Extracting Topics from Semi-structured Data for Enhancing Enterprise Knowledge Graphs

Neda Abolhassani(✉) and Lakshmish Ramaswamy

Department of Computer Science, University of Georgia, Athens, GA 30602, USA
{neda,laks}@cs.uga.edu

Abstract. Unifying information across the organizational data silos that lack documentation, structure and automated semantic discovery has been of an intense interest in the recent years. Enterprise knowledge graph is a common tool of data integration and knowledge discovery and it has become a backbone to APIs that demand access to structured knowledge. A piece which was previously unnoticed in building enterprise knowledge graph, is adding an abstract layer of themes and concepts which is mapped to various documents stored as semi-structured files in databases. Augmenting enterprise knowledge graphs by concepts will help companies to find the trends in their data and get a holistic view over their entire data stores. Extracting topics from semi-structured data suffers from lack of corpus or description as its major challenge. In this research, we investigate the impact of self-supplementation of words and documents on probabilistic topic modeling upon semi-structured data. Another contribution of this paper is finding the best tuning of probabilistic topic modeling that fits semi-structured data. The extracted topics are potential summaries and concepts about the dataset. Moreover, they can be mapped to their sources of origin in order to extend the enterprise knowledge graph. We consider 2 inferencing techniques and demonstrate the results on real data pools from Open City data and Kaggle data containing 7.5 GB and 1.15 GB of data stored in MongoDB collections, respectively. We also propose a selection heuristic for effective identification of topics hidden in various data sources.

Keywords: Topic modeling · Knowledge graphs · Semi-structured data · MongoDB · Gibbs Sampling · Variational Bayes

1 Introduction

In recent years, many organizations have focused on discovering insight from data that is isolated in various sources. The expensive task of knowledge discovery, performed by data experts, faces several key issues. Analyzing the ever increasing and rapidly produced amount of data, particularly the data with heterogeneous structures, is a true bottleneck. Constructing knowledge graphs is a

X. Wang et al. (Eds.): CollaborateCom 2019, LNICST 292, pp. 101–117, 2019.
https://doi.org/10.1007/978-3-030-30146-0_8

potential solution to this problem. A knowledge graph consists of metadata information about the data sources. It holds the relationships and semantics which is hidden in the raw data. In addition, it can capture data governance and lineage information for security and documentation purposes. In terms of structure, knowledge graphs are compatible with RDF data model which has an ontology as its schema. The ontology makes knowledge graphs highly extensible [1].

Integrating information of different files and databases plays a key role in knowledge graph construction. The information integration, summarizing the topics (covered in all the collections), and detecting the documents containing similar themes, make knowledge graph an excellent tool for metadata exploration. Adding these topics to the knowledge graphs is important for monumental tasks such as finding trends, recommending relevant contents, and getting insight on the organization's dataset without reading every document.

Concept extraction and summarization of the growing volume of isolated and semi-structured data stored in files or NoSQL databases -which is the focus of this paper- is essential in identifying and connecting contextually related data elements. These operations, in turn, demand vast amounts of domain knowledge and complex tools. The following challenges are also faced:

- Schema-less nature of semi-structured data or NoSQL databases leads to serious difficulties in detecting logical connections and overlaps among different datasets.
- The data fields has no rich corpus or description and the values are sparse and distributed among different collections as short strings and numbers.
- The data fields usually do not have the natural structure of a sentence and they do not have additional context for understanding words.

For example, consider a data analyst who is interested in analyzing the U.S. cities data stored in various MongoDB collections. MongoDB collections do not have explicit linkage points and they can store sparse and semi-structured contents. There are thousands of documents covering miscellaneous topics such as health care facilities, government jobs, health benefits of employees, school, crime, student loans, local weather archives, etc. Categorizing the dataset to general topics such as education, employment, weather, and health along with connecting the topics to their associated data sources summarizes the dataset contents. The summaries will help the data analyst to get a holistic view about the dataset and find trends in it. In case of the semi-structured data, most of the records contain short strings or numerical values and the short terms do not provide adequate term co-occurrence information. Figure 1 is a sample of a JSON document describing employee wage's data of the city San Jose stored in a MongoDB collection. The same keys are repeated in all documents of the collection and the values are too short to contain enough term co-occurrences. The other collections that are not about the employee wages have also similar keys such as *name* and *department*. This makes it difficult to distinguish between the collection themes.

