Combining Serendipity and Active Learning for Personalized Contextual Exploration of Knowledge Graphs

Federico Bianchi [a], Matteo Palmonari [a] Marco Cremaschi [a] and Elisabetta Fersini [a]
[a] *University of Milan - Bicocca, Viale Sarca 336, Milan, Italy*
federico.bianchi@disco.unimib.it,
palmonari@disco.unimib.it,
cremaschi@disco.unimib.it,
fersiniel@disco.unimib.it

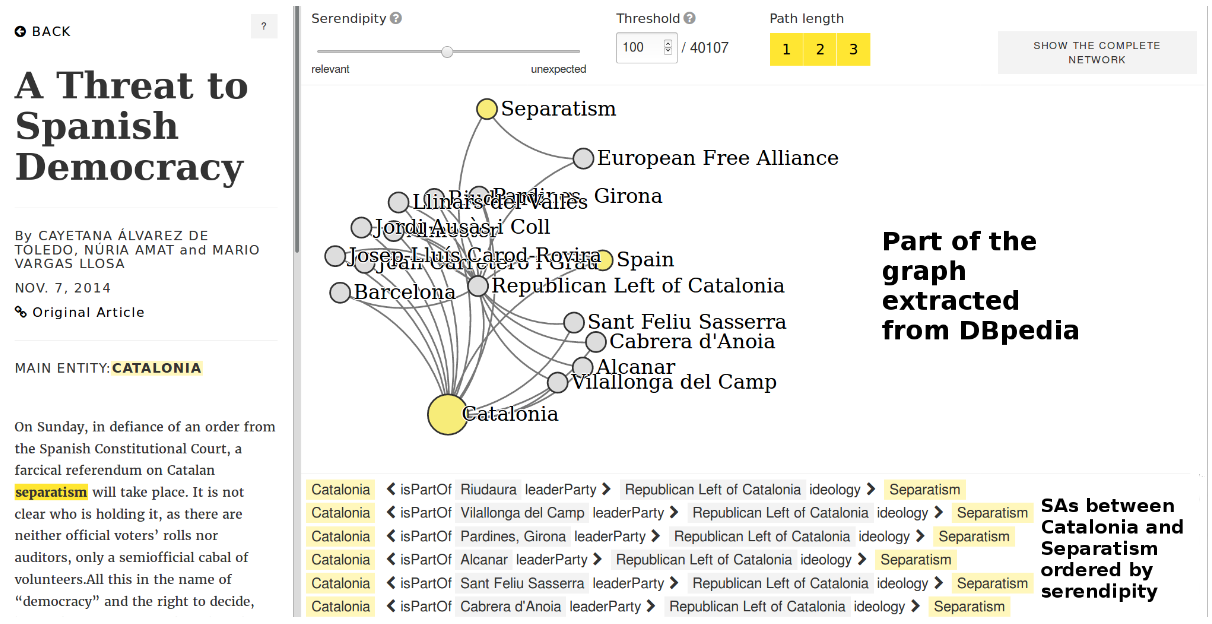
Abstract. Knowledge Graphs (KG) represent a large amount of Semantic Associations (SAs), i.e., chains of relations that may reveal interesting and unknown connections between different types of entities. Applications for the contextual exploration of KGs help users explore information extracted from 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 (KGs) describe real-world entities and their properties. Some of these properties represent links to other entities, providing a rich source of relational information. Languages recommended by W3C like RDF[1] can be used to publish KGs as (open) linked data, but KGs are frequently used also in the industry. KGs support a large variety of applications, including those targeted to knowledge exploration. Differently from query answering approaches, designed to return information relevant to specific information needs explicitly expressed by the users, knowledge exploration approaches are designed to deliver information interesting for the users in a more proactive fashion [1].

[1]https://www.w3.org/RDF/

We use *contextual KG exploration* to refer to a KG exploration setting in which a user who is carrying out a familiar task, e.g., querying a search engine, watching media content [2], reading a text of interest [3,4], receiving content extracted from a KG, selected and pushed to her in a proactive fashion. In these approaches, an input text (which may be extracted from any media content) is used as an entry point to determine which pieces of information can be interesting to show users in this context. Named Entity Recognition & Linking techniques [3] can be used to find one or more mentions of KG entities within the text. Then some information about these entities, selected based on its estimated relevance, is shown to the users. Examples of these approaches are entity expansion in [3] and entities extracted from Google's KG returned shown to users posting entity queries to Google's search engine.


BACK

A Threat to Spanish Democracy

By CAYETANA ÁLVAREZ DE TOLEDO, NÚRIA AMAT and MARIO VARGAS LLOSA

NOV. 7, 2014

Original Article

MAIN ENTITY: CATALONIA

On Sunday, in defiance of an order from the Spanish Constitutional Court, a farcical referendum on Catalan separatism will take place. It is not clear who is holding it, as there are neither official voters' rolls nor auditors, only a semiofficial cabal of volunteers.All this in the name of "democracy" and the right to decide,

Serendipity: relevant — unexpected

Threshold: 100 / 40107

Path length: 1 2 3

SHOW THE COMPLETE NETWORK

Catalonia < isPartOf Riudaura leaderParty > Republican Left of Catalonia ideology > Separatism
Catalonia < isPartOf Vilallonga del Camp leaderParty > Republican Left of Catalonia ideology > Separatism
Catalonia < isPartOf Pardines, Girona leaderParty > Republican Left of Catalonia ideology > Separatism
Catalonia < isPartOf Alcanar leaderParty > Republican Left of Catalonia ideology > Separatism
Catalonia < isPartOf Sant Feliu Sasserra leaderParty > Republican Left of Catalonia ideology > Separatism
Catalonia < isPartOf Cabrera d'Anoia leaderParty > Republican Left of Catalonia ideology > Separatism

Fig. 1. An example of contextual exploration of KG with SAs from the DaCENA interface

1.1. Contextual Exploration of KG with Semantic Associations

In our previous work [4], we designed an application that goes beyond showing plain properties of entities like the above-mentioned applications do: with our data journalism application DaCENA [2], we let users who are reading a text explore **Semantic Associations** (SAs) extracted from the KG. SAs are loop-free semi-walks of finite length that connect two entities in the KG [5,4]. SAs between entities extracted from the input text reveal complex connections between entities, which provide new and interesting insights into the topic of the input text.

DaCENA presents to a user a set of SAs extracted from a KG as a data context for the article that she is reading. The reference KG for DaCENA is DBpedia[3]. End users can read the article and explore the extracted SAs from an interactive interface. Figure 1 shows a screenshot of the interface, with a news article extracted from the NYT [4] (curious readers can play with the example shown in the figure online [5]). The graph shows the k-most interesting SAs, 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. 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 to ensure that we use fresh data. Therefore, texts and data are processed off-line so as to make the interactive visualization features as much fluid as possible (for further details about DaCENA processing steps we refer to [4]).

