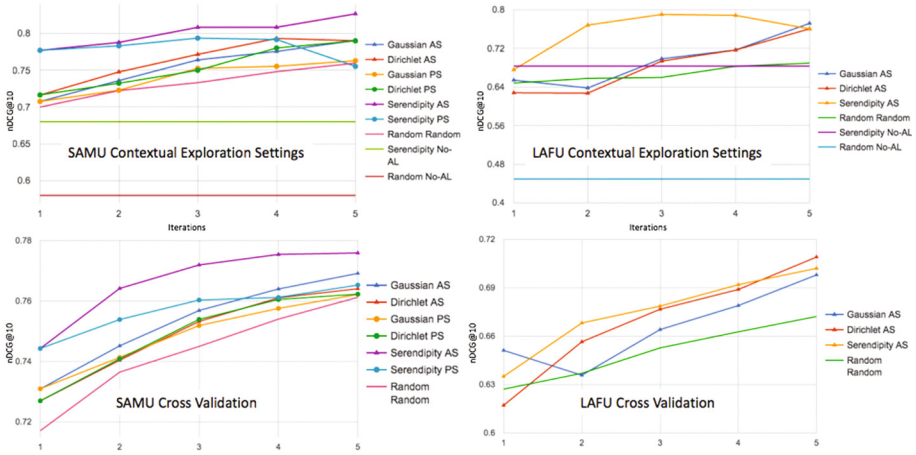

Fig. 4. nDCG@10 in contextual exploration and cross validation settings

5 Related Work

We compare our work to previous work in the field of interactive KG exploration and of learning to rank approaches for KG exploration.

Interactive Knowledge Graph Exploration. Several methods, described and compared in a recent survey [1], combine navigation, filtering, sampling and visualization to let users explore large data sets. One approach to entity expansion provides an example of contextual KG exploration, but does not focus on the retrieval of SAs like our approach [2]. RelFinder is a web application that finds SAs between two entities selected by a user [21]. Other applications similar to RelFinder also incorporate measures to evaluate and explain SAs [5,6,22]. Refer [3] is a Wordpress Plugin that help a user enrich an article with additional information extracted from KBs like Wikipedia. The plugin finds entities in the article and recommends SAs that are estimated to by unknown to the user. Refer is an example of contextual exploration of KG; the main difference between their approach and our approach is that we introduce a model to order all SAs, introducing a machine learning model to personalize the exploration. None of the approaches mentioned above or surveyed in [1] introduces methods to learn information to show to the users based on their explicit feedback. An interesting approach seen in the literature [23] uses genetic programming to find strong relationships in linked data; in their experiments, eight judges were asked to evaluate the relationships, but relationships with low inter-user agreement were not considered positive examples for training because not interesting for all users. Since different users have different interests, we train our model based on the preferences of individual users using ALR.

Learning to Rank and Active Learning for KG Exploration. Learning to rank has been extensively applied in document retrieval [14] but only in one

approach to KG exploration [7]. This approach use a variant of SVM to rank SAs extracted from Freebase, but does not try to minimize the inputs needed to learn the ranking function through active learning. In addition, some of their features are specifically tailored on the Freebase structure while our features can be easily applied to any KG (with the exception of Temporal Relevance, which requires bridges from the KG to Wikipedia). Active learning to rank introduces techniques to select the most informative observations to train the model. In our approach, we have implemented and tested two different techniques proposed for document retrieval. A first approach collects labels over individual observations (SAs in our case) and solves the cold-start problem by randomly selecting positive and negative instances from a subset of the data reserved for training [15]. In our interactive approach we pick the SAs that are labeled by the user form the same set that has to be ranked, which is coherent with contextual KG exploration scenarios. However, we have also conducted tests with data split in a training and a test set to show the robustness of the model. In addition, we provided a principled approach to solve the cold-start problem in our domain. The second approach, which collects labels over pairs of observations [16], seems to be not only less efficient, but also less effective for ranking SAs. To the best of our knowledge, ours is the first attempt to apply active learning to rank to the problem of exploring SAs.

6 Conclusion

Experimental results presented in this paper suggest that active learning approaches can be effectively used to optimize the ranking of SAs extracted from KGs, thus supporting personalized exploration of complex relational knowledge made available in these graphs. Such personalization mechanisms have also shown to be important for knowledge exploration, since different users are interested in different content. We have also found that, an approach that combines our Serendipity measure [4] and AUC-based active sampling outperforms different alternative configurations. In future work, we plan to analyze the impact of individual features on the performance of an active learning to rank model for SAs, and evaluate the use of additional measures. In addition, we want to incorporate our active learning to rank model into the DaCENA application, by tackling the challenge of designing human-data interaction patterns that can engage the users. We would also like, in the future, to improve the performance of our application. So far, we preferred to have fresher information via a SPARQL endpoint despite the longer processing time, because processing is performed offline. In journalism, freshness of information is relevant and we plan to further investigate methods to refresh/update SAs after processing in the future.