To the best of our knowledge, this work is the first to consider the challenges of the concept extraction from the semi-structured sparse data. This research

```
 "_id" : ObjectId("599b6b9bc87166670e03b9dc"),
"Name" : "Lee,Alan M",
"Department" : "Police",
"Job Title (as of 12/31/16)" : "Police Sergeant",
"Total Cash Compensation" : "216,251.46",
"Base Pay" : "127,441.60",
"Overtime" : "73,253.61",
"Sick and Vacation Payouts" : "",
"Other Cash Compensation" : "15,556.25",
"Defined Contribution Plan Contributions - City Paid" : "",
"Medical Dental Vision" : "16,430.04",
"Retirement Contributions - City Paid*" : "119,803.25",
"Long Term Disability, Life, Medicare" : "2,970.36",
"Misc Employment Related Costs" : "" }
```


Fig. 1. Example records of cities' data set

evaluates the benefits of the concept extraction over real world data for the purpose of knowledge graph construction. Latent Dirichlet Allocation (LDA) [2] is often used for topic modeling and concept extraction from large corpus of text documents. However in the case of semi-structured data, scanning files and data collections word by word does not lead to effective topic extraction using LDA, because of the lack of enough textual words in the files that convey a specific concept. Therefore, we design various configurations of words and documents related to the semi-structured nature of data. The proposed approach is to augment the base data and emphasize more on descriptive words. Furthermore, we do a comparative study on the designed word-document configurations in order to figure out which one creates the appropriate corpus that leads to a correct topic modeling from semi-structured data. In summary, this paper makes the following contributions:

- Proposes a corpus self-supplementation approach based on the nature of semi-structured files in order to solve the short text topic modeling problem.
- Does a comparative study about applying LDA topic modeling based on four different word and document configurations, two feature extraction methods and two inferencing algorithms in order to find the best tuning that fits the semi-structured data.
- Advocates a simple proposed heuristic topic selection approach in order to choose the best topics from the non-replicable LDA results.

The rest of the paper is organized as follows: In the next section, emerging industrial use cases and related literature are highlighted. In Sect. 3, the preliminary steps of topic extraction from semi-structured data are explained. Section 4, gives a detailed description of the concept extraction procedure. Multiple experiments are run and evaluated in Sect. 5. We conclude the paper with some future works in Sect. 6.

2 Motivation and Related Work

Developments in the industry research is still ongoing in the data integration and knowledge discovery field. According to Gartner [3], knowledge graphs are

included in the hype cycle for blooming technologies. In this section, we cover some emerging use cases and complementary enterprise knowledge graph tools. Since concept extraction and topic modeling is a focus of this paper and we are dealing with the short and sparse data in semi-structured files, a section is devoted to the related works in short text topic modeling.

2.1 Modern Knowledge Graph Tools

There have been a number of recent developments in metadata integration. Google Knowledge Graph is a significant example of linked data knowledge bases that enhances the Web search. It provides a short summary about the searched topic in a structured format along with a list of contextually related websites. Knowledge graphs for Web search engines were the inspiration to solve the data silos integration problem in enterprises. One of these solutions is Google Goods [4]. It is a platform for metadata integration across heterogeneous datasets of Google's data lake. Its main storage is the temporal key-value store, BigTable [5], that keeps the linkages between datasets as a catalog. Another metadata integration system is Ground [6], a UC Berkeley developed open-source data context service, which extracts and manages the metadata over the various versions of data. A recent development is CoreKG [7,8] which offers a Rest API for extracting metadata, enriching the extracted information by providing more features (e.g. synonyms and stems), linking the extracted features to the external knowledge bases, and querying data using available data virtualization tools. WhereHows [9] is another example of enterprise knowledge graphs. It is a LinkedIn product which helps employees to find the answers to questions, such as who owns a workflow, what datasets were aggregated to form a view, when was the last ETL, where is a specific profile data, and how was the dataset originally created. The connections to the backend repository of schedulers and a strict schema context help WhereHows extracts and stores its operational metadata. In another study, we have introduced UMR [10] which is a metadata integration platform providing data reconciliation, single logical view, traceability and data lineage. In UMR, the technical and business metadata is extracted via various profiler tools. These modern tools cover many aspects of hidden knowledge in heterogeneous data sources. A missing piece that we are covering in this paper is adding summaries and concepts about the semi-structured data to the enterprise knowledge graphs.

2.2 Emerging Use Cases

Concept extraction from semi-structured data introduces several use cases in different industries, specifically, the industries that support enterprise knowledge graphs. Concept extraction enables companies, such as Amazon that work in various industries (online shopping, web services, media), to segment their technical data into each of their sections and products. Categorizing technical data, which is declared in semi-structured format via clustering it into distinct topical

groups, provides a logical view about the data and reduces the manual tasks of data experts.

Figure 2 demonstrates an example knowledge graph which integrates document stores containing JSON documents about open city data. The collected data can help an urban planner understand city's problems and conduct city services. There are nodes in the knowledge graph for different collections in databases (e.g. *Police_Reports*) and different data fields along with their data types (e.g. *Location, String*). There are also edges connecting database nodes to data field nodes and data field nodes to other data field nodes that have overlapping information. In order to query this knowledge graph, the user must know the address of each data field and its parent database. A single abstract layer containing the summary of concepts in each data source (e.g. *Crime*) and connecting it to its data source of origin has various benefits. The user can search for her topic of interest, and the knowledge graph will automatically resolve the address of its data source (through the dotted lines shown in Fig. 2). Moreover, the user can get a global view about various data silos and monitor their data trends.