1.2. Challenges for Personalized Contextual Exploration of KG with SAs

Deciding which properties are valuable to be shown to users is challenging for every contextual KG exploration approach, but the problem is even more compelling when SAs are included in the information that is delivered to users, because of the very large amount SAs that can be found between entity pairs extracted from even relatively short texts. For example, for the article used in the exploration use case depicted in Figure 1, we extracted 40.107 SAs from DBpedia. Otherwise, preliminary user studies to evaluate the usability of our DaCENA application suggest that users do not want to look at more than 100 SAs.

A crucial problem to support users in exploring SAs is to provide effective methods to identify those few ones among the many that can be extracted, which are

[2]http://www.dacena.org
[3]http://wiki.dbpedia.org/
[4]https://goo.gl/RFvqZh
[5]http://www.dacena.org/article/84

more interesting for the user in this context. In other words the crucial problem to solve to support SA exploration is to find an effective ranking function. Several approaches have been proposed that use contextless measures based on graph analytics to rank and filter SAs [5,6]. Others make use of machine learning methods to learn which associations are more interesting based on labels provided by a group of annotators [7]. In our previous work we defined a context-aware measure that considers the relevance of the SAs with respect to the input text [4]. However, none of the above-mentioned ranking approaches places the individual user in the loop, supporting the personalized exploration of SAs by adapting the ranking function to her preferences. These approaches are thus inadequate if *different users are interested in different kinds of SAs*. We refer to the latter as to the "personalization hypothesis" for KG exploration with SAs. For example, if we consider the exploration use case depicted in Figure1, 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.

1.3. Contribution of this Paper

In this paper we propose a pay-as-you-go approach to personalized contextual exploration of large KGs with SAs. With our approach a user who is reading a text can explore a set of SAs heuristically ranked by Serendipity [4], i.e., estimated to be relevant to the input text as well as unexpected (and thus interesting) for her. Then she can iteratively refine the ranking by rating few SAs at the time, thus expressing her preferences. We use a learning to rank algorithm, i.e., RankSVM [8] to learn from the individual user ratings and an active sampling method [9] is used to select the SAs on which ratings are collected. To support RankSVM, we use a combination of state-of-the-art measures that consider the graph topology, the semantics of paths that occur in SAs, relevance with respect to the text, and the temporal relevance of the entities occurring in the SAs. In order to learn better from a limited number of rates we define an Active Learning to Rank (ALR) approach that supports the iterative improvement of the ranking quality, measured as adherence to the individual user preferences, while minimizing the user effort. On the one hand, our active learning method overcomes a limitation of a previous approach [7], which does not attempt to minimize the number of ratings collected from the users. On the other hand, by using Serendipity as heuristic function, we solve the cold-start problem that characterizes active sampling methods for ALR, i.e., the selection of informative samples requires an initial ranking, and at the same time, we are able to show to the user reasonably interesting SAs even before collecting any ratings.

Experiments conducted with two different datasets show that our approach is capable of improving the quality of the ranking over time using a limited amount of user ratings. We compare our approach with different baselines and alternative configurations for ALR. These configurations use different bootstrapping methods to overcome the cold start problem using different sampling methods. We show that our approach performs significantly better in terms of ranking quality improvement than all these alternative approaches, despite being more efficient and supporting a full pay-as-you-go, and thus more appealing, interaction model. Finally, the datasets constructed to evaluate our method provide clear evidence for the personalization hypothesis, supporting the main motivation behind our approach.

To the best of our knowledge, our approach provides the first application of active sampling to learning to rank for SAs and the first full pay-as-you go approach to contextual exploration of KGs with SAs. This paper presents an extended version of a paper accepted for publication at ESWC 2017 [10]. While in the latter we explored the performance of different approaches, in this paper we describe in more details our approach based on the combination of Serendipity and Active Learning, proving the significance of the results of our experiments and providing further insights about our findings.

The paper is organized as follows: in Section 2, we explain our ALR; in Section 3 we describe the experiments conducted to evaluate our model; in Section 4 we discuss related work, while in Section 5 we draw some conclusions and discuss future work.

2. Serendipity and Active Learning To Rank for Semantic Associations

In this section we explain in details our ALR model.

2.1. ALR Loop and Algorithms

We want to learn a ranking function for each user and we use the RankSVM [8,11] algorithm to do this.

RankSVM has been chosen for because it is considered a state of the art algorithm and has been widely used in the learning to rank domain [12,13]. This force us to solve two problems: we cannot expect that a user is willing to provide the large number of ratings that would be required to effectively train the RankSVM algorithm (**user-effort problem**); RankSVM requires an initial set of ratings to initialize the model (**cold-start problem**). To solve the user-effort problem we decide to use active learning techniques [14], which have been proposed to reduce the number of observations needed to train a classifier. In our case, active learning is used to reduce the number of ratings required to train the learning to rank algorithm.

To solve the cold-start problem we use Serendipity, an heuristic ranking function [4]. The advantage of using an heuristic function is that we can collect the ratings needed to initialize the RankSVM model on a set of SAs that are also (heuristically) estimated to be interesting for the user. Consider the example with the DaCENA application discussed in Section1. Only a subset of SAs deemed to be more interesting for the user can be shown to her in the user interface. By using Serendipity to bootstrap RankSVM, we can let users rate SAs that are in the subset shown in the user interface. Unfortunately, being Serendipity an heuristic function not specifically introduced to select samples that are informative for training RankSVM, it is uncertain weather this method is able to sample also useful training data. In addition, in order to initialize RankSVM, we need at least two ratings with different scores, meaning that we may need to sample more than once in order to initialize RankSVM. Alternative approaches proposed to solve the cold-start in previous work use clustering algorithms to collect ratings on a sample of SAs that are more representative, and thus more suitable to train RankSVN. Otherwise, these approaches could require that a user rates a set of SAs that are not interesting to him. In the experiments of Section 3 we will show that Serendipity leads to better performance than these clustering algorithms, despite the above-mentioned concerns. In addition, we will see that only a limited number of time we need to sample more than once before initializing the RankSVM model (see Section 3.3 for more details).

The model proposed to personalize the exploration of KG is based on the learning loop described in Figure 2. At each iteration, user ratings collected on small samples of SAs are used to train RankSVM and to update the ranking of the whole set of SAs. To better illustrate the loop, an example with two iterations, based on an actual run of our model, is depicted in Figure 3. Ratings shown in the red circles represent ratings given by one user that contributed to our experimental evaluation. Ratings shown in the blue clouds represent the ratings eventually given by the same user after having rated all SAs. We describe each step of the loop here below.

