Detecting Similar Linked Datasets Using Topic Modelling

Michael Röder[1], Axel-Cyrille Ngonga Ngomo[1](✉), Ivan Ermilov[1], and Andreas Both[2]

[1] AKSW, Leipzig University, Leipzig, Germany
{roeder,ngonga}@informatik.uni-leipzig.de
[2] Mercateo AG, Leipzig, Germany

Abstract. The Web of data is growing continuously with respect to both the size and number of the datasets published. Porting a dataset to five-star Linked Data however requires the publisher of this dataset to link it with the already available linked datasets. Given the size and growth of the Linked Data Cloud, the current mostly manual approach used for detecting relevant datasets for linking is obsolete. We study the use of topic modelling for dataset search experimentally and present Tapioca, a linked dataset search engine that provides data publishers with similar existing datasets automatically. Our search engine uses a novel approach for determining the topical similarity of datasets. This approach relies on probabilistic topic modelling to determine related datasets by relying solely on the metadata of datasets. We evaluate our approach on a manually created gold standard and with a user study. Our evaluation shows that our algorithm outperforms a set of comparable baseline algorithms including standard search engines significantly by 6 % F1-score. Moreover, we show that it can be used on a large real world dataset with a comparable performance.

1 Introduction

The Web of Data and the Linked Open Data Cloud have grown considerably over the last years and are continuing to grow steadily. Following the statistics of LODStats,[1] several thousands of RDF datasets can already be found online. With the growth of the number of datasets available as well as the growth of their size comes the problem of effectively detecting not only the links between the datasets (as studied in previous works [12]) but also of determining the datasets with which a novel dataset should be linked. A naive approach to linking these datasets would choose two datasets and check, whether they can be linked with each other. Such an approach would need $\mathcal{O}(n^2)$ pairwise comparisons of datasets to find possible candidates of linking, which is clearly impracticable. Addressing the problem of finding relevant datasets for linking is however of crucial importance to facilitate the integration of novel datasets into the Linked Data Clouds as well as the discovery of relevant data sources in enterprise Linked Data [12].

[1] http://stats.lod2.eu/.

H. Sack et al. (Eds.): ESWC 2016, LNCS 9678, pp. 3–19, 2016.
DOI: 10.1007/978-3-319-34129-3_1

In this paper, we study the search for similar datasets given an input dataset. In this context, we define two datasets as being similar if they cover the same topics and should thus be linked to each other. In particular, we aim to elucidate the question whether topic modelling (in particular LDA [3]) can be used to improve the search of similar datasets. To address this research question, we present different approaches pertaining to how datasets can be modelled for dataset search. We then compare these different modelling possibilities against the state of the art. Our findings are implemented into Tapioca, a search engine that takes a description of a dataset and searches for topically similar datasets that could be candidates for link discovery. Our engine learns topics of datasets by analysing their ontologies and uses these topics to map datasets to domains in a fuzzy manner. Based on this representation, Tapioca can compare the topic vector of an input dataset to datasets in its index so as to suggest topically similar datasets, which are assumed to be good candidates for linking. Note that we do not study the link discovery problem herein and address exclusively the search for data for linking under the assumption that datasets should be linked if they describe similar topics.

Our contributions are thus as follows:

- We present six combinations of approaches for modelling data in RDF datasets that can be used for dataset search.
- We apply topic modelling to these combinations, compare them with state-of-the-art baselines and show that topic modelling does lead to significant improvements over several baseline methods.
- We provide a gold standard for dataset search and make it available for future research on the topic.

The rest of this paper is structured as follows: In Sect. 2 we present other approaches related to our work. Section 3 introduces Latent Dirichlet Allocation—a model from the probabilistic topic modelling domain. In Sect. 4, our novel approach for a dataset search engine is presented and subsequently evaluated in Sect. 5. Section 6 concludes this paper. More information on Tapioca, the data we used for the evaluation and a demo can be found at http://aksw.org/projects/tapioca.

2 Related Work

Link discovery is a task of central importance when publishing Linked Data [12]. While a large number of approaches have been devised for discovering links between datasets, the task at hand is a precursor of link discovery and can be regarded as a similarity computation task. The usage of document similarities that are based on topic modelling is well known and have been widely studied in previous works, e.g., in [14]. Especially for information retrieval applications, topic modelling has been used for documents containing natural language. Buntime et al. [4] developed an information retrieval system that is based on an hierarchical topic modelling algorithm to retrieve documents topically related to a given query. Lu et al. [10] analysed the effect of topic modelling for information retrieval. Their results show that while its performance is not good for a keyword search, it has a good performance for clustering and classification tasks in which only a coarse matching is needed and training data is sparse. We think that the task of retrieving similar linked datasets matches this task description.