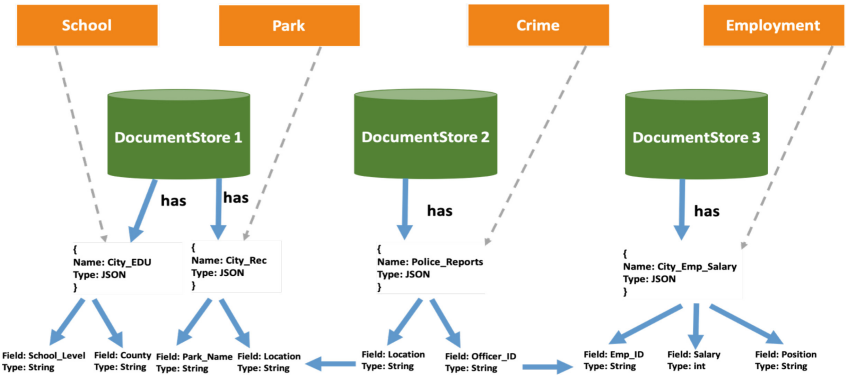


Fig. 2. Example knowledge graphs augmentation with concepts

Strategic planning and data monitoring are other use cases for the topic extraction from semi-structured data that exist in any data driven company. An organization can monitor the state of its semi-structured data and discover the trends in them more effectively using the discovered topics. There is always a demand to track documents that relate to decisions and recommendations in business planning. There are business nodes in an enterprise knowledge graph which should be linked to their correspondent technical documents. Discovering the topics and concepts while keeping track of the data sources that contain those topics connects the business and the technical nodes of an enterprise knowledge graph.

2.3 Short Text Topic Modeling

Existing topic modeling approaches such as, LDA and PLSA [11] have shown great achievements on long texts. However, uncovering the latent topics within the short and sparse texts which do not have a rich corpus and sufficient word co-occurrence is challenging. There are some methodologies proposed in the literature to solve the short text topic modeling problem. In [12] authors clustered twitter messages based on Probase which is Microsoft's probabilistic taxonomy obtained from billions of web pages [13]. They have developed a Baysian inference model to draw out concepts according to instances and attributes detected in Probase. Another conceivable approach is to enrich the short texts with auxiliary long documents [14]. [15] studies how word embedding for vector representation of words based on external knowledge sources like Wikipedia enhances the extraction of latent topics. However, finding relevant external long documents are not always feasible. [16] proposes a biterm topic modeling approach in which the topic assignments are drawn on corpus level distribution instead of the document level and the unordered biterms of a document are sampled from the same topics. Another approach is a two phase topic modeling which provides more autonomous word co-occurrences [17]. In the first phase the short texts are self aggregated into pseudo-documents generated based upon LDA, and in the second phase, a word is sampled based on the probability distribution over the generated pseudo-documents.

The above mentioned short text modeling methodologies were bench-marked over twitter datasets or news titles that are quite different in structure and context than the semi-structured data. The researches in the literature are usually using external long knowledge sources which is not always available for domain specific problems. In this paper, we evaluate LDA topic modeling with different word and document configurations and corpus self-supplementation based on the nature of the semi-structured data over MongoDB datasets while proposing a heuristic topic selection approach.

3 Preliminaries

The Latent Dirichlet Allocation (LDA) topic modeling technique is a generative probabilistic model where documents of a corpus are assumed to be probability distributions over latent topics $p(z|d)$ and topics are assumed to be distributions over words $p(w|z)$ [2]. The model also considers two Dirichlet priors for the per document topic distribution α and the per topic word distribution β. LDA uses posterior distribution to uncover the latent variable z which explains the topic. The following equation shows the LDA posterior distribution of the hidden variables given the words in the documents.

$$p(\phi_{1:K}, \theta_{1:D}, z_{1:D}|w_{1:D}) = \frac{p(\phi_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})} \tag{1}$$

Where K is the number of topics, D is the number of documents, θ is the distribution of topics in document d, and ϕ is the distribution of words in topic

k. Since the denominator of the above equation is not tractable, the solution is to approximate the posterior inference. Two representative inferencing methodologies in this area are Gibbs Sampling [18] and Variational Bayes [19].

Gibbs Sampling finds the posterior approximation based on an empirical distribution using Markov Chain Monte Carlo approach. It starts with random initialization of each word to one of the K topics and it samples a new topic assignment iteratively using the following equation while it assumes that all topic assignments except for the current one are correct.

$$p(z_i = k|w_i = w, z_{-i}, w_{-i}, \alpha, \beta) = \frac{n_{k,-i}^{(d)} + \alpha}{\sum_{k'=1}^{K} n_{k',-i}^{(d)} + K\alpha} \times \frac{v_{w,-i}^{(k)} + \beta}{\sum_{w'=1}^{W} v_{w',-i}^{(k)} + W\beta} \tag{2}$$

Where z_i is word i topic assignment and z_{-i} is all topic assignments to other words. $n_{k,-i}^{(d)}$ is the number of times document d uses topic k excluding the current assignment and $v_{w,-i}^{(k)}$ is the number of times topic k uses word w excluding the current assignment. The above equation finds how much a document likes topic k and how much a topic likes word w in each iteration.

