Perplexity of Index Models over Evolving Linked Data

Thomas Gottron[1], Christian Gottron[2]

[1] Institute for Web Science and Technologies, University of Koblenz-Landau, Germany
gottron@uni-koblenz.de
[2] Multimedia Communications Lab, Technische Universitä Darmstadt, Germany
Christian.Gottron@kom.tu-darmstadt.de

Abstract In this paper we analyse the sensitivity of twelve prototypical Linked Data index models towards evolving data. Thus, we consider the reliability and accuracy of results obtained from an index in scenarios where the original data has changed after having been indexed. Our analysis is based on empirical observations over real world data covering a time span of more than one year. The quality of the index models is evaluated w.r.t. their ability to give reliable estimations of the distribution of the indexed data. To this end we use metrics such as perplexity, cross-entropy and Kullback-Leibler divergence. Our experiments show that all considered index models are affected by the evolution of data, but to different degrees and in different ways. We also make the interesting observation that index models based on schema information seem to be relatively stable for estimating densities even if the schema elements diverge a lot.

Keywords: #eswc2014GottronTC

1 Introduction

Thanks to the Linked Data movement, the Web of Data has reached a point where billions and billions of RDF statements are publicly available. Many applications leverage this valuable resource of information and consume, analyse, present, interlink or produce new data on the Web. The fact that Linked Open Data (LOD) is provided in a distributed fashion across many different data sources motivates the need for approaches to index and cache data on the Web. These indices are used, for instance, to facilitate a fast lookup of data sources on the Web, which provide data with certain characteristics, to federate distributed queries, to estimate result set sizes or for caching data locally for faster access.

Along with the lively growth of the Web of Data comes a certain degree of development, maintenance and, thus, dynamics of the data. Data is not anymore just published in a static fashion, but more often and more frequently, data is updated, extended, revised and refactored. Recent investigations in this direction revealed substantial fluctuations and dynamics of LOD under various aspects and for various data sources [10,1].

These changes in evolving Linked Data are a challenge to index structures. Indices become outdated and—as a consequence—might provide incomplete or wrong information. So far, the sensitivity of index models towards evolving data has not been analysed. Therefore, in this paper we will take a first step in this direction, propose an application independent evaluation approach and perform an empirical survey of twelve

different prototypical indexing models and how they behave over evolving Linked Data. The index models are taken from related work and cover a substantial part of different application scenarios, methods and index granularities.

Our analysis is based on an empirical evaluation using real world data of weekly snapshots of Linked Data over a period of more than one year. We use implementations of index models to estimate densities for the distribution of the data. Density estimates are central to several applications, e.g. result set size estimation [12], query optimisation for federated querying [2], information theoretic analysis of LOD [4] or statistical schema induction [16]. Furthermore, densities have the advantage of being application and domain independent. As such they are suitable for a generic evaluation of index models and their sensitivity towards evolving data. For measuring the divergence between a distribution estimated over an index and the distribution estimated from data which has evolved since the creation of the index, we use metrics like perplexity, cross entropy and Kullback-Leibler divergence as well as the Jaccard-similarity over the set of index key elements. Based on these metrics we can see how stable the index models are over evolving data. While all index models are affected by changes in the data, we also make the interesting observation that index structures based on schema information seem to be relatively stable for estimating densities. This is surprising given that the underlying schema information is diverging a lot.

The rest of the paper is structured as follows. We start with a survey and unified formalisation of index models in Section 2. Subsequently we describe how we estimate densities of data distributions from these indices. In Section 4 we present our evaluation: the experimental setup, the applied metrics as well as an overview and discussion of the results. Thereafter, we survey research that is related to our work before we conclude the paper in Section 6.

2 Index Models for Linked Data

In recent years, various index models over LOD have been proposed. Many of them focus on specific aspects of the data or are dedicated to support application specific tasks. The index models in this paper have been selected on the basis of covering a wide range of methods and different levels of granularity. Before going into the details of the models we will briefly introduce a formalisation framework that helps us to describe index models in a unified, application and implementation independent way.