The Semantic Web is already used for information retrieval tasks. For example, Hogan et al. [8] as well as Tummarello et al. [15] published approaches for semantic web search engines retrieving single entities and consolidated information about them given a keyword query. One of the problems that have to be solved for this task is the consolidation of retrieved entities. Since inside different datasets a single entity could have different URIs, the workflow of such a search engine has to have a consolidation step identifying URIs mentioning the same entity. In both approaches two resources are assumed to mention the same entity if (1) they are connected by an `owl:sameAs` property[2] or (2) both resources have an OWL inverse functional property with the same value. The values of such inverse functional properties are typically assumed to be unique, e.g., an e-mail address. This problem is further studied in [7]. These approaches differ from our topical search engine, since they can't be used to identify topical similar datasets for linkage, because the entities must have been already linked—directly or indirectly by inverse functional properties.

The search engine proposed in [15] has an additional consolidation step summarising properties that are assumed to describe the same fact. This summary is created by using the name of the property, i.e., the last part of its URI. Additionally, the authors wrote that they want to use the labels of the properties in a future release of their search engine. This usage of labels or names of properties to decide whether they stand for a similar fact overlaps with our approach to detect topically similar datasets based on the labels of their properties or classes.

Kunze and Auer [9] proposed a search engine for RDF datasets that is mainly based on filters that work similar to a faceted search. For ranking, the authors use a similarity function that comprises different aspects. One of these aspects is called topical aspect and is based on the vocabularies, that are used inside the different datasets. We will use this aspect as a baseline for comparison and explain it in more detail in Sect. 5.1.

Recently, Sleeman et al. [13] proposed an approach to use topic modelling with RDF data. While their work has a similar basis it differs in many ways since it aims at other use cases. Their approach generates a single document for every entity described in a dataset while our approach creates documents that describe a complete dataset. Thus, their documents are based on a different set of triples and on different textual data gathered from the dataset.

3 Latent Dirichlet Allocation

3.1 Overview

Our approach uses Latent Dirichlet Allocation (LDA) to identify the topics of RDF datasets. LDA is a generative model for the creation of natural language documents [3]. This process is based on probabilistic sampling rules [14] and the following assumptions [3]:

[2] `owl` is the abbreviation for http://www.w3.org/2002/07/owl.

- Every topic is defined as a distribution over words ϕ with higher probabilities for words that are essential for the topic.
- A document is a mixture of topics. Thus, it has a distribution over topics θ.

The generation of a corpus based on a given vocabulary as well as the hyper parameters α and β is defined as follows:

1. Create the set of topics T by sampling a distribution over words ϕ for every topic t using a dirichlet distribution and a prior β.
2. Create every single document d of the corpus using the following steps.
   (a) Create a distribution over topics using a dirichlet distribution and the prior α.
   (b) For every word w in the document, choose a topic that creates it by sampling a topic index z from $\theta^{(d)}$.
   (c) Sample a word from the $\phi^{(z)}$ distribution of the topic t_z.

$$\phi^{(z)} \sim Dir(\beta) \quad \theta^{(d)} \sim Dir(\alpha) \quad z \sim Discrete(\theta^{(d)}) \quad w \sim Discrete(\phi^{(z)}) \quad (1)$$

Figure 1 shows the generative model using plate notation and Eq. 1 contains the relations between the elements. It can be seen that only the word tokens w are observable. All other elements are hidden and have to be derived from the observed word tokens. Therefore, several inference algorithms have been developed that try to estimate all hidden distributions [3,6]. In our work, we use an inference algorithm that is based on Gibbs sampling [6]. Additionally, we use hyper parameter optimisation to automatically determine α and β during inference [16]. The inference algorithm generates the topics as distributions over words and the document's distribution over topics.

3.2 Number of Topics

An important parameter of LDA inference is the number of topics. If this number is too low, the topic model is not able to describe the complexity of the training data. If it is too high, one of the model's main assumption, i.e., the orthogonality of the topics, will not hold anymore. Thus, picking a good number of topics has

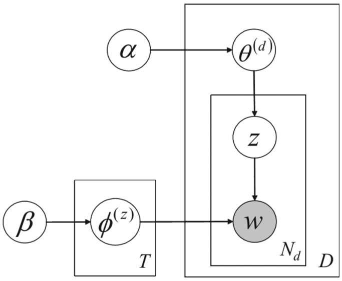

Fig. 1. LDA in plate notation [14].

a high influence on the model's performance. Unfortunately, there is no general applicable method to determine a good number of topics for a given corpus. In this section, we present two different methods that we will apply to our use case during the evaluation. Both methods suggest to determine topic models with different numbers of topics. After the generation, the single models are evaluated regarding their quality using different approaches [1,6].

The first approach is the calculation of $P(\mathbf{w}|T)$ proposed by Griffiths and Steyvers [6]. $\mathbf{w}$ is the set of all tokens present inside the corpus while T is the set of topics of the model. Thus, this probability shows how likely it is that the model could generate the corpus on which it has been trained. Since this probability is intractable, Griffiths and Steyvers presented an approximation by calculating the harmonic mean of a set of $P(\mathbf{w}|\mathbf{z}, T)$ where $\mathbf{z}$ are topic assignments that are sampled from the posterior $P(\mathbf{z}|\mathbf{w}, T)$.