Variational Bayes approximates $p(z|w)$ by a simpler distribution $q(z)$ for which marginalization is tractable. Variational inference turns into an optimization problem which assumes a variational family of distributions over the latent variables $q(z; v)$. It fits the variational parameters v to be close in Kullback Leibler (KL) distance to the exact posterior $KL(q||p)$. It approximates $p(\phi, \theta, z|w, \alpha, \beta)$ with the below equation.

$$q(\phi, \theta, z|\lambda, \gamma, \eta) = \Pi q(\phi_k|\lambda_k)\Pi q(\theta_d|\gamma_d)\Pi q(z_{d,n}|\eta_{d,n}) \tag{3}$$

Where λ, γ, and η are the hidden variables of the variational distribution. For further theoretical overview on Variational Bayes see [19].

Each input document to the LDA can be represented based on its word occurrences. Bag of Words (BoW) model and Term Frequency Inverse Document Frequency (TFIDF) model are two feature extraction methods that convert text documents to vectors representing the frequency of all the distinct words in documents. BoW only considers the term frequencies. However, TFIDF reduces the impact of words with higher frequencies in all documents. Since LDA shows topics as clusters of high ranked words, the association between a word and its data source of origin is necessary in building conceptualized layer of enterprise knowledge graphs. Therefore, another preliminary step to concept extraction is to store the mapping between the words that are ingested to LDA and their associated data sources while creating the feature vectors. The concept abstract layer will be connected to its data source of origin using this association map.

4 Corpus Self Augmentation-Based Topic Extraction

Our framework for concept extraction takes semi-structured data as inputs and returns a set of ranked word clusters that describe the hidden topics existing in

the entire dataset. The two novel aspects of the framework are its corpus creation and topic selection filtering. The corpus creation component results in building the best corpus that fits the semi-structured nature of data for better topic modeling and the topic selection filtering finds the most distinctive topic word clusters. The result concepts can further be visualized using topic visualization tools such as LDAviz [20]. The framework is illustrated in Fig. 3.

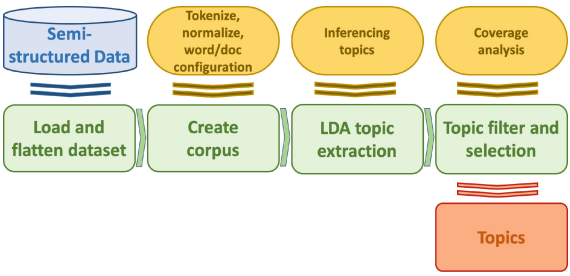


Fig. 3. An overview of the topic extraction framework for semi-structured documents.

Load and Flatten Dataset. This module ingests a variety of semi-structured datasets such as, JSON, CSV, and etc. Documents in a semi-structured file such as a JSON document in MongoDB or a column family in HBase can have nested formats. In a JSON example, a key can hold another set of key value pairs as its value. In this case, flattening the files can help in reducing the complexity of the analysis. In our experiments, we focused on MongoDB databases that contain collections of JSON documents. Therefore, flattening is a mandatory step. For instance, when we have $Key1$ with another document as its value containing two keys called $Key2$ and $Key3$, JSON flattening will result in generating new keys called $Key1.Key2$ and $Key1.Key3$. The new generated keys still keep their unique association to the values in the subsumed document.

Corpus Creation. This component makes different word-document configurations. First, each document of the data sources should be scanned, tokenized, and normalized. Next, the words and documents should be declared based on the contents of the semi-structured file. Table 1 shows the LDA configurations which include 4 different word-document settings. In case of MongoDB data, either a document or a collection can be considered as LDA input document. Attribute names, such as keys in JSON documents can be identified as words all by themselves. This setting is denoted as *Keys-MongoDB Collection* in Table 1. Values subsumed by the keys play a significant role in finding topics from the data. Therefore, in the other settings we makes long pseudo documents based on corpus self-supplementation. We take advantage of adding values to the LDA input words. If a key is appeared in multiple documents, the key is included once besides all the N values associated with it. This setting is denoted as *1Key_NValue-MongoDB Collection*. In this setting, the

semi-structured document is ingested after the normalization without any corpus self-supplementation. Corpus self-supplementation is another word-document configuration which helps in creating long pseudo documents. In this configuration, each value can be accompanied by its key although the same key may appear in the collection multiple times. *1Key_1Value-MongoDB Collection* refers to this setting. The last configuration is *1Key_1Value-MongoDB Document* which takes all the words of each MongoDB document as input words to LDA. Since keys are unique in each document, they appear only once in the *1Key_1Value-MongoDB Document* setting. Finally the corpus of words can be vectorized using BoW or TFIDF models.