In the context of LOD, we assume data to be available in the form of RDF triples and to be spread across different data sources. Thus, we can assume the atomic data items to be in the form of quads (s, p, o, c) where s, p, and o correspond to the subject, predicate and object of the RDF triple statement and c provides the context, i.e. the data source on the Web where this statement was published. Formally, we consider the sets of all possible URIs U, blank nodes B and literals L. Then in a quad (s, p, o, c) the subject $s \in U \cup B$ can be a URI or a blank node, the predicate $p \in U$ a URI, the object $o \in U \cup B \cup L$ a URI, a blank node or a literal and the context $c \in U$ a URI.

Thus, we assume an index model for LOD to operate on a data set R of (s, p, o, c) quads. Depending on the application scenario, the index model will typically not serve to store all information of the quads. Rather it will define a derived set D of managed

data items which typically constitutes a constraint of the quads to smaller tuples. In this paper we consider two different types of data item sets: (1) the set $D_{SPO} := \{(s, p, o) \mid \exists c : (s, p, o, c) \in R\}$ of full RDF triples and (2) the set $D_{USU} := \{s \mid \exists p, o, c : (s, p, o, c) \in R\}$ of all unique subject URIs (USUs). The data items in D_{SPO} are typically used in a context where an index is based on using a part of the quad information (e.g. the object) to look up matching triple statements (e.g. the subject or the predicate). D_{USU}, instead is typically used in index models where information from several statements (e.g. a set of several RDF types) is used to look up a specific entity URI.

Furthermore, an index model has to define a set $\mathcal{K}$ of key elements which are used to lookup and retrieve data items. These key elements are used as domain for a selection function $\sigma : \mathcal{K} \rightarrow \mathcal{P}(D)$ to select a subset of the data items in the index. In the context of this paper we consider only data structures for which the selection function σ is operating solely on information provided by the Linked Data set R and does not make use of external information or additional meta data (e.g. provenance information).

Accordingly we define an abstract index model as a tuple $(D, \mathcal{K}, \sigma)$ of the stored data items D, the key elements $\mathcal{K}$ used for the lookup index and the selection function σ to retrieve data from the index.

2.1 Triple Based Indexing

A very common approach for indexing RDF data is to use the three entries in the triples, i.e. the URIs filling the subject, predicate and object positions in the RDF statements. Such indices can be used to retrieve all statements affecting an entity in a subject or object position as well as all statements expressing a certain relation. Implementations of this index model can be found in RDF data stores as well as in a combined fashion such as the QTree index [8]. As data items we assume the full triple to be of interest, thus, we use D_{SPO}.

Subject Index: $I_S := (D_{SPO}, \mathcal{K}_S, \sigma_S)$, where:
- Key elements: $\mathcal{K}_S := \{s \in U \mid \exists p, o, c : (s, p, o, c) \in R\}$
- Selection function: $\sigma_S(k) := \{(s, p, o) \mid s = k\}$

Predicate Index: $I_P := (D_{SPO}, \mathcal{K}_P, \sigma_P)$, where:
- Key elements: $\mathcal{K}_P := \{p \in U \mid \exists s, o, c : (s, p, o, c) \in R\}$
- Selection function: $\sigma_P(k) := \{(s, p, o) \mid p = k\}$

Object Index: $I_O := (D_{SPO}, \mathcal{K}_O, \sigma_O)$, where:
- Key elements: $\mathcal{K}_O := \{o \in U \mid \exists s, p, c : (s, p, o, c) \in R\}$
- Selection function: $\sigma_O(k) := \{(s, p, o) \mid o = k\}$

2.2 Index Models Based on Meta Data (Keywords, Source)

Another common type of index makes use of meta data, e.g. the context or textual descriptions and labels used in triple statements. The textual descriptions for entities provide easy to understand descriptions which help human users to interpret the data. Accordingly, index models based on this information are of particular interest for HCI

scenarios. Index structures in this context hardly use a full literal as key elements for indexing, but rather apply term based relevance scores and retrieval methods. Thus, the key elements are terms w taken from a vocabulary V_R of observed words in the literal values of RDF statements in R. To obtain realistic indices we apply common techniques from the field of Information Retrieval, such as case folding and stemming. As queries we assume single term queries, which form the basis for more complex and combined queries in a typical Information Retrieval setting.