The second approach has been proposed by Arun et al. [1] and is based on the observation that LDA can be regarded as a non-negative matrix factorisation. This factorisation takes the corpus Matrix M of order $|D| \times |V|$ into two matrices M_1 of order $|D| \times |T|$ and M_2 of order $|T| \times |V|$ where D is the set of documents, V is the vocabulary and T is the set of topics. The proposed measure—which we will call $\mathcal{A}$ throughout the paper—is based on the idea that the sum of assignments to the single topics have to be the same in both matrices. But since the rows of both matrices represent probability distributions and are thus normalised, these sums cannot be used directly. Hence, $\mathcal{A}$ is defined as

$$\mathcal{A}(M_1, M_2) = KL(v_1||v_2) + KL(v_2||v_1) \tag{2}$$

where KL is the Kullback-Leibler divergence, v_1 is the distribution of singular values of M_1, $v_2 = L \times M_2$ and L is a vector containing the lengths of the single documents. [1] predicts that with an increasing number of topics the values of $\mathcal{A}$ will decrease until a certain point and start to increase from that point on. They argue that the lowest point inside this dip is created by the model with the best number of topics [1].

4 Our Approach

The goal of Tapioca is to detect topically similar datasets with the aim of supporting the link discovery process. Ergo, given a dataset D and a set of datasets $\mathcal{U} = \{D_1, D_2, \ldots, D_n\}$, our aim is to is to rank the datasets by their likelihood of containing resources that should be linked to resources in D. The basic assumption behind our approach towards this goal is that datasets that should be linked should have similar topics. Hence, we adopt a topic-based modelling of the problem.

The Tapioca search engine comprises three major components:

1. An index that contains known datasets,
2. A way to formulate a query and
3. A method to calculate the similarity between a given query and the indexed datasets.

Of these three, the most challenging component is the definition of *topical similarity* between datasets. A definition of a similarity automatically results in requirements for the indexing and querying components. Therefore, we concentrate on this similarity calculation and present our new probabilistic topic-modelling-based approach. We will use the two example datasets esd-columbia-gorge and esd-south-coast to explain our approach. These examples are derived from real RDF datasets generated from open government data published by the State of Oregon. They contain contracts that have been concluded by different education service districts in 2013. The Listings 1.1 and 1.2 show two example entities of these datatsets.[3]

An RDF dataset contains two types of information that are relevant for our purposes: The first ones are the *instances* that are described inside a dataset. However, instance data is not a good starting point for finding topically similarities between two datasets, since there would have to be at least one instance both datasets have in common. This would be like comparing the two example datasets, i.e., names, titles, keywords and numbers, but without knowing that the data comprises contract data. In such a case, we could only be sure that the two datasets are similar if we were able to find instances that occur in both datasets.

```
@prefix cg: <http://data.oregon.gov/resource/i3bn-rwu4/> .

cg:1
    a cg:Contract ,
    cg:type_of_contract_subcontract  "Material" ,
    cg:esd_name "Columbia Gorge Education Service District" ,
    cg:award_title  "Technology Equipment" ,
    cg:award_type   "Price Agreement" ,
    cg:contractor_name  "TelCompany" ,
    cg:original_start_amendment_date  "03-07-12" ,
    cg:original_award_value  32456.92 ,
    cg:total_award_value_amendments  32456.92 .
```

Listing 1.1. Example entity of the esd-columbia-gorge dataset.

```
@prefix sc: <http://data.oregon.gov/resource/qhct-wumz/> .

sc:1
    a sc:Contract ,
    sc:esd_name "South Coast ESD" ,
    sc:award_title  "Server" ,
    sc:award_type   "Lease" ,
    sc:contractor_information  "computer company" ,
    sc:start_date_expiration_date "7/1/10-6/30/14" ,
    sc:award_amount  5181.87 .
```

Listing 1.2. Example entity of the esd-south-coast dataset.

[3] The original datasets can be found at http://catalog.data.gov/dataset/contracts-esd-columbia-gorge-fiscal-year-2013-c3848 and http://catalog.data.gov/dataset/contracts-esd-south-coast-fiscal-year-2013-3cb8d. For a better explanation of our approach, we made minor changes, e.g., we added two contract classes.

A much more promising approach is to look at the *structure* of the datasets. By doing so, we would know that both datasets contain a class and properties related to contracts. Following these assumptions, our approach is based on extracting this structural metadata from a dataset and transform it into a description of the topically content of the dataset.

