Topic Modeling for RDF Graphs

Jennifer Sleeman, Tim Finin, and Anupam Joshi

Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore. MD 21250 USA
{jsleem1,finin,joshi}@cs.umbc.edu

Abstract. Topic models are widely used to thematically describe a collection of text documents and have become an important technique for systems that measure document similarity for classification, clustering, segmentation, entity linking and more. While they have been applied to some non-text domains, their use for semi-structured graph data, such as RDF, has been less explored. We present a framework for applying topic modeling to RDF graph data and describe how it can be used in a number of linked data tasks. Since topic modeling builds abstract topics using the co-occurrence of document terms, sparse documents can be problematic, presenting challenges for RDF data. We outline techniques to overcome this problem and the results of experiments in using them. Finally, we show preliminary results of using Latent Dirichlet Allocation generative topic modeling for several linked data use cases.

1 Introduction

Data presented as RDF triples can be problematic for tasks that involve identifying entities, finding entities that are the same, finding communities of entities and aligning ontological information. Data describing a resource can be sparse, making it harder to distinguish one resource from other similar resources. Data describing a resource can be noisy, having excessive data that is not relevant to the resource. There can be large volumes of data which may contribute to an increase in noise, errors, and ambiguities.

When data originating from multiple sources is used, combining and resolving resource information can be challenging. For example, when aligning attributes from one ontology to another, often there are attributes that simply are not alignable [24]. In this paper, we show how topic modeling can be used to support tasks such as aligning ontologies, recognizing type information, community detection and resolving resources that are the same. Though topic modeling can be challenged by problems related to sparseness and noise, we show ways to overcome these problems.

Topic modeling has quickly become a popular method for modeling large document collections for a variety of natural language processing tasks. Topic modeling is based on statistics of the co-occurrence of terms (typically words) and establishes topics that are groupings of terms to describe documents. It has

been used to describe [2] and classify documents [21, 1], as a feature selection method [8], sentiment analysis [16] and as a tool for clustering things of interest.

Topic modeling is a statistical method that results in abstract categories or topics from the processing of a set of documents. Several methods have been developed for generating topics. Early work by Deerwester et al. [5] introduced the concept of Latent Semantic Analysis (LSA) which uses singular value decomposition for finding the semantic structure of documents to improve indexing and retrieval. Hofmann [13] later used the concept of Probabilistic Latent Semantic Indexing (pLSI) to introduce a probabilistic generative approach. More recent work that has grown in popularity is Latent Dirichlet Allocation (LDA) [3, 2] which is also a probabilistic approach but differs from pLSI by the introduction of a conjugate Dirichlet prior and uses variational and sampling based methods to estimate posterior probabilities. The LDA graphical model is typically conveyed by a plate diagram as can be seen in Figure 1 where W represents the words, $\beta_{1..k}$ are the topics, $\theta_{d,k}$ is the topic proportion of topic k in document D, and $Z_{d,k}$ is the topic assignments.

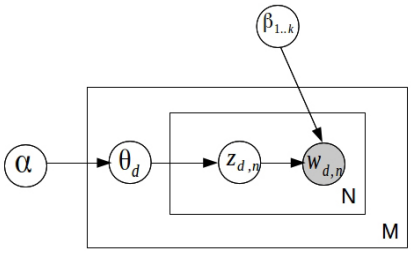

Fig. 1: The Graphical Model for LDA

With LDA, the terms in the collection of documents produce a vocabulary that is then used to generate the latent topics. Documents are treated as a mixture of topics, where a topic is a probability distribution over this set of terms. Each document is then seen as a probability distribution over the set of topics. We can think of the data as coming from a generative process that is defined by the join probability distribution over what is observed and what is hidden [2]. This generative process is defined as follows.

> For each document: (1) Choose a distribution over topics; (2) For each word in the document: (a) select a topic from the document's distribution over topics and (b) select a term from the associated distribution over terms.

The computational portion of LDA involves learning the topic distributions by means of inference. Though there are a number of variational and sampling based methods for performing the inference, Gibbs sampling [12] is frequently used.