Table 1. Summary of 16 different experimental LDA settings. The proposed word-document configurations are designed for the semi-structured data based on MongoDB terminology.

Inferencing technique	Feature extraction	Word-document configuration
Variational Bayes	BoW	Keys - MongoDB Collection
		1Key_Nvalues - MongoDB Collection
		1Key_1Value - MongoDB Document
		1Key_1Value - MongoDB Collections
	TFIDF	Keys - MongoDB Collection
		1Key_Nvalues - MongoDB Collection
		1Key_1Value -MongoDB Document
		1Key_1Value - MongoDB Collections
Gibbs sampling	BoW	Keys - MongoDB Collection
		1Key_Nvalues - MongoDB Collection
		1Key_1Value - MongoDB Document
		1Key_1Value - MongoDB Collections
	TFIDF	Keys - MongoDB Collection
		1Key_Nvalues - MongoDB Collection
		1Key_1Value - MongoDB Document
		1Key_1Value - MongoDB Collections

LDA Topic Extraction. This component runs LDA topic modeling based on two inferencing methods, Gibbs Sampling and Variational Bayes. The inferencing methods are well discussed in Sect. 3.

Topic Filter and Selection. The inferencing methods applied to LDA are based on random initiation. Thus each time LDA is run, it shows different words in the topic's top words cluster. We know intuitively that distinct data sources have higher chance of covering different topics and we want each topic word cluster to be as distinct as possible from the other topics. Therefore, based on

the word-data source map, the number of times each data source appears in a topic's top word cluster can be calculated. We define the coverage filter as the set of topics that have minimum similarity in their associated data sources, i.e.:

$$Coverage = min \sum_{i<j} sim(t_i, t_j) \tag{4}$$

Where each topic $t_i = [C_1, C_2, ..., C_n]$ and C_k is the number of times data source k appears in topic t_i. Here, n is the number of distinct data sources. The coverage filter can be applied to the topic results in two ways. In the first approach, a single LDA run generates a set of top ranked words as topics for the coverage filter. Afterwards, the filter is applied to all runs one by one and the run with the highest coverage is selected. In the second approach, all LDA runs generate sets of top ranked words as topics and then all possible combinations of sets of top ranked words are generated from the extracted topics. Next, the coverage filter will find the combination with the highest coverage. Using the coverage, we filter out the topic word clusters that only select their words from quite a few data sources and keeps the topic word clusters that covers as many data sources as possible.

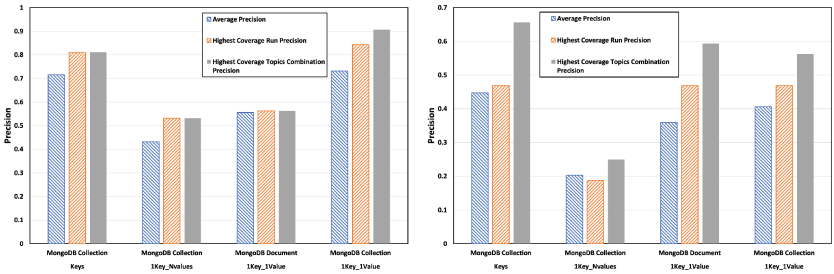
5 Evaluations and Results

In this section, we present the results of running 16 different configurations of LDA on real world data pools of Kaggle data [21] and Open City data. Table 2 shows some statistics over these datasets. Open City data is collected from 18 different cities of the United States. The collected data is about various topics such as, employment, crime, schools, and recreations. The dataset is stored in MongoDB databases as JSON documents and it has a total size of 7.5 GB. The Kaggle data, which is about movie and political elections, includes various JSON files stored in MongoDB databases with the total size of 1.15 GB. We compared the 16 different LDA configurations and the heuristic topic selection using the precision metric while considering the top 8 words in each topic. The precision is represented as follows:

$$Precision = \frac{|top\,relevant\,words\,in\,all\,topics|}{|top\,retrieved\,words\,in\,all\,topics|} \tag{5}$$

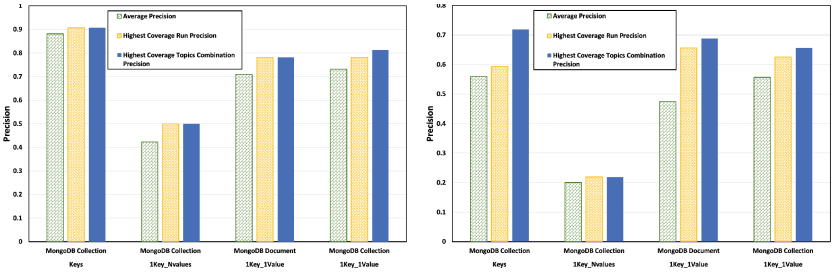
Table 2. Dataset statistics for the experiments

Data source	DB size	# Collections	# Documents
Kaggle data	1.15 GB	12	1,246,310
Open city data	7.5 GB	45	5,823,732



(a) BoW feature extraction (b) TFIDF feature extraction

Fig. 4. Evaluation of the experimental analysis based on Variational Bayes Inferencing on Open City data.