Our approach is thus based on three different steps as can be seen in Fig. 2. At first, the metadata of every single dataset is extracted. In the second step, the metadata is used to create a document describing the dataset. In the last step, a topic model is created based on the documents of the datasets. The resulting topic model and distributions enable a similarity calculation between single datasets based on their topic distribution. Additionally, the topic model can be used to determine the topic distribution of documents derived from new, unseen datasets. Thus, our approach is able to handle user input containing datasets that where not known during model inference. The steps underlying Tapioca are explained in more detail in the following subsections.

4.1 Metadata Extraction

Our approach for finding topical similarities between datasets is based on the metadata of these datasets and the RDF and OWL semantics[4] which underlie the Linked Data Web. The metadata comprises the classes and properties used or defined inside a dataset. To every URI of a class or property a frequency count c is assigned, i.e., the number of entities of an extracted class or the number of triples of an extracted property. If a dataset contains metadata, i.e., triples with elements of the VOID Vocabulary, these information are extracted as well. After the extraction, classes and properties of the well-known vocabularies RDF, RDFS, OWL, SKOS and VOID are removed because these vocabularies do not contain any information about the topic of a dataset. Table 1 contains the URIs that would have been extracted from the two example datasets. Note, that the

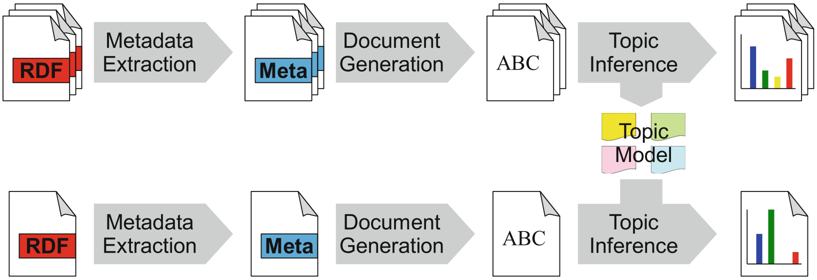

Fig. 2. The single steps of our approach. The upper part shows the index phase in which the topic model is generated while the lower part shows the handling of a query dataset.

[4] https://www.w3.org/RDF/ and http://www.w3.org/TR/owl-semantics/.

Table 1. Example URIs extracted from the two example datasets.

URI	Type
cg:Contract	Class
cg:type_of_contract_subcontract	Property
cg:esd_name	Property
cg:award_title	Property
cg:award_type	Property
cg:contractor_name	Property
cg:original_start_amendment_date	Property
cg:original_award_value	Property
cg:total_award_value_amendments	Property
sc:Contract	Class
sc:esd_name	Property
sc:award_title	Property
sc:award_type	Property
sc:contractor_information	Property
sc:start_date_expiration_date	Property
sc:award_amount	Property

table does not contain the `rdf:type` property, because it has been removed as part of the RDF vocabulary.

4.2 Document Generation

The generation of a document describing a certain dataset is based on the metadata extracted from this dataset. First, URIs and their frequency counts c are selected from the metadata. After that, the labels of the URIs are retrieved. The last step comprises the generation of the document corresponding to the dataset at hand by filtering stop words and determining the frequency of the single words.

There are three different possibilities to use the URIs contained in the metadata of a dataset, leading to three different variants V. Variant V_C uses only the class URIs of the dataset, while V_P uses its property URIs. V_{CP} uses both URI types—classes and properties. Depending on the variant, the URIs and their counts are selected for the next step.

The labels of each of the selected URIs are retrieved and tokenized. This label retrieval is based on the list of URIs that have been identified as label containing properties by Ell et al. [5]. If there are no labels available, the vocabulary part of the URI is removed and the remaining part is used as label. If this generated label is written in camel case or contains symbols like underscores, it is split into multiple words. The derived words inherit the counts c of their URI. If more than one URI created the same word, their counts are summed up.

After generating a list of words all stop words are removed[5]. After that the words are inserted into the document based on their frequency counts. Since LDA uses the bag-of-words assumption only the frequency of the words matters while their order makes no difference. However, using the extracted counts directly could result in large documents, because a dataset can contain millions of triples. Therefore, we tested two different variants to reduce the counts c to a manageable frequency f of a word inside the document. The first variant $V_{\cdot U}$ inserts every word only once, therewith creating a list of unique words with $f = 1$. The second variant $V_{\cdot L}$ uses the logarithm of the counts leading to $f = r(log(c) + 1)$ where r is the rounding function which results the next integer value.

Thus, the whole document generation has six different variants—the product of three different URI selections and two different word frequency definitions. Throughout this paper we will use their abbreviations – V_{CU}, V_{PU} and V_{CPU} for the variants that are using lists of unique words as well as V_{CL}, V_{PL} and V_{CPL} for the logarithm based variants.

At the end of the Document Generation every dataset is represented by a single document. With the variant V_{CPU}, the following two documents would have been created for the two example datasets.

contract type subcontract esd name award title contractor
original start amendment date value amendments

contract esd name award title type contractor information
start date expiration amount

4.3 Topic Model Inference

At this stage of our approach, there is a corpus containing a single document for every dataset. This corpus is used to generate a topic model using the LDA inference algorithm of the Mallet library [11]. The model comprises a distribution over topics for every document of the corpus ($\theta^{(d)}$) and a distribution over words for every topic of the model ($\phi^{(t)}$). The second type of distribution allows the inference of a θ distribution for a new document not contained inside the training corpus.