We describe how one might use topic modeling for RDF data and explore its application to several problems faced by the Semantic Web community. RDF data is less typical in terms of the documents that are used to create a topic model but since a bag of words is typically used with this model, we will show how RDF data can be used. Topic modeling was originally used to characterize relatively long documents, such as newswire articles or scientific papers. More recently, researchers have outlined successful strategies [28, 15, 21, 22], for using topic modeling for short texts such as tweets and SMS messages. We build on these ideas to establish an approach to using topic modeling with RDF data.

```
dbp:Alan_Turing  dbpo:award       dbp:Order_of_the_British_Empire .
dbp:Alan_Turing  dbpo:birthDate   "1912-06-23+02:00"^^xsd:date .
dbp:Alan_Turing  dbpo:birthPlace  dbp:Paddington .
dbp:Alan_Turing  dbpo:field       dbp:Computer_science .
dbp:Alan_Turing  rdfs:label       "Alan Turing"
dbp:Alan_Turing  rdf:type         dbpo:Scientist .
dbp:Alan_Turing  rdf:type         foaf:Person .

alan turing award birth date birth place field 1912-06-23+02:00 order
of the british empire paddington computer science scientist person
```

Fig. 2: A simple set of triples making up an RDF "document" and the word-like tokens extracted from them for our topic modeling system

There are several issues in applying topic models to short texts [28]. The first is the discriminative problem, where words in short documents do not discriminate as well as in longer ones. The second is that short documents provide much less context than longer ones. RDF data shares both of these and adds a third: none of its serializations are like any natural, human language.

2 Topic models and RDF graphs

2.1 Topic models for text

To give light to these problems we show how topic modeling for text documents differ from RDF documents by describing how topic models are used with text documents and how we apply them to RDF graphs.

Although there are a number of algorithms for defining and using topic models, they share several common aspects. A topic model uses a fixed set of K topics to describe documents in a corpus. K varies with the application and is usually between 100 and 1000. A text documents could be anything from a tweet to a 30-page scientific article, but they typically contain at least several paragraphs of text. The mixture of topics in a document is represented as a vector of real numbers between 0 and 1, where the kth number specifies the amount of topic k that the document exhibits. Using topic vectors makes it easy to define the "semantic" distance between two documents (often using the cosine similarity).

The K topics making up a topic model are not specified in advance, but learned by a statistical process that discovers the *'hidden thematic structure in a document collection'* [2]. This stems from the probability that a word will appear in a document about a different topic, which leads to an effective way to compute the topic model vector for a document given its the bag of words.

One common problem is that many of the automatically induced topics in a model may not correspond to concepts that are easy for people to identify. For topic models over text documents, the best that can be done is to list the most frequent words associated with each topic. This is often sufficient to recognize that topic number 32 has something to do with politics and elections where

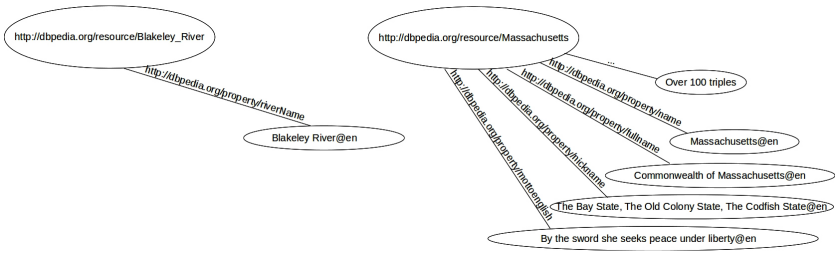

Fig. 3: Small and Large RDF Graphs

as topic number 126 seems to be about software and computer applications. However, there are typically topics that are difficult or impossible to associate with familiar concepts.

Once a topic model has been learned or trained from a document collection, it can be used to infer a document's topic vector from its bag of words. These vectors can then be used for a number of different tasks, such as classifying, clustering or recommending documents.

2.2 Topic models for RDF

While topic models were originally defined for text documents, they have been applied to other kinds of data (e.g., images and generic sequences) and can be used with RDF graphs. To do this, we must define what we will mean by a "document" and the word-like elements within them and also how to compile large collections of those "documents" to train our topic modeling system. For natural language, topic models sometimes exploit linguistic concepts like part-of-speech tags, stop words, and word lemmas and also apply normalization operations (e.g., downcasing, punctuation removal, abbreviation expansion, etc.) to improve performance or accuracy, so we might consider analogs to these notions for RDF data.

What's a document? We assume that a knowledge base is represented by triples, where a triple has a subject s, predicate p, and an object o, forming a $t(s, p, o)$ with the following definitions.

$$s \in (URI \cup Blank), p \in (URI) \text{ and } o \in (URI \cup Blank \cup Literal)$$

We define an 'entity' by $t_1 ... t_n \in T$ associated with a common s URI. In our current model, we treat a document as the set of triples that describe a single 'entity'. We experiment with this definition of a document by working with different parts of the triple, supplementing the triples with additional data, and including 1-hop in-bound and out-bound links.