Acknowledgement. We thank our colleague Federico Cabitza for his knowledgeable advises about the creation of the SAMU data set.

References

1. Bikakis, N., Sellis, T.: Exploration and visualization in the web of big linked data: a survey of the state of the art. preprint arXiv:1601.08059 (2016)
2. Redondo-García, J.L., Hildebrand, M., Romero, L.P., Troncy, R.: Augmenting TV newscasts via entity expansion. In: Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8798, pp. 472–476. Springer, Cham (2014). doi:10.1007/978-3-319-11955-7_69
3. Tietz, T., Jäger, J., Waitelonis, J., Sack, H.: Semantic annotation and information visualization for blogposts with Refer. In: VOILA 2016, vol. 1704, pp. 28–40 (2016)
4. Palmonari, M., Uboldi, G., Cremaschi, M., Ciminieri, D., Bianchi, F.: DaCENA: serendipitous news reading with data contexts. In: Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9341, pp. 133–137. Springer, Cham (2015). doi:10.1007/978-3-319-25639-9_26
5. Pirrò, G.: Explaining and suggesting relatedness in knowledge graphs. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9366, pp. 622–639. Springer, Cham (2015). doi:10.1007/978-3-319-25007-6_36
6. Cheng, G., Zhang, Y., Qu, Y.: Explass: exploring associations between entities via Top-K ontological patterns and facets. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8797, pp. 422–437. Springer, Cham (2014). doi:10.1007/978-3-319-11915-1_27
7. Chen, N., Prasanna, V.K.: Learning to rank complex semantic relationships. IJSWIS **8**(4), 1–19 (2012)
8. Gray, J., Chambers, L., Bounegru, L.: The Data Journalism Handbook. O'Reilly Media Inc., Sebastopol (2012)
9. Kang, J., Ryu, K.R., Kwon, H.-C.: Using cluster-based sampling to select initial training set for active learning in text classification. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 384–388. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24775-3_46
10. Yuan, W., Han, Y., Guan, D., Lee, S., Lee, Y.K.: Initial training data selection for active learning. In: ICUIMC, p. 5. ACM (2011)
11. Gershman, S.J., Blei, D.M.: A tutorial on bayesian nonparametric models. J. Math. Psychol. **56**(1), 1–12 (2012)
12. Tan, P.N., et al.: Introduction to Data Mining. Pearson Education India, Upper Saddle River (2006)
13. Lee, C.-P., Lin, C.-J.: Large-scale linear ranksvm. Neural Comput. **26**(4), 781–817 (2014)
14. Liu, T.-Y.: Learning to rank for information retrieval. Found. Trends Inf. Retr. **3**(3), 225–331 (2009)
15. Donmez, P., Carbonell, J.G.: Active sampling for rank learning via optimizing the area under the ROC curve. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 78–89. Springer, Heidelberg (2009). doi:10.1007/978-3-642-00958-7_10
16. Qian, B., Li, H., Wang, J., Wang, X., Davidson, I.: Active learning to rank using pairwise supervision. In: SIAM International Conference Data Mining, pp. 297–305. SIAM (2013)
17. Thalhammer, A., Rettinger, A.: PageRank on Wikipedia: towards general importance scores for entities. In: Sack, H., Rizzo, G., Steinmetz, N., Mladenić, D., Auer, S., Lange, C. (eds.) ESWC 2016. LNCS, vol. 9989, pp. 227–240. Springer, Cham (2016). doi:10.1007/978-3-319-47602-5_41

18. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. JACM **46**(5), 604–632 (1999)
19. Cabitza, F., Locoro, A.: Questionnaires in the design and evaluation of community-oriented technologies. Int. J. Web-Based Commun. **13**(1), 4–35 (2017)
20. Gwet, K.L.: Handbook of inter-rater reliability: the definitive guide to measuring the extent of agreement among raters. Advanced Analytics, LLC (2014)
21. Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., Stegemann, T.: RelFinder: revealing relationships in RDF knowledge bases. In: Chua, T.-S., Kompatsiaris, Y., Mérialdo, B., Haas, W., Thallinger, G., Bailer, W. (eds.) SAMT 2009. LNCS, vol. 5887, pp. 182–187. Springer, Heidelberg (2009). doi:10.1007/978-3-642-10543-2_21
22. Fang, L., Sarma, A.D., Yu, C., Bohannon, P.: Rex: explaining relationships between entity pairs. Proc. VLDB **5**(3), 241–252 (2011)
23. Tiddi, I., d'Aquin, M., Motta, E.: Learning to assess linked data relationships using genetic programming. In: Groth, P., Simperl, E., Gray, A., Sabou, M., Krötzsch, M., Lecue, F., Flöck, F., Gil, Y. (eds.) ISWC 2016. LNCS, vol. 9981, pp. 581–597. Springer, Cham (2016). doi:10.1007/978-3-319-46523-4_35