In our simple example, there might be three topics. While the words *subcontract*, *original*, *amendment*, *amendments* and *value* are marked with the first topic, the second topic could contain the words *information*, *expiration* and *amount*. The third topic contains the remaining words.

[5] The stop word list used can be found at https://github.com/AKSW/topicmodeling/blob/master/topicmodeling.lang/src/main/resources/english.stopwords.

contract type subcontract *esd name award title contractor*
original *start* amendment *date* value amendments

contract esd name award title type contractor information
start date expiration amount

4.4 Similarity Calculation

The similarity of two datasets d_1 and d_2 is defined as the similarity of their topic distributions $\theta^{(d_1)}$ and $\theta^{(d_2)}$. Since the topic distributions can be seen as vectors, we are using the cosine similarity of these vectors [14][6].

$$sim(d_1, d_2) = \frac{\theta^{(d_1)} \cdot \theta^{(d_2)}}{\left|\theta^{(d_1)}\right| \times \left|\theta^{(d_2)}\right|} \tag{3}$$

The `esd-columbia-gorge` document of our example would have $\theta = \{\frac{5}{14}, 0, \frac{9}{14}\}$ while the `esd-south-coast` document has $\theta = \{0, \frac{3}{12}, \frac{9}{12}\}$. Thus, the similarity of our example datasets would be 0.829.

5 Evaluation

The aim of our evaluation was threefold. In the first experiment, we focused on the evaluation of the different possible combinations of the features for topic modelling against several baselines. In our second experiment, we evaluated the two approaches for detecting the best number of topics presented in Sect. 3.2 to test whether they can be applied to dataset search. In the third experiment, we repeated the first two experiments at a larger scale to show that our approach works with larger data as well.

The dataset used for the evaluation is based on RDF datasets that have been indexed by LODStats. We removed those datasets that had no English description or not at least one class URI or one property URI of a vocabulary, that is not filtered out by our approach. The remaining evaluation dataset contained 1680 RDF datasets with 776 213 346 triples.

5.1 Baselines

We compare our approach with three baselines from the field of Information Retrieval as well as the Semantic Web. The first baseline is *tf-idf* [2] for which we extracted the metadata and generate a document for every dataset as described in Sects. 4.1 and 4.2. Let D be the set of known documents and V the vocabulary containing all known words w. Let $tf(d, w)$ be the number of times the

[6] Since we are comparing distributions, it would be possible to use the well-known Jensen-Shannon divergence instead of the cosine. However, during the evaluation of our approach both similarity calculations had a similar performance.

word w occurs inside the document d and let D_w be the set of documents that contain w at least once. Then, a vector can be generated for every document d by calculating a *tf-idf* value for every word w using

$$\textit{tf-idf}(d,w) = tf(d,w) * \textit{idf}(d,w) \quad \text{with} \quad \textit{idf}(d,w) = \log \frac{|D|}{|D_w|}. \tag{4}$$

Since *tf-idf* uses term frequencies and an instantiation of the single words is not needed, we used the pure frequencies instead of the logarithm or unique variant. After generating a vector for every document, the cosine similarity can be calculated.

The second baseline is the topical aspect (BL_T) used by Kunze and Auer [9] as part of their RDF search engine described in Sect. 2. The main idea of this topical aspect is to identify topically similar datasets based on the vocabularies that are used inside the datasets, i.e., the datasets contain URIs of the same vocabularies. Let D be the set of all known datasets and $d_1, d_2 \in D$. Let V be the union of the vocabularies used in d_1 or d_2 and let D_v be the set of all known datasets that are using the vocabulary v. Than, the BL_T is defined as

$$BL_T\,(d_1, d_2) = \sum_{v \in V} w(v) g(d_1, v) g(d_2, v) \tag{5}$$

$$\text{with} \quad w(v) = -\log q(v) \quad \text{and} \quad q(v) = \frac{|D_v|}{|D|}, \tag{6}$$

where $g(d, v)$ is a function that returns 1 if the vocabulary v is used inside the dataset d or 0 otherwise. The weighting function $w(v)$ is inspired by the *idf* term of the *tf-idf* function. Thus, the more datasets are using the vocabulary, the less important it is for the topical similarity and the lower its weighting [9].

The last baseline is using Apache Lucene[7]. The generated documents are indexed using the standard analysis of Lucene, i.e., the documents are tokenized, the tokens are transformed into their lower-cased form and Lucene's stop word filter is applied. For every dataset, its document is used to generate a weighted boolean query containing the words of the documents and their counts as weights. This query is used to retrieve similar documents from the index together with Lucene's similarity score for them.