Alternatively, we could define it as the set of triples in which a given node is either the subject or the object. If we consider a large dataset like DBpedia to be a document collection, we probably want to further restrict the nodes in the

Table 1: Example objects extracted from triples.

Object	Data Source
180(ttp://www.w3.org/2001/XMLSchema#integer	DBpedia
MIT_Building_10_and_the_Great_Dome,_Cambridge_MA.jpg@en	DBpedia
http://dbpedia.org/resource/Moli%C3%A8res,_Tarn-et-Garonne	DBpedia
http://dbpedia.org/resource/Thomas_Walker_(explorer)	DBpedia
http://rdf.freebase.com/ns/m.02hm9j7	Freebase
5dc1e44e:14e30687534:-7f9e	OAEI

graph that we will consider to be documents. A node like *dbp:Alan_Turing* makes a good subject but T-box nodes like *owl:sameAs* or *dbpo:birthDate* probably do not. Similarly, structural nodes such as Freebase's *compound value type* nodes or nodes that link a measurement with a *units* and a *numeric value* may not be suitable subjects for documents.

What's a word? The "words" in a document are extracted from the subjects, predicates and objects of each of its triples and the extractions are treated as bags of words. The words related to an entity (given by a URI) are tokenized by extracting all the triples related to a particular URI from a triple store. Then by removing paths from subjects, predicates and objects. Specifically, literals, i.e., strings, are used in which they are first sanitized with stop words removed. Figure 2 shows an example of a set of triples forming a simple document and its associated word-like tokens.

2.3 RDF Short Text Problem

Short text suffers from two distinct problems: sparseness affects how well the model can discriminate and the lack of context affects word senses [28]. If a word has multiple meanings often context can be used to identify the correct meaning. RDF data also suffers from "unnatural" language since RDF data is represented as triples, the natural structural clues found in human languages are not present.

Sparseness. RDF data can suffer from sparseness. If we choose to think of a document as a set of triples associated with a resource defined by an URI, the set can be large, resulting in a larger, more context enriched bag of words or small, offering very little information at all as shown in Figure 3. Even of the large set of triples, the data that could actually be used in the bag of words, after pre-processing could result in a smaller set of words.

Lack of Context. Context can be particularly problematic for RDF data, as often words are used that can have multiple meanings and due to the potential sparseness of RDF data in addition to the unnatural language characteristic, it could be hard to distinguish that meaning. For example, the description *Alternative rock* contains the word *rock*, without additional context, this word could be interpreted in multiple ways.

Unnatural Language. RDF data suffers from unnatural language issues. Since RDF data is graph-based the natural structure of a sentence does not exist. Often the components of a sentence provide additional context for understanding words which may be polysemous or homonymous. In addition, the text is more prone to error during pre-processing. For example, it is not uncommon to find parts of a triple that have unexpected characters, unusual letter casing, pointers to another resource and data that is simply hard to parse. We show some of these examples in Table 1.

The short text problems in RDF. Researchers tend to take two approaches to overcoming short text related problems. They either supplement the text or they create modified versions of LDA to support their specific problem. We currently use the approach of supplementing text using a set of baseline techniques. Our future work will include additional techniques for supplementing text and a modified LDA algorithm for RDF graphs. We show ways to supplement RDF data in Figure 4. We could simply use the object literals of the triple, for example "University of North Carolina" is an object literal for the resource "University_of_North_Carolina_at_Greensboro". We could also use the predicates, in addition to the object literal. For example, the predicate "http://dbpedia.org/property/name", may be the predicate for the triple with the object literal "University of North Carolina". We may also choose to use Wordnet to supplement the RDF data. For example, for the word "Boston" if we take a subset of synsets and the definition, we enrich the word "Boston" with the following data: [capital of Massachusetts, state capital and largest city of Massachusetts; a major center for banking and financial services, Beantown, Bean Town, Boston, Hub of the Universe]. We also looked at using 1-hop in-links and 1-hop out-links. For example, "Boston" may refer to a mayor, which with 1-hop we could consume the triples related to the mayor of Boston. We could do this similarly with in-links.

3 Related Work

Work by Hong et al. [15] focuses on Twitter data and takes the approach of both developing a modified version of LDA and also defining a number of modeling schemes to be used. They take the approach of inferring a topic mixture for messages and authors where each word in a document is associated with an author latent variable and a topic latent variable. In our work we are not proposing a modification to LDA but rather a way to supplement RDF triples such that the data is better suited for LDA modeling.