Step 1 - Bootsrapping. In the bootstrapping phase, SA are ranked by Serendipity. The user can view this pre-ordered set of SAs and then decide, if she is not satisfied with the result, to start a personalization phase. In this case, a small number of top-k SAs ranked by Serendipity are selected to be rated by the user. In our experiments, we use k=3 and k=5 on two different datasets (see Section 3 for more details about the choice of these parameters). **Step 2 - User Ratings.** The user labels the SAs selected in the first step (thus the SAs obtained with the serendipity heuristic) using ratings in a graded scale, e.g., $< 1, 2, 3, 4, 5, 6 >$, where higher grades represent higher interest for an SA. **Step 3 - Ranking.** We use the ratings provided by the user to train RankSVM, which ranks all the remaining SAs by assigning them a score. **Step 4 - User Decision.** The user can see the output of the ranking function and if she is satisfied with the ranking obtained so far, the loop stops. Else, we further improve the ranking by letting the user trigger the selection of a new sample of SAs to rate. **Step 5 - Active Sampling.** An active sampling algorithm proposed for document learning [9], referred to as AUC-Based Sampling in the paper, is used to find the observations (SAs) for which ratings are estimated to be more informative. The algorithm uses the scores determined by the learned ranking function, which motivates the reason for using a different algorithm for sampling the data used to bootstrap the model. 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 Figure 3, it can be noticed that the quality of the ranking improves after the second iteration (Step 3). The main steps of the loop, i.e., bootstrapping, ranking and active sampling, are explained in more details here below.

Bootstrapping with Serendipity. Serendipity is defined as a parametric linear combination of *Relevance*, a measure that evaluates the relevance of an SA with respect to a text, and *Rarity*, a measure that evaluates how much an SA may be unexpected for the users. A SA is relevant if a virtual document representing its en-

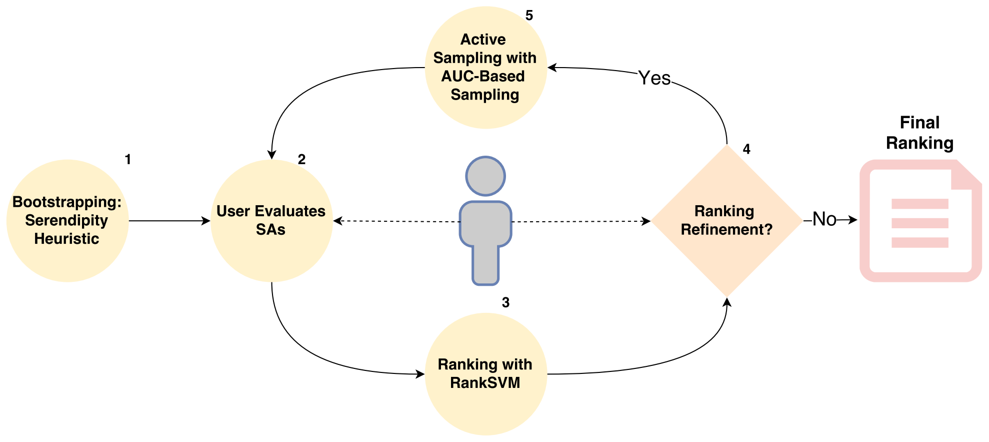


Fig. 2. Workflow of the Active Learning to Rank model

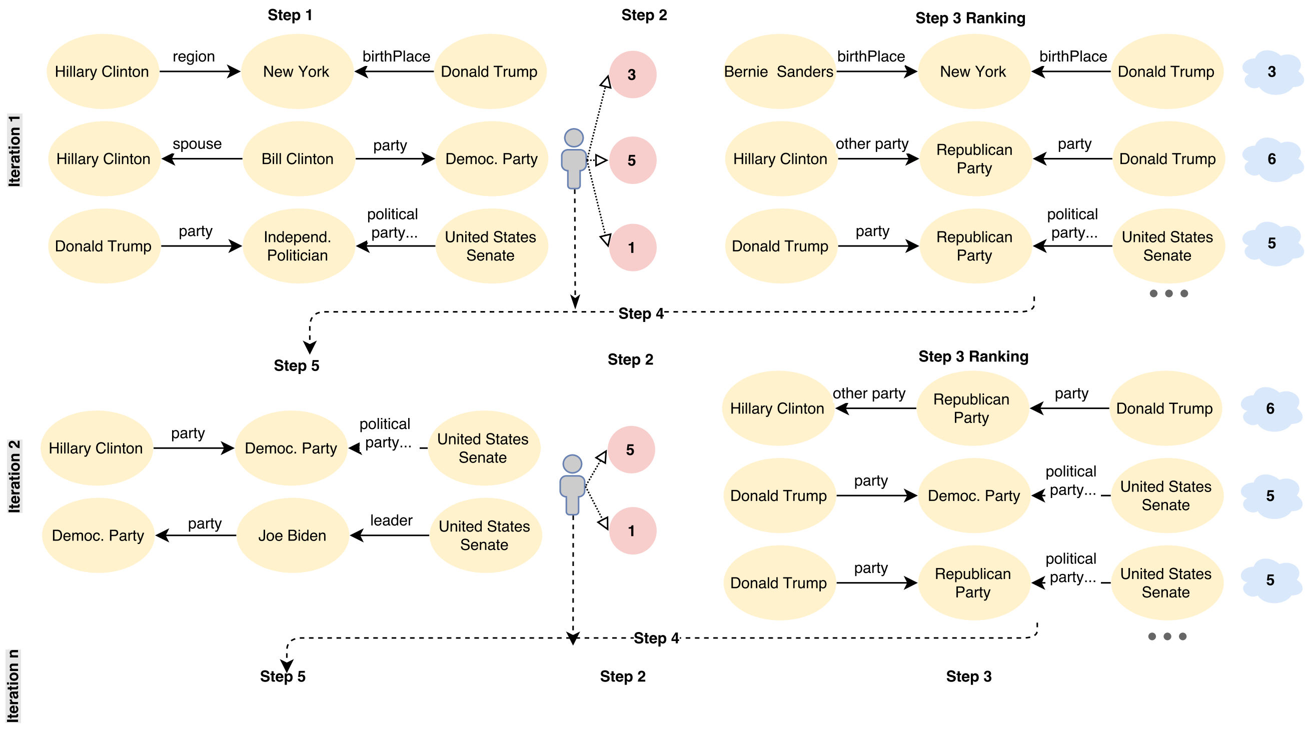


Fig. 3. Example of iterative ranking refinement with the ALR model

tities is similar to the input text. Relevance of a SA π to an input text *text* is thus defined as follows:

$$relevance(text, \pi) = cos(v_{text}, v_\pi),$$

where v_{text} and v_π are word vectors representing the text and the SA respectively. The text from which we generate v_π is built as the concatenation of short texts describing the entities occurring in π in the KG (in particular, we used DBpedia abstracts[6]). Word vectors are weighted using TF-IDF, where for each input text (and the SAs extracted from it) we build a dedicated vector space.

An SA is unexpected when it is composed by properties that are not frequently used in the KG, which can

[6]http://dbpedia.org/property/abstract

be captured by a Rarity measure. Rarity of a property p can be defined as follows:

$$rarity(p) = 1 - n\_frequency(p)$$

where, $n\_frequency(p)$ represents the frequency of p in the KG normalized into the range $[0, 1]$ using the max-min method. In other words rarity is defined as the inverse normalized frequency of a property. Rarity of a SA is defined as the average rarity of the properties occurring in the SA.