5.2 Experiment I

For the first experiment, we randomly selected 100 RDF datasets to generate a gold standard. Two researchers independently determined topically similar datasets. For solving this task, they got the description of those datasets as well as the possibility to take a deeper look inside the data itself. The ratings of both researchers were compared and showed an inter-rater agreement of 97.58 %. Cases in which the ratings differed were discussed to compile a final rating. With

[7] http://lucene.apache.org/.

this approach 86 dataset pairs could be identify as topically similar.[8] Table 2 shows the features of the corpora that have been created by the different variants of our approach based on these 100 datasets (3 659 152 triples).

For all six approaches presented above, we calculated the similarities of every dataset to every other dataset using the leave-one-out method: One dataset was used as query while the topic model was trained using the other 99 datasets of the gold standard. The result of this step was a ranked and scored list of corresponding datasets for each of the datasets in our gold standard. We then searched for a similarity threshold that led to a maximal F1-score over all datasets. For every variation of our approach, we run experiments in the range of [2, 200] topics. Since the F1-score of the variant V_{AL} was still rising near 200 topics, we further increased the number of topics for this variant until 500.[9]

The best F1-scores that were achieved by the different variants and the different baselines are shown in Table 3. Based on this data, our approach clearly outperforms all baselines if the document generation is based only on properties

Table 2. Features of the corpora generated by the different variants.

Variant	Words	Tokens
V_{AL}	10 182	252 406
V_{AU}	10 182	34 264
V_{CL}	9 500	239 108
V_{CU}	9 500	32 020
V_{PL}	1 173	14 078
V_{PU}	1 173	2 501

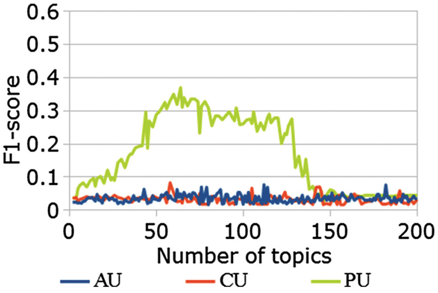

Fig. 3. The F1-scores of the three unique word based variants.

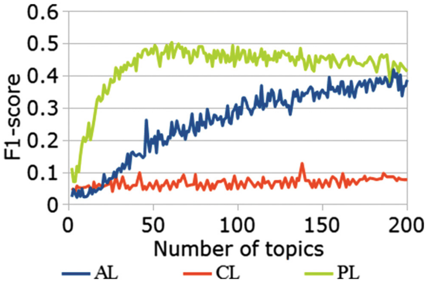

Fig. 4. The F1-scores of the three logarithm based variants.

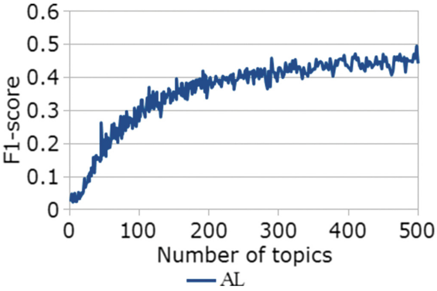

Fig. 5. The F1-scores of V_{AL} for different numbers of topics in the range $[2, 500]$.

[8] The gold standard can be found at the project's web page.

[9] For all topic numbers, the inference was carried out with 1040 iterations, $\alpha = 0.1$, $\beta = 0.01$ and a hyper parameter optimisation after every 50 iterations starting after iteration 200.

Table 3. Best F1-scores achieved by the different variants and the baselines. In the most left column, there are the results for the variants V_{CL}, V_{PL} and V_{AL} while the results of V_{CU}, V_{PU} and V_{AU} are in the second most left column.

URIs used	Tapioca (log.)	Tapioca (unique)	*tf-idf*	BL_T	Lucene
Class	0.128	0.083	0.103	0.292	0.096
Properties	**0.505**	0.350	0.436	0.356	0.418
Both	0.495	0.078	0.444	0.333	0.241

and logarithmic counts. Moreover, our approach performs much better with logarithmic counts than with unique word frequencies. In Fig. 3, we also see that with varying numbers of topics V_{AU} and V_{CU} stay at a low level. Only V_{PU} achieves competitive F1-scores. We think that this has two causes. First, the unique-based variants do not assign a weight to the labels regarding the importance that a class or a property has inside a dataset. Secondly, it has already been shown that LDA does not perform well on short documents in which many different words appear rarely, e.g., messages of short messaging services [17].

Regarding the URIs used for the document creation, it can be seen that all approaches show a poor performance if they are only based on classes. These variants are only able to find similar datasets if the similarity is very obvious, e.g., different eagle-i[10] datasets that are using the same vocabularies. Additionally, they have the drawback, that only 88 out of the 100 datasets define or use classes which makes them unable to calculate similarities for 12 datasets.