(a) BoW feature extraction (b) TFIDF feature extraction

Fig. 5. Evaluation of the experimental analysis based on Gibbs Sampling Inferencing on Open City data.

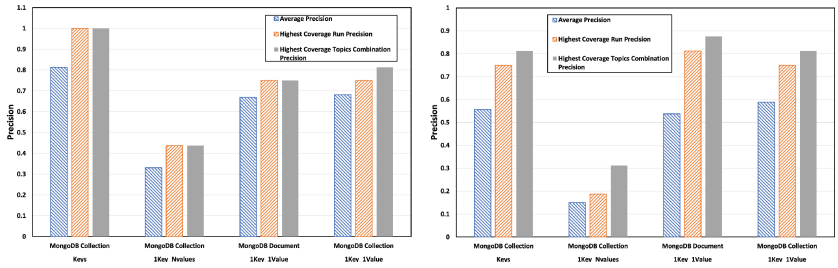
As an alternative evaluation, we compared all of the configurations and the heuristic topic selection using the UMass coherence proposed by Mimno et al. [22]. Given the T top words of a topic z the coherence score is defined as:

$$Coherence_{UMass} = \Sigma_{i=2}^{T}\Sigma_{j=1}^{i-1} log\frac{D(w_i^{(z)}, w_j^{(z)}) + 1}{D(w_j^{(z)})} \tag{6}$$

Where $D(w)$ is the document frequency of term w and $D(w, w')$ is the number of documents containing both words w and w'. Based on the above equation, when the words co-occurred within the same document of the corpus and they belong to the same topic, the coherence score is high. This metric is more intrinsic in nature because it compares words of the original corpus and it does not need any external source of knowledge.

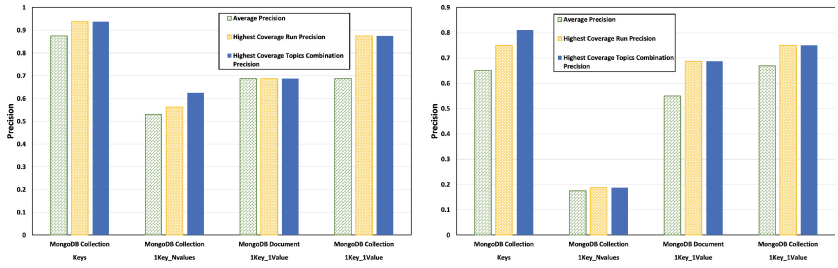
In our experiments, the number of iterations for Gibbs Sampling inferencing is set to 1000. Each LDA configuration is run for 10 times because the applied inferencing methods are based on random initiation. The average precision of the LDA topic modeling based on the Variational Bayes and Gibbs Sampling

with their associated feature extraction and word-document configurations are depicted in Figs. 4, 5, 6, and 7. The precisions of the heuristic topic selection which belong to the run with the highest coverage and the topic combination with the highest coverage are also shown along with the average precision.



(a) BoW feature extraction (b) TFIDF feature extraction

Fig. 6. Evaluation of the experimental analysis based on Variational bayes Inferencing on Kaggle data.



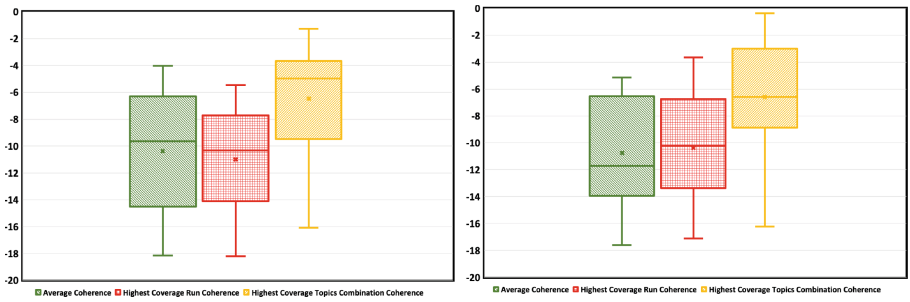
(a) BoW feature extraction (b) TFIDF feature extraction

Fig. 7. Evaluation of the experimental analysis based on Gibbs Sampling Inferencing on Kaggle data.