Since Relevance and Rarity are normalized in the interval $[0, 1]$, they can be smoothly combined. Let $\alpha \in [0, 1]$ 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 as

$$S(\pi, text) = \alpha relevance(\pi, text) + (1-\alpha)\ rarity(\pi)$$

We show an example of the top-2 ranked SAs and one of the bottom-2 ranked SAs, computed using Serendipity with $\alpha = 0.5$ in Figure 4. The text[7] from which those SAs were extracted from is shown on the left-hand side of the figure. The top ranked SAs contain the entity Hillary Clinton, which is relevant to the article text. The order of the SAs showed is consistent with the definition of Serendipity: the predicate *birthPlace* is used frequently in DBpedia (and thus probably uninteresting) and *George W Cate* and *Joseph K Allen* are not mentioned in the article text. The *birthPlace* predicate is present also in the top-2 ranked SAs, but higher relevance of the entities occurring in these SAs to the article text lead to higher Serendipity scores.

Ranking with RankSVM We choose to use Rank SVM to learning the ranking function based on the user ratings. Rank SVM has become one of the state of the art algorithm for learning to rank documents [8,11]. RankSVM is an adaptation of Support Vector Machines to solve ranking problems and represents the items to rank as feature vectors. RankSVM is based on a pairwise ranking algorithm. This means that the input to the model is a set of pairs that contains two observations with their relative order. As an example, if in the training set there are two feature vectors x_i and x_j, the input to the RankSVM would be $\{(x_i, x_j), y\}$, where y is a label that indicates the relative order between x_i and x_j. The label $y = +1$ if x_i should be ranked higher then x_j and viceversa. RankSVM is capable to build such pairs based on a set of ratings provided on an arbitrary set of observations. The model is trained at an iteration t using the SAs rated by an individual user until iteration t and update the ranking of the remaining SAs.

Active Sampling. We use AUC-based Sampling as active sampling algorithm in our model [9]. This algorithm has the main aim of minimizing rank loss and does not consider the pairwise structure of the problem making the algorithm sub-optimal for this task, but faster to compute. This method optimizes the Area Under the ROC Curve and accepts ratings on an arbitrary set of observations. Other methods that consider the pairwise structure are more expensive to compute. An example of the SAs selected by this active learning to rank algorithm can be seen in Step 5 of Figure 3. While it was designed and developed in a binary learning to rank setting the algorithm was able to obtain good results even in a multi-rating setting as shown in Section 3.

2.2. *Features*

To represent the SAs as features vectors we used different measures. Many of the used measures return values with different ranges. In order to make measures comparable, we normalize the values by scaling to have zero mean and unit variance [15].

2.2.1. *Topological Features*

Here we present the features that are based on the topological structure of the graphs used to represent the data. These features are mostly based on centrality measures that evaluates the importance of the entities in the SAs and can be computed considering two different graphs: the first one is the KG from which we extract the SAs, which will be referred to as *global graph*; the second one is the subgraph of the KG that consists of the SAs extracted from the input text, which will be referred to as the *local graph*. Using these two graphs we can compute *global* and *local* scores of centrality for the entities. Entity with an high centrality scores can be considered important due to the high number of links they have (and thus, more or less interesting for a user depending on her preferences). Once we have computed centrality scores for all the entities in a SAs, scores used as features are computed as the average of the scores computed for the entities. Then we normalize each score as defined above.

Global PageRank. We use the data in [16] to collect a global PageRank score inside DBpedia. In this, way

[7]https://www.theguardian.com/us-news/2016/jun/16/bernie-sanders-will-work-with-clinton-donald-trump-speech

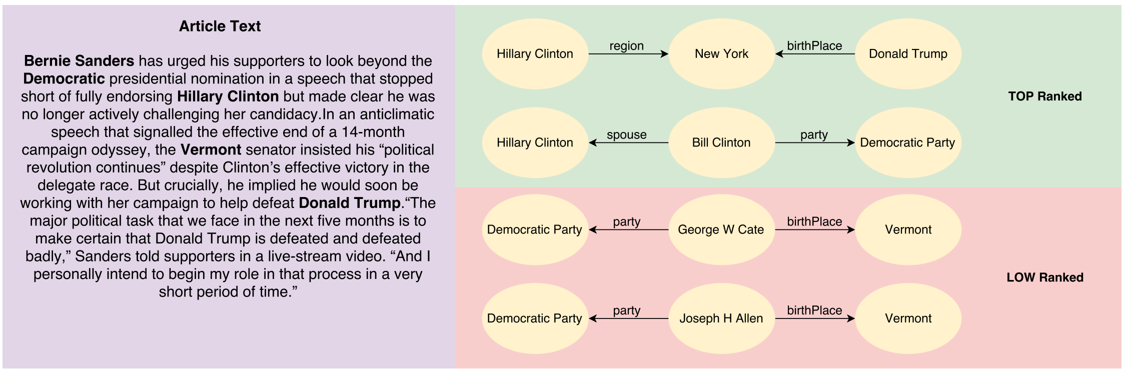

Article Text
Bernie Sanders has urged his supporters to look beyond the Democratic presidential nomination in a speech that stopped short of fully endorsing Hillary Clinton but made clear he was no longer actively challenging her candidacy.In an anticlimatic speech that signalled the effective end of a 14-month campaign odyssey, the Vermont senator insisted his "political revolution continues" despite Clinton's effective victory in the delegate race. But crucially, he implied he would soon be working with her campaign to help defeat Donald Trump."The major political task that we face in the next five months is to make certain that Donald Trump is defeated and defeated badly," Sanders told supporters in a live-stream video. "And I personally intend to begin my role in that process in a very short period of time."
Hillary Clinton
region
New York
birthPlace
Donald Trump
Hillary Clinton
spouse
Bill Clinton
party
Democratic Party
TOP Ranked
Democratic Party
party
George W Cate
birthPlace
Vermont
Democratic Party
party
Joseph H Allen
birthPlace
Vermont
LOW Ranked

Fig. 4. Example of top-2 and bottom-2 SAs ordered by Serendipity

we are able to get an overall estimation of the importance of an entity inside the KG. With this feature, entity that are central in DBpedia (like United States of America[8] and Barack Obama[9]) will be considered important even if they have a limited number of links in the subgraph extracted from the text.

Local PageRank. We compute PageRank on the local graph defined by the SAs extracted from the input text.

Local HITS. We ran the HITS (Hyperlink-Induced Topic Search)[17] 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 in the feature vectors describing a SA: one for the average of the authority values and one for the average of the hub values.

2.2.2. *Predicate-Based Features*

In this section we present features that considers predicates occurring in the SAs. We decided to use features that are focused on finding the most important/interesting predicates inside the SAs.

Path Informativeness. It is a measure defined in [5], based on the concept of Predicate Frequency Inverse Triple Frequency (PF-ITF). This measure tries to identify discriminative paths in a way that is similar to the ones commonly defined for text like TF-IDF (Term Frequency - Inverse Document Frequency).

Path Pattern Informativeness. It is a measure based path patterns, defined in [5], to get the informativeness of patterns extracted from paths. The path pattern is used to generalize the paths in a dataset.

[8]http://dbpedia.org/page/United_States

[9]http://dbpedia.org/page/Barack_Obama

Rarity. This measure is the unexpectedness factor presented in the Serendipity Heuristic, more details can be found in Section 2. For example predicate frequently used in DBpedia, like birthPlace, have low rarity score.