Since topic modeling works on the co-occurrences of terms, sparse documents can be problematic. Work by Yan et al. [28] bring light to this problem in terms of 'short text'. As described in this work, often researchers aggregate short text documents or customize the topic model to train on aggregated data. Others make assumptions as to how documents relate to topics. They take the approach of a generative model specifically for 'biterms' which is an 'unordered word-pair

Data Included	***Description***	***Example***
Object Only	Use only the object literals.	<http://dbpedia.org/resource/Berklee_College_of_Music,http://dbpedia.org/property/city, Boston@en> => {Boston}
Object and Predicate	Use the predicate names and object literals.	<http://dbpedia.org/resource/Berklee_College_of_Music,http://dbpedia.org/property/city, Boston@en> => {Boston city}
Object, Predicate and Subject	Use the full triples.	<http://dbpedia.org/resource/Berklee_College_of_Music,http://dbpedia.org/property/city, Boston@en> => {Berklee College Music Boston city}
Object, Predicate and Wordnet Synsets and Definitions	Use predicate name, object literal and the synsets and definitions of both from Wordnet.	<http://dbpedia.org/resource/Berklee_College_of_Music,http://dbpedia.org/property/city, Boston@en> => {Boston capital Massachusetts state capital largest city Massachusetts major center banking financial services Beantown Bean Town Boston Hub Universe city large densely populated urban area include several independent administrative districts metropolis urban center city}
Object, Predicate and Object Outlinks	If there are any non-literals that reference a resource, retrieve the predicates and objects from that resource.	<http://dbpedia.org/resource/Berklee_College_of_Music,http://dbpedia.org/property/state,http://dbpedia.org/resource/Massachusetts> => {state Massachusetts title bay state footer Harvard University MIT widely regards top handful universities worldwide academic research myriad disciplines.....}
Object, Predicate and Subject Inlinks	If the subject is referenced by any other resources, retrieve the predicates and subjects from that resource.	http://dbpedia.org/resource/Berklee_College_of_Music=>{John_Mayer almaMater Trey_Parker almaMater Geoff_Zanelli almaMater Harlan_J._Brothers almaMater The_Click_Five origin}
Object, Predicate, Subject Inlinks and Object Inlinks	Include predicates and objects of resources referring to this resource and resources this resource refers to.	{John_Mayer almaMater Trey_Parker almaMater Geoff_Zanelli almaMater Harlan_J._Brothers almaMater The_Click_Five origin state Massachusetts title bay state footer Harvard University MIT widely regards top handful universities worldwide academic research myriad disciplines.....}

Fig. 4: Bag of Words Variations

co-occurrence'. Again, they specifically address short text by modifying LDA. Though we think this work has merit in this paper we specifically look at how to modify the data itself.

Work by Phan et al. [21] describes how external data sources can supplement short text. They describe a framework for classifying short, sparse text which includes collecting large data sets that are used to create hidden topics. These hidden topics are then used in conjunction with the small data set to support classification. In this approach they were able to address the data sparseness problem and expand their training set to be more representative. This approach differs from ours in that they supplement the short text with large data sets that they apply topic modeling to whereas we supplement the RDF and then apply topic modeling.

Work by Dietz et al. [6] uses topic modeling for bibliographical data and the results are presented as RDF data. However this work does not address the problem of using topic modeling directly on RDF data.

4 Applying Topic Modeling

Given our description of topic modeling and how it could be used with RDF data, we have outlined a number of ways topic modeling could be applied to research tasks within the Semantic Web community.

4.1 Predicting entity types

Often there is a need to associate type information with entities that are defined within RDF data [17, 26, 20]. For example, it is not clear from its name what the resource *'http://dbpedia.org/resource/City_of_Golden_Shadow'* refers to. However, with associated type information, the types *book*, *WrittenWork* and *Creative Work* are associated with the resource. By predicting type information, when type information does not exist, types provide additional information about the entity, supporting tasks such as knowledge base population, entity coreference resolution and entity linking.

We use topic modeling to support entity type recognition by creating a topic model from a sample of data which contains known type information. We use the model to associate topics to the types. Given new data with missing type information, we then infer topics for new entities. Using KL divergence, for each entity with an unknown type, we measure the divergence between its topic vector and the topic vectors of each known type. Based on this measure we assign known types to new entities.