Another observation of the experiment is that BL_T does not perform well. Thus, the assumption that topically similar datasets are using the same vocabularies does not hold in reality. One core reason might be that many of the datasets we consider have been generated automatically from tables or CSV files. Every generated dataset has an own, generated vocabulary URI like the two example datasets in Sect. 4.

The Figs. 3, 4 and 5 show the influence of the number of topics on the models performance. For V_{PL}, V_{AL} and V_{PU}, there is a range of numbers of topics in which the F1-score is maximised. Models with too few topics have a much worse performance while—especially for V_{PL} and V_{AL}—the performance deterioration caused by too many topics is rather small. Thus, we can summarise that finding a good number of topics is important for our approach. However, in case an exact number cannot be determined, a high number of topics should be preferred.

5.3 Experiment II

Based on the results of the first experiment, we evaluated whether the two approaches for determining a good number of topics presented in Sect. 3.2 are useful in the present use case. Thus, for the topic range [2,200] we generated topic models using all documents of the gold standard datasets that have been

[10] The gold standard contains datasets of the eagle-i project. https://www.eagle-i.net.

generated by the V_{PL} variant of our approach. For every number of topics we generated five models, calculated $P(\mathbf{w}|T)$ as well as $\mathcal{A}$ and determined the average values of these five runs.

Figure 6 shows the average logarithm of $P(\mathbf{w}|T)$ and reveals that the probability increases steadily with an increasing number of topics. Thus, this method would recommend a much higher number of topics than the 61 topics with which the V_{PL} variant performed best. The average value of $\mathcal{A}$ is shown in Fig. 7. The curve shows a dip as described by Arun et al. [1]. But the minimum value of this dip has been achieved by models with 11 topics with which V_{PL} has only an F1-score of 0.21.

From this experiment, we can summarise that none of these approaches seems to be appropriate to determine a good number of topics for our use case. Therefore, we have to fall back on a simple alternative that we will present during the third experiment.

5.4 Experiment III

To evaluate whether our approach can handle a larger number of datasets, we repeated the first two experiments but trained the model on the complete LOD-

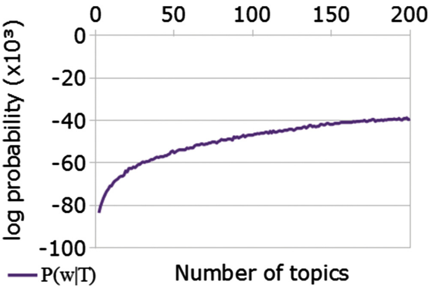

Fig. 6. Average $\log(P(\mathbf{w}|T))$ calculated on the gold standard corpus of the V_{PL} variant.

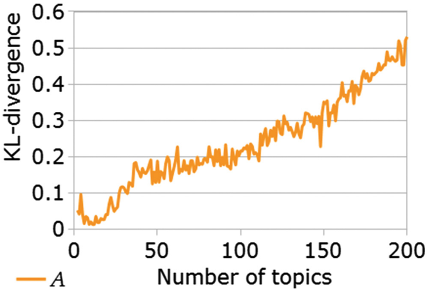

Fig. 7. Average values of $\mathcal{A}$ calculated on the gold standard corpus of the V_{PL} variant.

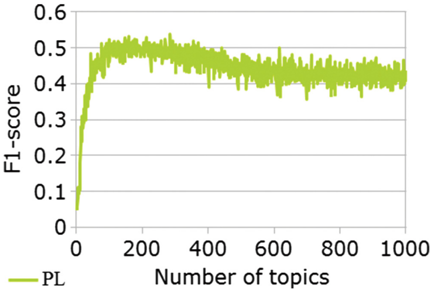

Fig. 8. The F1-scores of V_{PL} calculated on the complete LODStats corpus for different numbers of topics.

Table 4. Best F1-scores achieved by TAPIOCA and the baselines for the complete LODStats corpus.

Approach	Classes	Properties	Both
TAPIOCA (log.)	—	**0.538**	—
tf-idf	0.103	0.436	0.444
BL_T	0.014	0.014	0.014
Lucene	0.214	0.241	0.385

Stats dataset. In detail, this means that for every dataset of the gold standard, we removed it from the set of all 1 680 LODStats datasets. We trained the variant V_{PL} of our approach on the 1 679 remaining datasets and calculated the similarity between the removed dataset and the other 99 datasets contained in the model. After that we compared the similarities with the gold standard and searched the similarity threshold that maximised the F1-score. Using the V_{PL} document creation, the 1 680 documents of the complete LODStats dataset comprise 175 080 tokens of 5 816 different words.

Figure 8 shows the F1-score achieved by V_{PL}. The maximum F1-score of 0.538 was achieved by a model with 284 topics. The results in Table 4 show that even with a much larger input our approach is able to achieve an F1-score that is higher than the scores of the baselines and comparable to the score achieved in the first experiment.