2.2.3. *Relevance Features*

This last section illustrates the features that have been defined to find those SAs that are more related to the context of the article that was analyzed.

Relevance. This measure is the relevance factor presented in the Serendipity Heuristic, more details can be found in Section 2.

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[10] for the page Paris, we see that the page has been accessed 8.331 times on 12-11-2015 and 171.988 times on 14-11-2015, 13-11-2015 being the date of terrorist attacks in Paris. The scores used as features are computed as the average of the scores computed for the entities.

3. Experiments

The validity of the personalization hypothesis and the evaluation of the performance of our model are tested through experiments. The targets of our application are fairly educated users familiar with IT technologies. In our experiments we have involved master students from Computer Science, Mechanical Engineering and Communication Sciences with good En-

[10]http://tools.wmflabs.org/pageviews/

glish reading skills. All the data sets used in our experiments are available online[11]. The experiments were run on a machine with a Intel Core i5 (4th Gen, 1.6Ghz).

3.1. Experimental Settings

Two different dataset were built to test the performance of our approach. We collected these dataset because we weren't able to find an open dataset that could be used in our domain: in our approach the context is given by the text, that is fundamental for our exploration scenario. Each one consists of triples $< text_i, A_i, ratings_{u,i} >$, where $text_i$ is a piece of text that was extracted from an article retrieved from online news platforms like NYT[12] and The Guardian[13], A_i is the set of all SAs extracted from $text_i$ with the DaCENA application, and $ratings_{u,i}$ contains the ratings assigned by a user u to every SA that is present 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 show an example of data contained in the dataset used for experiments in Figure 5, we show the piece of the article[14] user had to read for the experiments, two SAs extracted form that text and the respective rating given by the user. 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 [18].

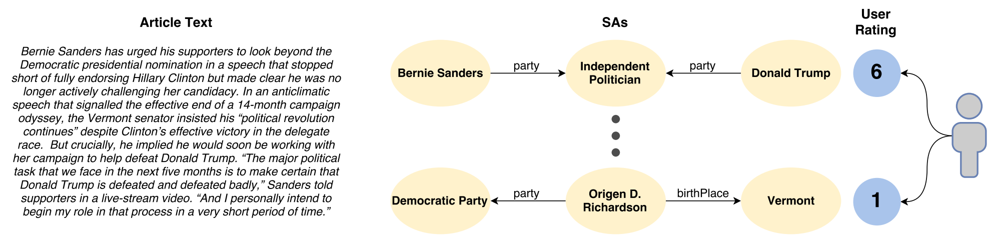
Differently from a five-valued ordinal scale, this scale provides a symmetric range that clusters scores in two sets: scores with a negative tendency (1, 2 and 3) and scores with a positive tendency (4, 5 and 6). Each user had to evaluate the complete set of associations extracted from one text, thus we had to choose texts small enough to let users perform their task without being subject to fatigue bias [18]. We thus selected the first self-contained paragraphs of articles from NYT and Guardian with the following features: the article's topic concerns politics and 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 minutes - little below the fatigue bias threshold mentioned in [18]. 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 on the selection of the articles. After evaluating the first article we asked the user if she wanted to evaluate an other article or if she wanted to stop; some articles were evaluated by different users and some users evaluated more then one article. We stopped gartering users for the evaluation when we collected evaluations by at least 3 users on each article (for a total number of 5 articles), which resulted in a total of 14 different users, and 25 gold standards (personal rankings). The average number of SAs for each article present in the dataset is equal to 73.2.

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. The articles contained 890 and 3539 SAs respectively. 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. At the end of this evaluation we had three dataset (coming from two articles) on which experiment on, with an average number of SAs equal to 2656 SAs. .

Measures for the Evaluation. Using the ideal rankings in the two dataset, we decided to 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. nDCG@10 was used to give more importance to the first retrieved SAs. In addition, we wanted to have an aggregate performance measure, and thus we computed the Area Under the nDCG@10 Curve (AUNC), the curve is based on the nDCG@10 values at each iteration.

Settings of the experiments We carry out experiments in two different settings. 1) In *Contextual Exploration Settings*, we consider the workflow as implemented in a system that supports contextual exploration: the set from which we select the SAs to label is the same set

[11]https://github.com/vinid/semantic-associations-survey
[12]www.nytimes.com/
[13]https://www.theguardian.com/
[14]https://www.theguardian.com/us-news/2016/jun/16/bernie-sanders-will-work-with-clinton-donald-trump-speech



Bernie Sanders has urged his supporters to look beyond the Democratic presidential nomination in a speech that stopped short of fully endorsing Hillary Clinton but made clear he was no longer actively challenging her candidacy. In an anticlimatic speech that signalled the effective end of a 14-month campaign odyssey, the Vermont senator insisted his "political revolution continues" despite Clinton's effective victory in the delegate race. But crucially, he implied he would soon be working with her campaign to help defeat Donald Trump. "The major political task that we face in the next five months is to make certain that Donald Trump is defeated and defeated badly," Sanders told supporters in a live-stream video. "And I personally intend to begin my role in that process in a very short period of time."

Fig. 5. Example of data contained in the dataset used for the experiments

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. 2) In *Cross Validation Settings*, active sampling always picks SAs from the training set similarly to the approach used in active learning to rank [9]. This latter approach was not defined for interactive exploration and in fact is used because it is helpful to evaluate the robustness of the model. We can use 2Fold-Stratified Cross Validation (CV) to make sure that results can be reasonably generalized and do not depend on specific data. 2Fold CV was used because the cardinality of the dataset was low. With 2Fold CV we were able to have enough data on which the active sampling algorithms were able to select the observations to rate and enough data to test the model.

3.2. Configurations and Baselines

Direct comparison with other state-of-the-art approaches is difficult because we could not find an active learning to rank approach for SAs, in addition often appraoches over SAs were evaluated in a context-free settings, while in our contextual exploration scenario, input text is a basic component for the ranking. Thus we compared our approach to others we defined. In those other approaches, that use a learning to rank model, we used again RankSVM to rank SAs. A summary of the configurations we have defined is visible in Table 1, where we show, for each algorithms the basic configuration for the three steps of our model, namely **Bootstrapping**, **Active Sampling** and **Ranking**. The algorithms that have are signed with the blue columns are the ones that use active sampling techniques to reduce the number of observations needed to learn the model, while the ones in red do not use active sampling. We will illustrate the details of the algorithms in the remaining part of this section.

The **Bootstrapping** phase of our model is currently guided by the serendipity heuristic (with $\alpha = 0,5$). We decided to compare it with the use of clustering algorithm, since they have been already used in active learning settings [19,20], in which they are used to find the first observations for machine learning models. The choice of a clustering approach was due to the following idea: the observations nearer to the clusters means can be considered representative of the cluster and thus it could be informative to learn preference among clusters. We consider two clustering algorithms: the Gaussian Mixture Model and the Dirichlet Gaussian Mixture Model. Dirichlet Gaussian Mixture Model was chosen due to its ability to automatically find the number of clusters for the dataset. For the Gaussian Mixture Model we use the silhouette coefficient [15] to detect the best number of clusters in each dataset. The first SAs to show to the user for evaluation are those nearer to the mean of each cluster found by the clustering algorithm, thus we selected one SAs for each cluster.