4.2 Entity Disambiguation

The need to match instances across different data sets or to link new instance information with existing knowledge base instances is common [10, 27, 14, 25]. This method usually involves taking information from each instance and applying a matching algorithm to identify which instances are likely the same. Topic modeling supports creating clusters of entities that are closely related, which can be used as a preprocessing step for matching instances or disambiguating entities. In our work, we assume an existing knowledge base and create a topic model from its data. With this, we can compute topic vectors for new entities to be integrated into the knowledge base. We use cosine similarity to compare the new entity topic vectors to vectors for existing KB entities. We treat this approach as a candidate selection method, where the entities that have similar topic vectors should be evaluated for similarity.

4.3 Ontology alignment - Class and Property Alignment

Ontology alignment [23] can include classes, properties and instances. For example, from the OAEI initiative [19] *oaei_101#author*[1] from ontology 1 aligns with *oaei_103#author* from ontology 3. We use a topic model based on one ontology then we infer topic vectors for our second ontology, making our properties and classes our 'entities' of interest. We take the cosine similarity to directly align properties and classes.

[1] We use the abbreviation oaei_101, oaei_103, and oaei_205 for http://oaei.ontologymatching.org/tests/101/onto.rdf, http://oaei.ontologymatching.org/tests/103/onto.rdf and http://oaei.ontologymatching.org/tests/205/onto.rdf, respectively.

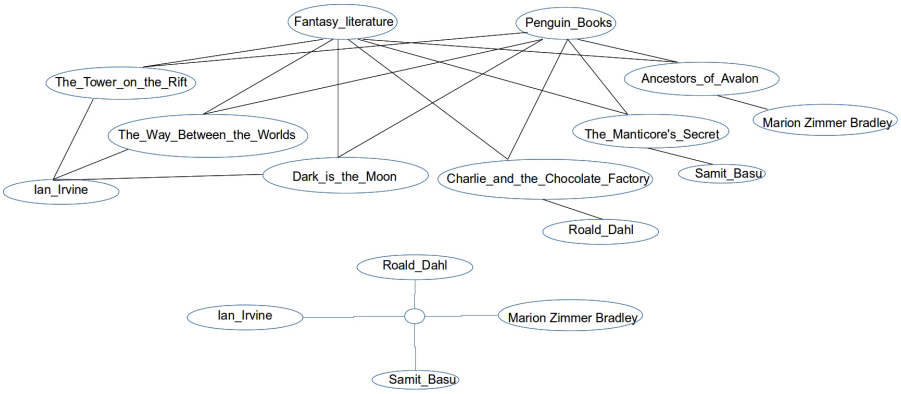

Fig. 5: Example of Fantasy Literature Penguin Publishing Author Community

4.4 Community detection

Community detection approaches [29, 11, 9] can be categorized as topological or topical [7]. We address topical detection in this work. By examining the graphs of RDF data and based on highly connected vertices, communities can be detected by node connections. As an example, we took a set of resources from a DBpedia [4] sample and clustered resources that are fiction literature and produced communities based on sharing the same publishers and the same genre. A community of authors can be seen in Figure 5.

This topical approach can be performed by using topic modeling. For example, authors of the *Fantasy Literature* and *Penguin Publishing* community might have more topics in common than authors of other genres associated with different publishing companies.

In our work we build a topic model from a data set that we identify as having communities of interest. From this model we then associate topic vectors to each entity. We look for entities which have a number n of topics in common. In order to find entities which have n topics in common, we create a histogram from the topic probabilities for each entity. Assuming a topic defines a sub-community, we use the histogram to tell us where the most density is among topics for the entity and set a threshold so that we only consider the topics which are most relevant to the entity. From this we assign entities to topic sub-communities. We then find entities that have n sub-communities in common.

5 Experiments and Results

We experimented with different ways to build a bag of words from the RDF data based on approaches in Figure 4. For each problem which we outlined previously, we used LDA without any modifications to the algorithm itself. Our goal with this work was to show how to supplement RDF data to overcome issues related to spareness, lack of context and the use of unnatural language. We did

see improvement in supplementing the RDF data with repetition of key words, and using a limited set of synsets and definitions from Wordnet. Specifically, when working with large graphs, using the object literals alone may be sufficient but we have observed better results when including either the predicate or the predicate and the subject. We have also found that where the graphs are particularly sparse, using Wordnet [18] synsets and definitions can improve performance. Using 1-hop in-links and 1-hop out-links often increased the noise factor which negatively impacted the performance. We limited the data we incorporated from in-links and out-links to predicates that were of type *name* and *label*. This approach reduced the noise but we didn't see significant improvements in performance. Our future work will include exploring links more, possibly by examining graph similarities.