Keys in JSON files, headers in CSV files, column names in columnar databases, and elements in XML files, summarize the meaning of their subsumed content. The subsumed contents can be numbers, named entities, and short text descriptions. Therefore, corpus self-supplementation, which is accompanying keys with values as the input words of the LDA topic modeling for semi-structured data, shows higher precision in our results compared to the *1Key_NValue-MongoDB Collection* in which the semi-structured document is ingested as it is. The high precision of *Keys - MongoDB Collection* corpus also is a proof of the impact of Keys in the quality of topic extraction. The experiments show lower average precision when key and values both are considered

as words (*1Key_1Value*) and MongoDB documents are considered as LDA documents compared to the case where MongoDB collections are LDA documents with the same word configuration. The reason is considering MongoDB documents as LDA documents results in fewer words in each document of the corpus.

The TFIDF feature extraction reduces the impact of words such as, *id* which occur in most of the collections and have less semantical value. However, the precision of the configurations which include TFIDF approach are lower than the precision of BoW model in all the different setting. TFIDF feature extraction is slightly shrinking the corpus and this can justify the result. The results also show better precision for average Gibbs Sampling inferencing compared to average Variational Bayes. Figures 4, 5, 6, and 7 also demonstrate the precision of our heuristic topic selection approach which is even more than the average precision or very close to it in all of our experiments. This proves that the LDA results which cover heterogeneous data sources define more reasonable topics.



(a) Open City Data (b) Kaggle Data

Fig. 8. Coherence evaluation of the two approaches of implementing topic selection heuristic compared to the average coherence upon the 16 different topic modeling configurations.

Figure 8 shows the coherence result of the two datasets. In both of the datasets, the second approach of implementing the heuristic topic selection which finds the best topic combination from the LDA models generated from all runs demonstrates the best coherence. The minimum (worst) coherence among all the 16 different LDA configurations belongs to the *1Key_NValue*-MongoDB Collection in which the semi-structured document is ingested as it is without any self-supplementation.

Tables 3 and 4 illustrate the top words of LDA topic modeling approach for both datasets. In Open City data, Variational Bayes inferencing and BoW feature extraction along with the *1Key_1Value-MongoDB Collection* corpus self-supplementation resulted in the best topics. The top selected words of each topic can clearly show a distinct concept. Topic 1 can be labled as *school*, topic 2 as *crime*, topic 3 as *recreation/park*, and topic 4 as *employment*. In Kaggle data,

Table 3. Topics learned from the best LDA configuration with the highest data source coverage on Open City data. Configuration: Variational Bayes inferencing, BoW feature extraction, *1Key_1Value - MongoDB Collection*

Topic1	Topic2	Topic3	Topic4
id	id	Park	Total
School	Type	id	id
Address	Address	Objectid	Job
Objectid	Victim	Zone	Contribution
Elementary	Offense	Thegeom	Department
City	Name	Acre	Name
X	Crime	Parkname	Code
Name	Date	Location	Pay

Variational Bayes inferencing and BoW feature extraction along with the *Keys-MongoDB Collection* resulted in the best topics. The top selected words of each topic can again show a distinct concept. Topic 1 can be labled as *movie*, and topic 2 as *election*.

Table 4. Topics learned from the best LDA configuration with the highest data source coverage on Kaggle data. Configuration: Variational Bayes inferencing, BoW feature extraction, *Keys - MongoDB Collection*

Topic1	Topic2
id	Index
Rating	Election
Year	Fair
Runtime	Voter
Genre	Vote
Director	Electoral
Score	Law
Title	Ballot

Table 5 shows precision and elapsed running time of the two of the word-document settings that resulted in the highest precision among the average of 10 runs. This table captures the trade off between running time and precision of the two evaluated inferencing techniques. Although Gibbs Sampling perform with higher quality, it takes longer than the Variational Bayes to run. Table 5 compares only 4 of the experiments. However, Variational Bayes has almost 50% time performance improvement in the average case of all the 16 different settings compared to Gibbs Sampling. According to the results shown in Figs. 4, 5, 6, and 7 and the running time performance, Variational Bayes accompanied

by the BoW feature extraction and the topic selection heuristic filter (the second implementation approach which find the best topics combination) provide the high quality tuning for LDA with more time efficiency.

Table 5. Average running time and average precision of the best performing algorithms for each scenario

Setting	Precision		Elapsed time (Seconds)	
	Kaggle data	Open city data	Kaggle data	Open city data
Variational Bayes Keys MongoDB Collection BoW	81%	71%	0.54	1.94
Gibbs Sampling Keys MongoDB Collection BoW	87%	88%	30.90	30.96
Variational Bayes IKey_IValue MongoDB Collection BoW	68%	73%	8.36	58.02
Gibbs Sampling IKey_IValue MongoDB Collection BoW	68%	73%	33.09	58.62

6 Conclusion and Future Work

In light of the results presented in Sect. 5, we conclude that the LDA model performs with the highest precision (above 80%) for semi-structured data when considering keys only as the words. The second highest precision is provided when using the corpus self-supplementation. It allows more impact on the keys while considering their subsumed values from an entire collection. This configuration accompanied by BoW model has an average precision of 73% on the Open City data. The heuristic coverage filter for topic selection also improves the precision to 90%. Regarding the inferencing techniques, we often see higher precision for Gibbs Sampling. However, the Variational Bayes inferencing is more time efficient and has almost 50% time performance improvement in the average case of all the 16 different settings. The trade off between the precision and the time performance in Variational Bayes and Gibbs Sampling upon semi-structured data match the results of the same techniques upon the unstructured data in the literature. The results also show the same trends on the Kaggle dataset regarding the precision and time performance.