The active sampling phase of our model is done using AUC Based Sampling [9] (AS), we thus selected Pairwise Sampling [21] (PS) as an alternative to it. PS is based on the pairwise structure of the RankSVM approach, and it tries to find those observations that could be more interesting to evaluate by studying the uncertainty of the observations with respect to the ranking problem. PS combines two different kind of uncertainty a Global Uncertainty and Local Uncertainty, to prevent the selection of outliers in the active sampling phase, the algorithm is explained in [21]. Parameters of these two algorithms have been determined experimentally in cross validation, we set $\lambda = 0.8$ for AS and $p = 1$ for PS. Three baseline algorithms that do not use proper active learning were also tested:

– *Random + Random*: we use RankSVM to learn a ranking function, but the model is trained using ratings over randomly sampled SAs. Randomly selected SAs are thus used for the first step (initialization) and the active learning step. This random algorithm has been run multiple times to stabilize the results (100.000 times).

	Algorithm	**Bootstrapping**	**Sampling**	**Learning**
🟦	Serendipity AS	Serendipity Heuristic	AUC-Based Sampling	RankSVM
🟦	Serendipity PS	Serendipity Heuristic	Pairwise Sampling	RankSVM
🟦	Dirichlet AS	Dirichlet Gaussian Mixture Model	AUC-Based Sampling	RankSVM
🟦	Dirichlet PS	Dirichlet Gaussian Mixture Model	Pairwise Sampling	RankSVM
🟦	Gaussian AS	Gaussian Mixture Model	AUC-Based Sampling	RankSVM
🟦	Gaussian PS	Gaussian Mixture Model	Pairwise Sampling	RankSVM
🟥	Random Random	Random	Random	RankSVM
🟥	Random - No AL	No Bootstrapping	No Active Sampling	No Learning to Rank
🟥	Serendipity - No AL	No Bootstrapping	No Active Sampling	No Learning to Rank

Table 1

Details of the algorithms configurations

– *Serendipity No-AL*: we consider the ranking determined with Serendipity, which is not based on active learning and does not change across iterations. In this case RankSVM is not used.
– *Random No-AL*: we consider random rankings of SAs, which are not based on active learning and do change across iterations. In this case RankSVM is not used.

The last two algorithms are considered to understand if an incremental learning with active learning to rank can outperform a order given by a simple heuristic function and a random approach.

Configuration Details. To compare the approaches with the serendipity heuristic we had to define a few configurations so that the training set for the various algorithms would have been of the same size and thus balanced. In the SAMU data sets Dirichlet and Gaussian Clustering, in the first iterations, detected 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 SAs, on the average, for both Serendipity and Random we choose to select 3 SAs when using Serendipity and Random in the bootstrapping step. The active sampling step for this dataset extracts the top-2 ranked SAs as determined by the active sampling techniques used in the configuration. For the Random Random approach we again select 2 random SAs. The number of observations collected at each iteration was increased in LAFU since this dataset it is bigger. The clustering algorithms in this dataset detected an average number of cluster equal to 5, leading to 5 SAs to be labeled when using Serendipity and Random. In the active sampling, for the LAFU data set, we selected 6 observations to be labeled at each iteration for both AS and PS. Table 2 shows the number of SAs that each algorithms selects for each step. The *Temporal Relevance* could not be used in LAFU as a feature because articles in this dataset were not recent enough to be able to extract the page views. We used a RankSVM with polynomial kernel (degree equal to 2) on LAFU, since dataset were bigger and nonlinearity was more probable, that was able to output the results of a single iteration in what we considered interactive time (less then 2 seconds); on the SAMU dataset RankSVM was run with a linear kernel.

3.3. Results and Discussion

User interests and personalization. One of our assumption was that the personalization was needed, because different user are interested in different SAs. We measured Inter-Rater Reliability [22] (IRR) among users who provided ratings in the SAMU dataset, to prove that a personalized exploration is not only important but needed. IRR is used to compute the rate of agreement between the users, in our case a low rate of agreement would indicate a disagreement of the interest of different users with respect to the same SAs. We used to measures for evaluating the IRR of the dataset: Krippendorff's alpha (weighted using an ordinal matrix, since our context uses ordinal values), which output was 0.06154, and Kendall's W, from which we obtained a score of 0.2608. The Table 4 shows these results. IRR is low and distant from 1, the value that usually represents unanimity between the raters. We show, in Table 3, the distribution of the ordinal ratings for both datasets. We remark again that the SAs in the SAMU dataset have a rating scale that range from 1 (low interest) to 6 (high interest) while the one from LAFU are from 1 (low interest) to 3 (high interest).

Contextual Exploration Settings. The reader can view the results in Figure 6, where we plot the average nDCG@10 for the first five iterations of the model. On

	SAMU	LAFU
Step 1 #SAs		
Serendipity	Top-3 SAs with serendipity	Top-5 SAs with serendipity
Random	3 randomly extracted SAs	5 randomly extracted SAs
Clustering	3 SAs found on average	5 SAs found on average
Step 3 #SAs		
AS&PS	Top-2 SAs found with AL	Top-6 SAs found with AL
Random	2 randomly extracted SAs	6 randomly extracted SAs

Table 2

Number of observations selected at each steps by the algorithms

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

Table 3

Rating distribution in the two data sets

Krippendorff's alpha	0.0615
Kendall's w	0.268

Table 4

Inter-Rater Reliability on the SAMU data set

the average a user at the 5th iteration has given rating to 12 SAs for the SAMU dataset and 30 for the LAMU dataset. On the SAMU dataset the best algorithm appears to be the Serendipity AS one. One interesting thing to notice is that active learning, performs better then methods that do not use an incremental learning method (Random No-AL and Serendipity No-AL). This is important because it proves that active sampling is useful for maximizing the ranking function. In the LAFU dataset Serendipity No-AL was able to get a good result and, if we do not consider the Serendipity AS configuration, active learning required three iterations to perform better then Serendipity No-AL. The Random Random approach, that uses a random selection method for active sampling, performs better with each iterations, but the overall performance is low, compared to the other algorithms that use active sampling. This is an indication of how much the initial training set and the SAs selected actively are important for ranking algorithms. The areas computed can be found in Table 6 and show that Serendipity AS is the algorithm with the biggest area.

Cross Validation Settings. Cross validation has been used to provide evidence that the module is robust and the main result obtained is that the model with the best performance is ours. The conclusions are similar to the one already explained in the section above, the general performance in this case is slightly less good, mainly because in the cross validation setting the algorithms are not able to access to the testing data. The plots can be found in Figure 6 while the computed areas are in Table 5.