5.1 Predicting entity types

We used DBpedia data and created two different randomly selected data sets. One was used to actually build the topic model, it had 6000 unique resources. The second data set had 100 unique resources. We associated topics to known types and then used the model to infer types for each entity in the second data set. We then used KL divergence to compare topic vectors. From this we mapped types from the first data set to entities in the second data set. We tested with 200 topics and 400 topics with resources that had an average of seven types that should be recognized. Our ground truth in this case was the type defintions in the DBpedia data set. We removed the type defintions for our test data set then evaluated our predictions with what was defined by DBpedia. Though our test set was relatively small, we were able to see how precision changed based on data variations. As can be seen in Figure 6 we saw the highest precision using predicates and objects and in Figure 7 we saw the highest precision using predicates and objects that included the Wordnet synsets and definitions. Though it was clear that objects alone did not perform as well as including the predicate, future work will further explore the relationship between supplemental data and the number of topics chosen for the model.

5.2 Entity Disambiguation

For this experiment we took a subset of DBpedia data as our knowledge base including 300 unique entities with an average of 19 triples per entity and used this data set to build a topic model. We created a second data set with 100 unique entities obtained from the same data, except we obfuscated the subjects such that subjects could not be directly matched. For example, the unobfuscated subject "Falling_in_Love_with_Jazz" became "Jxiiwhw_wh_Uaka_kwki_Uxhh". We used this approach as to create a ground truth for entity matching. We used a lookup table to correlate between the obfuscated subjects and the unobfuscated subjects to evaluate our approach. We associated topics with each entity in our knowledge base. We then took our obfuscated data set and inferred topics for each entity. From this we used cosine similarity to compare entities and tried to

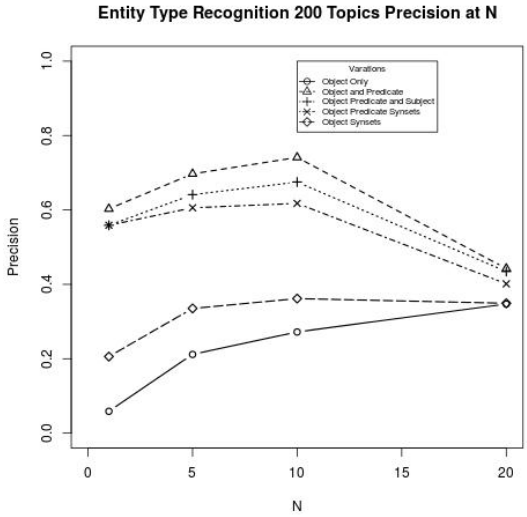

Fig. 6: Entity types with 200 topics

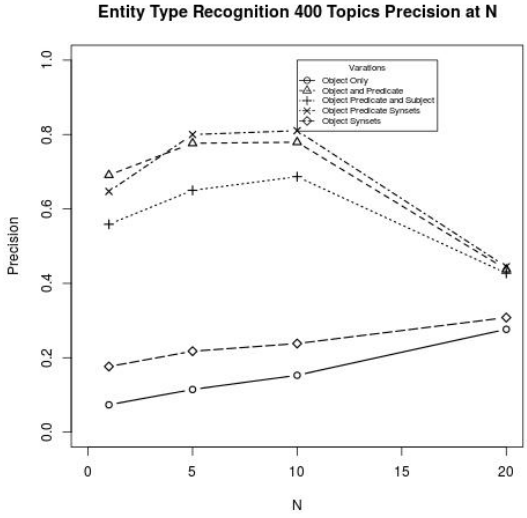

Fig. 7: Entity types with 400 topics

match entities from the two data sets. Though topic modeling is too coarse to use directly to match instances, it does provide a way to significantly reduce the number of candidates that need to be matched. Our experiments showed topic modeling was a reasonable approach for candidate selection reducing on average the number of candidates from 1000s to 100s. However, more work is required to show how this method could be used in conjunction with a entity matching algorithm.