Keyword Index: $I_{keyword} := (D_{SPO}, \mathcal{K}_{keyword}, \sigma_{keyword})$, where:
- Key elements: $\mathcal{K}_{keyword} := \{w \mid w \in V_R\}$
- Selection function: $\sigma_{keyword}(k) := \{(s,p,o) \mid o \in L \wedge k \textit{ contained in } o\}$

Index models focusing on the context in the quads, i.e. the source providing the information on the Web, are geared more towards settings where the provenance of a statement is of importance (e.g. for evaluating the credibility of information or to be able to consult the original publisher).

Context Index: $I_C := (D_{SPO}, \mathcal{K}_C, \sigma_C)$, where:
- Key elements: $\mathcal{K}_C := \{c \mid \exists s,p,o : (s,p,o,c) \in R\}$
- Selection function: $\sigma_C(k) := \{(s,p,o) \mid (s,p,o,k) \in R\}$

In this setting, rather than considering a concrete context URI, data is sometimes aggregated per pay level domain (PLD)[1] to better reflect the authorities behind the published data. Such an aggregation of data is used in various contexts, among which Linked Data analysis, e.g. in [10]. Defining the function *pubSuffix* to provide the PLD for a given URI, a PLD level index can be defined as follows:

PLD Index: $I_{PLD} := (D_{SPO}, \mathcal{K}_{PLD}, \sigma_{PLD})$, where:
- Key elements: $\mathcal{K}_{PLD} := \{\textit{pubSuffix}(c) \mid \exists s,p,o : (s,p,o,c) \in R\}$
- Selection function: $\sigma_{PLD}(k) := \{(s,p,o) \mid (s,p,o,c) \in R \wedge \textit{pubSuffix}(c) = k\}$

2.3 Schema Level Indexing

Several index models use schema level information to organise RDF data. Most of these models use joint information from several triple statements to provide a schema level description for entities. These descriptions then serve as key elements. Thus, we will use the set of USUs D_{USU} as data elements for these index models.

The assignment of classes to entities is of relevance in many contexts. It allows for specifying a categorial identification of entities and is used in various applications. As key elements both of the following is considered: URIs used as objects in statements with an rdf:type predicate as well as URIs which are specifically modelled to be a class

[1] The pay level domain consist of the public suffix (e.g. `.org`, `.co.uk`) and the preceding domain name. It represents the level at which Internet users can directly register names and is thus a good estimate for an authority responsible for Linked Data. We obtained the list of public suffixes from `http://publicsuffix.org/list/`.

themselves.

RDF Type Index: $I_T := (D_{USU}, \mathcal{K}_T, \sigma_T)$, where:
- Key elements: $\mathcal{K}_T := \{o \mid \exists s, c : (s, \mathsf{rdf{:}type}, o, c) \in R\} \cup \{s \mid \exists c : (s, \mathsf{rdf{:}type}, \mathsf{rdfs{:}Class}, c) \in R\}$
- Selection function: $\sigma_T(k) := \{s \mid \exists c : (s, \mathsf{rdf{:}type}, k, c) \in R\}$

A variation to this index model is to form groups of types which jointly describe an entity [9]. These *type sets* characterise an entity more specific and precise.

RDF Type Set (TS) Index: $I_{TS} := (D_{USU}, \mathcal{K}_{TS}, \sigma_{TS})$, where:
- Key elements: $\mathcal{K}_{TS} := \mathcal{P}(\mathcal{K}_T)$
- Selection function: $\sigma_{TS}(k) := \{s \mid (\forall t \in k : (\exists c : (s, \mathsf{rdf{:}type}, t, c) \in R)) \wedge \forall (s, \mathsf{rdf{:}type}, o, c) \in R : (o \in k)\}$

An analogous extension to indexing individual predicates is to consider the set of predicates used to describe the properties of a specific entity. This *property set* (sometimes also referred to as *characteristic set*) provides a more specific description of the entity. Such index models are used for accurate result set size estimations when querying distributed RDF data stores [12] or for generating SPARQL queries to feed into federated querying testbeds [3].

Property Set (PS) Index: $I_{PS} := (D_{USU}, \mathcal{K}_{PS}, \sigma_{PS})$, where:
- Key elements: $\mathcal{K}_{PS} := \mathcal{P}(\mathcal{K}_p)$
- Selection function: $\sigma_{PS}(k) := \{s \mid (\forall p \in k : (\exists o, c : (s, p, o, c) \in R)) \wedge \forall (s, p, o, c) \in R : (p \in k)\}$

In the context of statistic schema induction [16] also the set of incoming properties is considered. This corresponds to the set of predicates which all affect the same object in RDF triple statements.