Interactive Time. We designed an approach that should interact with users. One of the most important requirements from the user side is the interactive time. An user is willing to use an application only if the response he gets from the platform are given in a short period of time. We computed the average number of seconds the model needs to train the ranking algorithms and provide a ranking to the user; in this case we tested the Serendipity AS algorithm. For the SAMU dataset the model was able to compute the result in 0.35 seconds while for the LAFU dataset the average number of seconds required is around 2. This means that as the dataset gets bigger, the algorithms requires longer time to compute the result.

Initial training set analysis. If a user in the first step (that can be done with the serendipity heuristic or the clustering algorithms) evaluates all the SAs with the same ordinal degree, we can not initialize the Rank SVM model. We thus evaluate the *time for first iteration* value, that corresponds to the average number of iterations needed for each method to have a training that can be used to training the learning to rank algorithm. Table 7 shows the result. We show data for both Cross Validation (CV) and Contextual Exploration Setting (CE). In LAFU data set the algorithms can find the first observations needed to train the model becomes more difficult for methods with the exception of Dirichlet that can probably adapt itself to the size of the dataset in an easier way, result that is consistent with how it is defined. Analyzing the plots 6 we can see that while Dirichlet clustering is able to rapidly gain an initial training set for the RankSVM model (typically around 1 iteration), this training set makes the

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

Table 5

AUNC in Cross Validation

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

Table 6

AUNC in Contextual Exploration

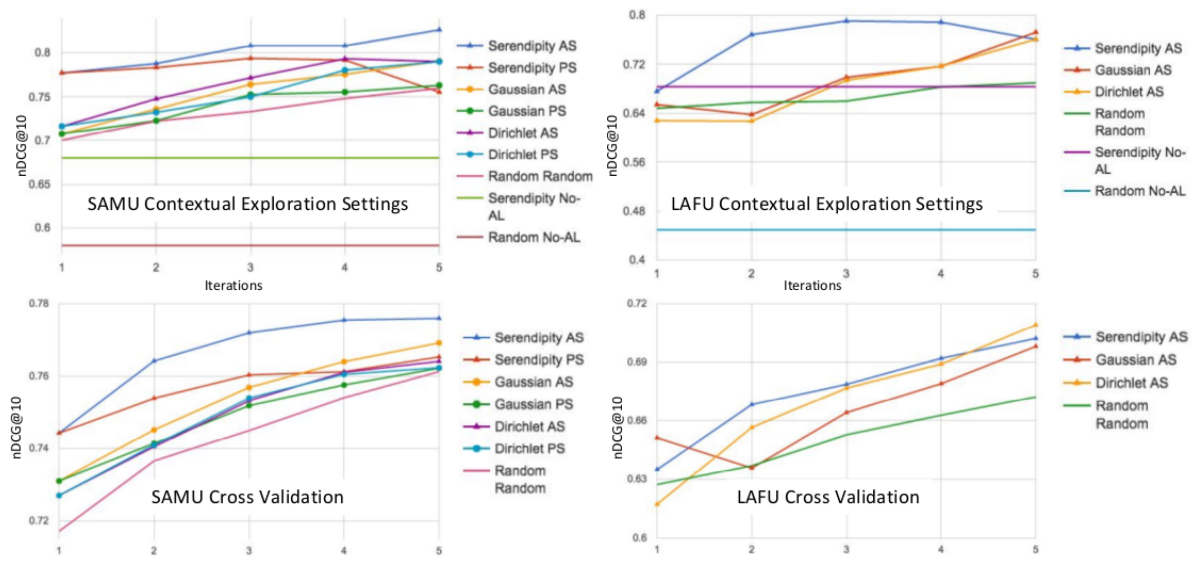


Fig. 6. nDCG@10 in Contextual Exploration and Cross Validation settings

model obtain a less good performance compared to the Serendipity one.

Wilcoxon test. To verify the results obtained by our model we performed the Wilcoxon test, on the various nDCG@10 obtained by the **last** iteration of the Cross-Validation, to prove that the final result of each algorithm is statistically significant with respect to the others. We generated a matrix with the algorithms configurations on the columns and the datasets on the row. An element $x_{i,j}$ of this matrix is the nDCG@10 of the last iteration of the Cross Validation for the i-th dataset obtained with the j-th algorithm. P-values obtained with the Wilcoxon test (that was run with an $\alpha = 0.05$) can be seen in Table 8, we signed with a "*" the values on which we are statistically significant. We can conclude that AS is significant with respect to PS and Random. When we use AS we can see that results with Serendipity, Dirichlet or Gaussian are correlated, this is because the only difference between Serendipity AS, Gaussian AS and Dirichlet AS lies in the first iteration. However, from an applicative point of view, Serendipity provides an approach that is smoother and cleaner for a user, because the SAs retrieved in the first iteration could already be interesting (since the heuristic measure combines relevance with unexpectedness), while using a clustering algorithm could force the user to evaluate SAs that are not interesting. Results of the test suggest that not only results obtained with AS seem more significant of those obtained with PS, but even of those obtained with a Random approach.

Discussion. We observed that different users have different interests, which motivates the need for personalization in KG exploration approaches. The serendipity heuristic with the AS algorithm model introduced in this paper shows remarkable improvement over other configurations. While the difference between Serendipity, Gaussian and Dirichlet is not statistically

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

Table 7

Average #Iterations to find the first training set for the ranking model

Wilcoxon	Dirichlet AS	Gaussian PS	Dirichlet PS	Serendipity AS	Serendipity PS	Random
Gaussian AS	0.07548	0.003781*	0.02365*	0.7915	0.173	0.0008081*
Dirichlet AS		0.09032*	0.5077	0.615	0.5782	0.05158*
Gaussian PS			0.1014	0.002255*	0.1073	0.731
Dirichlet PS				0.01247*	0.9578	0.4908
Serendipity AS					0.09573	0.02748*
Serendipity PS						0.5965

Table 8

P-values for the Wilcoxon signed-rank test on the SAMU dataset

significant, the serendipity heuristic is able to show a pre-ordered set of SAs from the first iteration, and this might be of big interest for the user. This is important for the user experience, since a user might not be interested in evaluating SAs that do not interest her.

Summary The results we got leave us with the following assertions:

- ALR is a good way to rapidly gain feedback from the user while optimizing their own ranking function
- The serendipity heuristics is a good method to initialize the ALR model and is able to show to the users a pre-ordered set of SAs from the beginning
- Our experiments showed that user are interested in different kind of SAs and thus the personalization of the exploration is an important and fundamental task

4. 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 specific entities selected by a user [23]. Other applications similar to RelFinder also incorporate measures to evaluate and explain SAs between two specific entities [6,5,24]. inWalk is another application for interactive linked data exploration based on thematic graphs, whose nodes represent clusters of similar linked data, and edges are proximity relations between the clusters [25]. Aeemo is another example of tool for knowledge exploration [26]: it uses a keyword based search to help user find summarized information about an entity using Wikipedia, Twitter and Google News. 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 [27] 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 an ALR approach.