5.3 Ontology alignment

We tested two different alignments that included aligning properties and classes. We used the data from *oaei_101* and *oaei_103*, where the class and property names are identical. We also used the data from *oaei_101* and *oaei_205*, where class and property names to be aligned are not spelled the same. For example, *oaei_101#Booklet* and *oaei_205#Brochure* should be recognized as alignable. We used the OAEI ref alignments to evaluate our approach. This alignment document indicates which properties and classes should be aligned. We excluded instance alignments for this evaluation. We did however extract the instance data only to generate a topic model. Our evaluation examined how well we aligned properties and classes. We tested with 25, 50, 100 and 200 topics and saw the best performance with 50 topics. We exercised the different variations for the RDF data. The ontologies are good examples of sparseness, by using repetition and supplemental data we were able to get approximately 80% precision, where we selected the top N candidates of either an attribute or class match.

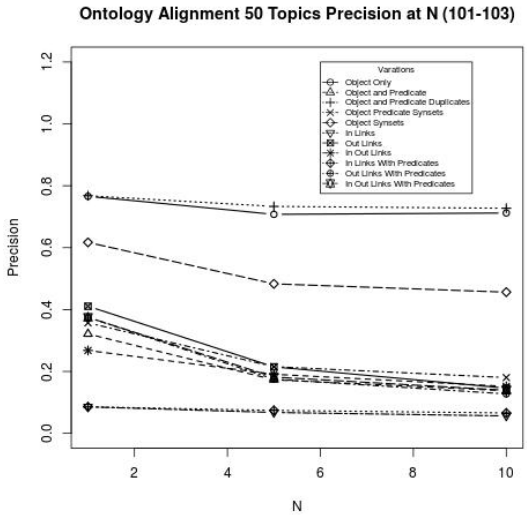

Fig. 8: Ontology alignment (101-103) with 50 topics

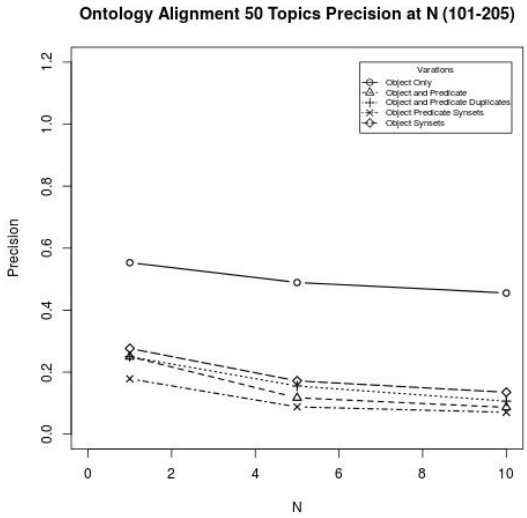

Fig. 9: Ontology alignment (101-205) with 50 topics

5.4 Community detection

We took a sample of the DBpedia data set and performed topic modeling using 50, 100, and 200 topics. From this we associated a set of topics with each entity. We then looked for entities that had n topics in common. Commonality is based on first identifying topics for each entity that are most relevant given their probabilities. then comparing entities based on this subset of topics. Our data set did not include ground truth for this evaluation. However, as seen in figure 10 our preliminary results found interesting communities, such as a community that included *Vini_Lopez* and *Bruce_Springsteen* who are related by playing in the same band. Future work will perform more comprehensive experiments to evaluate this method further.

6 Conclusion

We described a framework for applying topic modeling to RDF graph data and described how it can be used in a number of linked data tasks, including predicting entity types, instance matching, ontology alignment, context identification and community detection. By supplementing RDF data we can address the problems related to spareness, lack of context and unnatural language. We have used different problems in Semantic Web research to exercise LDA modeling. For preliminary results over a small amount of data, topic modeling shows promise for a number of tasks. Repetition and Wordnet supplemental data improves performance. More work is needed to determine how we could use in-links and out-links to supplement the data without increasing the noise. Our results, though preliminary, provide some insight into how a basic LDA model might perform given

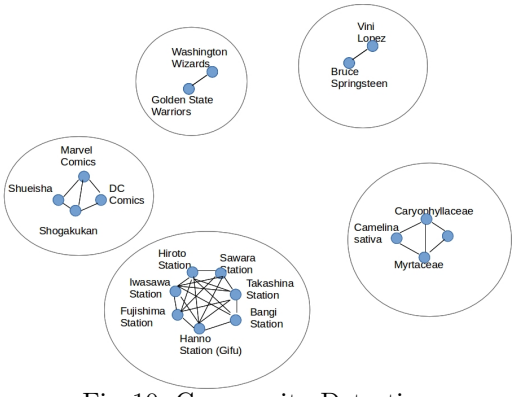

Fig. 10: Community Detection

variations of the text. Our future work looks to modify the LDA algorithm itself to work specifically with graph based data.