We repeated the calculation of $P(\mathbf{w}|T)$ and $\mathcal{A}$ for the complete LODStats corpus (for the sake of space, we do not show the resulting figures since they are similar to the results of the second experiment). While the average value of $P(\mathbf{w}|T)$ increases steadily with a larger number of topics, the minimum of the average value of $\mathcal{A}$ is at 33 where the F1-score is only 0.336. But since the gold standard is part of the dataset our search engine indexes, we can use it as a pragmatic way to determine a good number of topics. This pragmatic method assumes that a good topic model that has been trained on the datasets of the gold standard and additional datasets should give a high F1-score if it is compared to the gold standard. Thus, in practice we shall train multiple models with different numbers of topics on the same large dataset that comprises the gold standard datasets and use the model that achieves the highest F1-score compared to the gold standard.

6 Conclusion

The aim of this work was to present Tapioca—a search engine that tackles the problem of finding topically similar linked datasets inside the LOD cloud. With this search engine we address the gap between creating an RDF dataset and linking it to other datasets. Our evaluation shows that our approach is better than several baselines and performs well on a large number of datasets. We could identify different parts of a datasets metadata and show that the properties are most important for determining the datasets topic. Additionally, we created a gold standard for this task that can be downloaded from the projects web page[11].

The most challenging future task is the search for a good number of topics that can be used to generate the topic model and that is not bound to the gold standard created by us. Besides this, another challenge is the handling of classes and properties that only have labels in foreign languages instead of English.

[11] http://aksw.org/Projects/tapioca.html.

Additionally, we want to increase the search engine's usability, including a tool with which a user can extract the metadata from its own dataset easily.

References

1. Arun, R., Suresh, V., Veni Madhavan, C.E., Narasimha Murthy, M.N.: On finding the natural number of topics with latent Dirichlet allocation: some observations. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS, vol. 6118, pp. 391–402. Springer, Heidelberg (2010)
2. Baeza Yates, R.A., Neto, B.R.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston (1999)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
4. Buntine, W., Lofstrom, J., Perkio, J., Perttu, S., Poroshin, V., Silander, T., Tirri, H., Tuominen, A., Tuulos, V.: A scalable topic-based open source search engine. In: Proceedings of the WI 2004, pp. 228–234, September 2004
5. Ell, B., Vrandečić, D., Simperl, E.: Labels in the web of data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 162–176. Springer, Heidelberg (2011)
6. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. Nat. Acad. Sci. **101**(suppl. 1), 5228–5235 (2004)
7. Herzig, D.M., Mika, P., Blanco, R., Tran, T.: Federated entity search using on-the-fly consolidation. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013, Part I. LNCS, vol. 8218, pp. 167–183. Springer, Heidelberg (2013)
8. Hogan, A., Harth, A., Umrich, J., Kinsella, S., Polleres, A., Decker, S.: Searching and browsing linked data with swse: the semantic web search engine. Web Semant. Sci. Serv. Agents World Wide Web **9**(4), 365–401 (2011)
9. Kunze, S., Auer, S.: Dataset retrieval. In: IEEE Seventh International Conference on Semantic Computing (ICSC), pp. 1–8, September 2013
10. Lu, Y., Mei, Q., Zhai, C.: Investigating task performance of probabilistic topic models: an empirical study of PLSA and LDA. Inf. Retrieval **14**(2), 178–203 (2011)
11. McCallum, A.K.: Mallet: A machine learning for language toolkit (2002). http://mallet.cs.umass.edu
12. Ngomo, A.-C.N., Auer, S., Lehmann, J., Zaveri, A.: Introduction to linked data and its lifecycle on the web. In: Koubarakis, M., Stamou, G., Stoilos, G., Horrocks, I., Kolaitis, P., Lausen, G., Weikum, G. (eds.) Reasoning Web 2014. LNCS, vol. 8714, pp. 1–99. Springer, Heidelberg (2014)
13. Sleeman, J., Finin, T., Joshi, A.: Topic modeling for rdf graphs. In: 3rd International Workshop on Linked Data for Information Extraction, 14th International Semantic Web Conference (2015)
14. Steyvers, M., Griffiths, T.: Probabilistic topic models. Handb. Latent Semant. Anal. **427**(7), 424–440 (2007)
15. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: live views on the web of data. Web Semant. Sci. Serv. Agents World Wide Web **8**(4), 355–364 (2010)
16. Wallach, H.M., Mimno, D.M., McCallum, A.: Rethinking LDA: why priors matter. In: Advances in Neural Information Processing Systems, vol. 22, pp. 1973–1981 (2009)

17. Zhao, W.X., Jiang, J., Weng, J., He, J., Lim, E.-P., Yan, H., Li, X.: Comparing twitter and traditional media using topic models. In: Clough, P., Foley, C., Gurrin, C., Jones, G.J.F., Kraaij, W., Lee, H., Mudoch, V. (eds.) ECIR 2011. LNCS, vol. 6611, pp. 338–349. Springer, Heidelberg (2011)