Learning to Rank and Active Learning for KG Exploration Learning to rank has been extensively applied in document retrieval [28] 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 [9] collects labels over individual observations (SAs in our case) and solves the cold-start problem, mentioned before, by randomly selecting positive and negative instances from a subset of the data reserved for training. In our interactive approach we pick the SAs that are labeled by the user from 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 [21], 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. While many active learning to rank techniques and methods have been proposed in literature [29,30,9,21], to the best of our knowledge none of them as been applied for SAs active ranking. We choose to concentrate our attention only on two of them, mainly because they could be easily applied in a pairwise learning to rank setting with RankSVM, a well-known state-of-art algorithm in this context. the one that we use in our approach [9] has been chosen to provide a method that was computationally fast while the second one [21] was selected to test a pairwise sampling algorithm that uses uncertainty measure like those seen in active learning for classification problems [14](In Section 3 a few more details were given in Section). One approach that has been proposed the application of active learning in the context for KG exploration, has been applied to a classification problem, i.e., to decide which nodes should be included in a graph summary [31], which is very different from the learning to rank problem discussed in this paper.

5. Conclusion

Experimental results show that our approach, based on a serendipity heuristic and with the use of an AUC based active learning to rank algorithm is able to increase the ranking of the SAs while keeping the number of feedbacks requested to the user low. Moreover we have been able to prove that personalization of KG exploration is necessary since user are interested in different kind of relations. 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. An other research area that we plan to explore is the one of online learning to rank with the use of implicit feedbacks (like clicks) given by the users on the SAs; combining implicit and explicit feedbacks could greatly improve the performance of the models while lowering the effort requested to the user to get personalized results. 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 off-line. In journalism, freshness of information is relevant and we plan to further investigate methods to refresh/update SAs after processing in the future.

6. Acknowledgement

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

References

[1] Nikos Bikakis and Timos Sellis. Exploration and visualization in the web of big linked data: A survey of the state of the art. *preprint arXiv:1601.08059*, 2016.

[2] José Luis Redondo-García, Michiel Hildebrand, Lilia Perez Romero, and Raphaël Troncy. Augmenting TV newscasts via entity expansion. In *ESWC*, pages 472–476. Springer, 2014.

[3] Tabea Tietz, Joscha JÃd'ger, JÃűrg Waitelonis, and Harald Sack. Semantic annotation and information visualization for blogposts with Refer. In *VOILA '16*, volume 1704, pages 28 – 40, 2016.

[4] Matteo Palmonari, Giorgio Uboldi, Marco Cremaschi, Daniele Ciminieri, and Federico Bianchi. Dacena: Serendipitous news reading with data contexts. In *ESWC*, pages 133–137. Springer, 2015.

[5] Giuseppe Pirrò. Explaining and suggesting relatedness in knowledge graphs. In *ISWC*, pages 622–639. Springer, 2015.

[6] Gong Cheng, Yanan Zhang, and Yuzhong Qu. Explass: exploring associations between entities via top-k ontological patterns and facets. In *ISWC*, pages 422–437. Springer, 2014.

[7] Na Chen and Viktor K Prasanna. Learning to rank complex semantic relationships. *IJSWIS*, 8(4):1–19, 2012.

[8] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

[9] Pinar Donmez and Jaime G Carbonell. Active sampling for rank learning via optimizing the area under the ROC curve. In *ECIR*, pages 78–89. Springer, 2009.

[10] Federico Bianchi, Matteo Palmonari, Marco Cremaschi, and Elisabetta Fersini. Actively learning to rank semantic associations for personalized contextual exploration of knowledge graphs. In *ESWC*, 2017.

[11] Ching-Pei Lee and Chih-Jen Lin. Large-scale linear ranksvm. *Neural computation*, 26(4):781–817, 2014.

[12] R Busa-Fekete, György Szarvas, Tamás Elteto, and Balázs Kégl. An apple-to-apple comparison of learning-to-rank algorithms in terms of normalized discounted cumulative gain. In *20th European Conference on Artificial Intelligence (ECAI 2012): Preference Learning: Problems and Applications in AI Workshop*, volume 242. Ios Press, 2012.

[13] Tao Qin, Tie-Yan Liu, Jun Xu, and Hang Li. Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, 13(4):346–374, 2010.

[14] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.

[15] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education India, 2006.

[16] Andreas Thalhammer and Achim Rettinger. PageRank on Wikipedia: Towards General Importance Scores for Entities. In *ESWC 2016, Revised Selected Papers*, pages 227–240. Springer International Publishing, Cham, 2016.

[17] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *JACM*, 46(5):604–632, 1999.

[18] Federico Cabitza and Angela Locoro. Questionnaires in the design and evaluation of community-oriented technologies. *International Journal of Web-Based Communities (to appear)*, 13(1), 2017.

[19] Jaeho Kang, Kwang Ryel Ryu, and Hyuk-Chul Kwon. Using cluster-based sampling to select initial training set for active learning in text classification. In *PAKDD*, pages 384–388. Springer, 2004.

[20] Weiwei Yuan, Yongkoo Han, Donghai Guan, Sungyoung Lee, and Young-Koo Lee. Initial training data selection for active learning. In *ICUIMC*, page 5. ACM, 2011.

[21] Buyue Qian, Hongfei Li, Jun Wang, Xiang Wang, and Ian Davidson. Active learning to rank using pairwise supervision. In *SIAM Int. Conf. Data Mining*, pages 297–305. SIAM, 2013.

[22] Kilem L Gwet. *Handbook of inter-rater reliability: The definitive guide to measuring the extent of agreement among raters*. Advanced Analytics, LLC, 2014.

[23] Philipp Heim, Sebastian Hellmann, Jens Lehmann, Steffen Lohmann, and Timo Stegemann. Relfinder: Revealing relationships in rdf knowledge bases. In *SAMT*, pages 182–187. Springer, 2009.

[24] Lujun Fang, Anish Das Sarma, Cong Yu, and Philip Bohannon. Rex: explaining relationships between entity pairs. *Proceedings VLDB*, 5(3):241–252, 2011.

[25] Silvana Castano, Alfio Ferrara, and Stefano Montanelli. inwalk: Interactive and thematic walks inside the web of data. In *EDBT*, pages 628–631. Citeseer, 2014.

[26] Andrea Giovanni Nuzzolese, Valentina Presutti, Aldo Gangemi, Alberto Musetti, and Paolo Ciancarini. Aemoo: exploring knowledge on the web. In *ACM Web Science Conference*, pages 272–275. ACM, 2013.

[27] Ilaria Tiddi, Mathieu dâĂŹAquin, and Enrico Motta. Learning to assess linked data relationships using genetic programming. In *ISWC*, pages 581–597. Springer, 2016.

[28] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[29] Bo Long, Olivier Chapelle, Ya Zhang, Yi Chang, Zhaohui Zheng, and Belle Tseng. Active learning for ranking through expected loss optimization. In *ACM SIGIR conference on Research and development in information retrieval*, pages 267–274. ACM, 2010.

[30] Wei Chu and Zoubin Ghahramani. Extensions of gaussian processes for ranking: semisupervised and active learning. *Learning to Rank*, page 29, 2005.

[31] Meng Fang, Jie Yin, and Xingquan Zhu. Active exploration for large graphs. *Data Mining and Knowledge Discovery*, 30(3):511–549, 2016.