Acknowledgment. This work was supported by NSF grants 0910838 and 1228673.

References

1. Bíró, I., Szabó, J., Benczúr, A.A.: Latent dirichlet allocation in web spam filtering. In: 4th int. Workshop on Adversarial Information Retrieval on the Web. pp. 29–32. ACM (2008)
2. Blei, D.M.: Probabilistic topic models. Comm. of the ACM 55(4), 77–84 (2012)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. the Journal of machine Learning research 3, 993–1022 (2003)
4. DBpedia: Dbpedia data set. http://dbpedia.org/Datasets (2011)
5. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. JAsIs 41(6), 391–407 (1990)
6. Dietz, L., Stewart, A.: Utilize probabilistic topic models to enrich knowledge bases. In: Proc. of the ESWC 2006 Workshop on Mastering the Gap: From Information Extraction to Semantic Representation (2006)
7. Ding, Y.: Community detection: Topological vs. topical. Journal of Informetrics 5(4), 498–514 (2011)
8. Duric, A., Song, F.: Feature selection for sentiment analysis based on content and syntax models. Decision Support Systems 53(4), 704–711 (2012)
9. Erétéo, G., Buffa, M., Gandon, F., Grohan, P., Leitzelman, M., Sander, P.: A state of the art on social network analysis and its applications on a semantic web. In: 7th Int. Semantic Web Conference (2008)
10. Ferraram, A., Nikolov, A., Scharffe, F.: Data linking for the semantic web. Semantic Web: Ontology and Knowledge Base Enabled Tools, Services and Applications (2013)
11. Fortunato, S.: Community detection in graphs. Physics Reports 486(3), 75–174 (2010)
12. Griffiths, T.: Gibbs sampling in the generative model of latent dirichlet allocation (2002), http://bit.ly/1IA88Pc

13. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 50–57. ACM (1999)
14. Hogan, A., Zimmermann, A., Umbrich, J., Polleres, A., Decker, S.: Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora. Journal of Web Semantics 10, 76–110 (2012)
15. Hong, L., Davison, B.D.: Empirical study of topic modeling in twitter. In: Proceedings of the First Workshop on Social Media Analytics. pp. 80–88. ACM (2010)
16. Lin, C., He, Y.: Joint sentiment/topic model for sentiment analysis. In: 18th ACM Conf. on Information and Knowledge Management. pp. 375–384 (2009)
17. Ma, Y., Tran, T., Bicer, V.: Typifier: Inferring the type semantics of structured data. In: 29th Int. Conf. on Data Engineering. pp. 206–217. IEEE (2013)
18. Miller, G.: Wordnet: a lexical database for English. CACM 38(11), 39–41 (1995)
19. Ontology alignment evaluation initiative - OAEI 2014 campaign, http://oaei.-ontologymatching.org/2014/
20. Paulheim, H., Bizer, C.: Type inference on noisy rdf data. In: International Semantic Web Conference (2013)
21. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: 17th WWW Conf. pp. 91–100. ACM (2008)
22. Ramage, D., Dumais, S.T., Liebling, D.J.: Characterizing microblogs with topic models. ICWSM 10, 1–1 (2010)
23. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. Knowledge and Data Engineering, IEEE Transactions on 25(1), 158–176 (2013)
24. Sleeman, J., Alonso, R., Li, H., Pope, A., Badia, A.: Opaque attribute alignment. In: Proc. 3rd Int. Workshop on Data Engineering Meets the Semantic Web (2012)
25. Sleeman, J., Finin, T.: Computing FOAF co-reference relations with rules and machine learning. In: 3rd Workshop on Social Data on the Web. ISWC (2010)
26. Sleeman, J., Finin, T., Joshi, A.: Entity type recognition for heterogeneous semantic graphs. In: AI Magazine. vol. 36, pp. 75–86. AAAI Press (March 2105)
27. Song, D., Heflin, J.: Domain-independent entity coreference for linking ontology instances. Journal of Data and Information Quality 4(2), 7 (2013)
28. Yan, X., Guo, J., Lan, Y., Cheng, X.: A biterm topic model for short texts. In: 22nd Int. Conf. on the World Wide Web. pp. 1445–1456 (2013)
29. Zhang, H., Giles, L., Foley, H., Yen, J.: Probabilistic community discovery using hierarchical latent gaussian mixture model. In: AAAI. pp. 663–668 (2007)