Considering the fact that we have the word-data source mapping as explained in the preliminary step, each topic can be connected to its relevant data sources when we build an enterprise knowledge graph. This work has applications in recommendation systems and data trend discovery. It also allows enterprises to explore and understand their large scale semi-structured data with less effort. Our research in finding optimal concept extraction approach for semi-structured data has been very promising and can add another level of knowledge to the knowledge graph systems. However, an engaging direction for future work is adding state-of-the-art automatic topic labeling to our framework for describing the cluster of words selected as concept representatives in a more extensive way.

Acknowledgments. The authors would like to thank Dr. Pouya Asrar and Kamal Shadi for their valuable feedback in this project. This research has been partially funded by the National Science Foundation (NSF) under grants CCF-1442672 and SCC-1637277 and gifts from Accenture Research Labs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or other funding agencies and companies mentioned above.

References

1. Pan, J.Z., Vetere, G., Gómez-Pérez, J.M., Wu, H.: Exploiting Linked Data and Knowledge Graphs in Large Organisations. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-45654-6
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
3. Hype cycle for emerging technologies. https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/. Accessed 22 Oct 2018
4. Halevy, A.Y., et al.: Managing Google's data lake: an overview of the goods system. IEEE Data Eng. Bull. **39**(3), 5–14 (2016)
5. Chang, F., et al.: Bigtable: a distributed storage system for structured data. ACM Trans. Comput. Syst. (TOCS) **26**(2), 4 (2008)
6. Hellerstein, J.M., et al.: Ground: a data context service. In: CIDR (2017)
7. Beheshti, A., Benatallah, B., Nouri, R., Tabebordbar, A.: CoreKG: a knowledge lake service. Proce. VLDB Endowment **11**(12), 1942–1945 (2018)
8. Beheshti, A., Benatallah, B., Nouri, R., Chhieng, V.M., Xiong, H., Zhao, X.: CoreDB: a data lake service. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 2451–2454. ACM (2017)
9. Data discovery and lineage for big data ecosystem. https://github.com/linkedin/WhereHows. Accessed 22 Jan 2018
10. Abolhassani, N., et al.: Universal metadata repository: integrating data profiles across an organization. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI), pp. 452–459. IEEE (2018)
11. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, pp. 289–296. Morgan Kaufmann Publishers Inc. (1999)
12. Song, Y., Wang, H., Wang, Z., Li, H., Chen, W.: Short text conceptualization using a probabilistic knowledgebase. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence-Volume Volume Three, pp. 2330–2336. AAAI Press (2011)
13. Wu, W., Li, H., Wang, H., Zhu, K.Q.: Probase: a probabilistic taxonomy for text understanding. In: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, pp. 481–492. ACM (2012)
14. Jin, O., Liu, N.N., Zhao, K., Yu, Y., Yang, Q.: Transferring topical knowledge from auxiliary long texts for short text clustering. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, pp. 775–784. ACM (2011)

15. Qiang, J., Chen, P., Wang, T., Wu, X.: Topic modeling over short texts by incorporating word embeddings. In: Kim, J., Shim, K., Cao, L., Lee, J.-G., Lin, X., Moon, Y.-S. (eds.) PAKDD 2017. LNCS (LNAI), vol. 10235, pp. 363–374. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57529-2_29
16. Yan, X., Guo, J., Lan, Y., Cheng, X.: A biterm topic model for short texts. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1445–1456. ACM (2013)
17. Quan, X., Kit, C., Ge, Y., Pan, S.J.: Short and sparse text topic modeling via self-aggregation. In: IJCAI, pp. 2270–2276 (2015)
18. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proc. Nat. Acad. Sci. **101**(suppl 1), 5228–5235 (2004)
19. Hoffman, M., Bach, F.R., Blei, D.M.: Online learning for latent Dirichlet allocation. In: Advances in Neural Information Processing Systems, pp. 856–864 (2010)
20. Sievert, C., Shirley, K.: LDAvis: a method for visualizing and interpreting topics. In: Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces, pp. 63–70 (2014)
21. Kaggle open datasets. https://www.kaggle.com/datasets/. Accessed 02 Oct 2019
22. Mimno, D., Wallach, H.M., Talley, E., Leenders, M., McCallum, A.: Optimizing semantic coherence in topic models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 262–272. Association for Computational Linguistics (2011)