Incoming Property Set (IPS) Index: $I_{IPS} := (D_{USU}, \mathcal{K}_{IPS}, \sigma_{IPS})$, where:
- Key elements: $\mathcal{K}_{IPS} := \mathcal{P}(\mathcal{K}_P)$
- Selection function: $\sigma_{IPS}(k) := \{o \mid (\forall p \in k : (\exists s, c : (s, p, o, c) \in R)) \wedge \forall (s, p, o, c) \in R : (p \in k)\}$

The combination of property sets and type sets leads to the definition of *extended characteristic sets* (ECS). ECS based index models constitute a more extensive approach to schema level indexing as they combine both type and property information. Such index models have been used for measuring redundancy in schema information on the LOD cloud [4,5] and for measuring dynamics of LOD on a schema level [1].

Extended Characteristic Set (ECS) Index: $I_{ECS} := (D_{USU}, \mathcal{K}_{ECS}, \sigma_{ECS})$, where:
- Key elements: $\mathcal{K}_{ECS} := \mathcal{P}(\mathcal{K}_P \cup \mathcal{K}_T)$
- Selection function: $\sigma_{ECS}(k) := \{s \mid (\forall p \in k \cap \mathcal{K}_P : (s \in \sigma_{PS}(p))) \wedge (\forall t \in k \cap \mathcal{K}_T : (s \in \sigma_{TS}(t)))\}$

While ECS based index models already address a lot of the schema information encoded in LOD, there exist models which make use of yet more complex and more fine

grained structures to capture schema information. SchemEX as a schema level index for querying distributed Linked Data falls into this category [9,7].

SchemEX Index: $I_{SchemEX} := (D_{USU}, \mathcal{K}_{SchemEX}, \sigma_{SchemEX})$, where:

- Key elements: $\mathcal{K}_{SchemEX} := \mathcal{P}(\mathcal{K}_{TS} \times \mathcal{P}(\mathcal{K}_{PS} \times \mathcal{K}_{TS}))$
- Selection function: $\sigma_{SchemEX}(k = (ts, E)) := \{s \mid s \in \sigma_{TS}(ts) \wedge \forall (ps, ts_2) \in E : (s \in \sigma_{PS}(ps) \wedge \exists o \in \sigma_{TS}(ts_2) : (\forall p \in ps : (\exists c : (s, p, o, c) \in R))\}$

3 Index Based Estimates for the Distribution of Data Elements

We implemented[2] the index models presented in Section 2 to estimate density functions over key elements. This means, we estimate how probable it is for an element to belong to one specific index key k and—conversely—the amount of data obtained when querying the index for this key element k.

Depending on the type of index model, we look up the distribution of triples or USUs over the index structure. If we consider the distribution over an index $I = (D, \mathcal{K}, \sigma)$, this effectively corresponds to modelling a random variable X taking values of the key elements $\mathcal{K}$. The density we estimate is the distribution of this random variable X. This means we need to determine the probability $P(X = k)$ for each entry $k \in \mathcal{K}$ to be associated with a data item. To estimate the densities we use the count information of data elements associated with the key elements in an index. This corresponds to using a maximum likelihood estimation to derive a distribution, i.e.

$$P(X = k) = \frac{|\sigma(k)|}{\sum_{k' \in \mathcal{K}} |\sigma(k')|} \tag{1}$$

where $\sigma(k)$ indicates the result set obtained from an index when querying for a specific key element k.

As we are considering evolving data it is highly likely that the set of key elements is not stable but evolving itself. Thus, it can happen that certain key elements will disappear as the data evolves (i.e. there are no more data items for that index entry) or might come up as novel, previously unseen key elements. For instance we might encounter new combinations of properties which spawn new property sets in an I_{PS} implementation. As we are interested in comparing densities over our indices we need to consider the effect of such zero-size entries. Using a maximum likelihood estimation for the densities would lead to zero probabilities for certain events which renders comparisons of densities impractical. We apply smoothing to overcoming zero probabilities. We make use of Laplace smoothing which adds a small constant value of λ to all counts obtained for the number of results $|\sigma(k)|$. The parameter λ was set to 0.5 in our experiments.

[2] The implementation of the index models has been released under an open source license at `https://github.com/gottron/lod-index-models`.

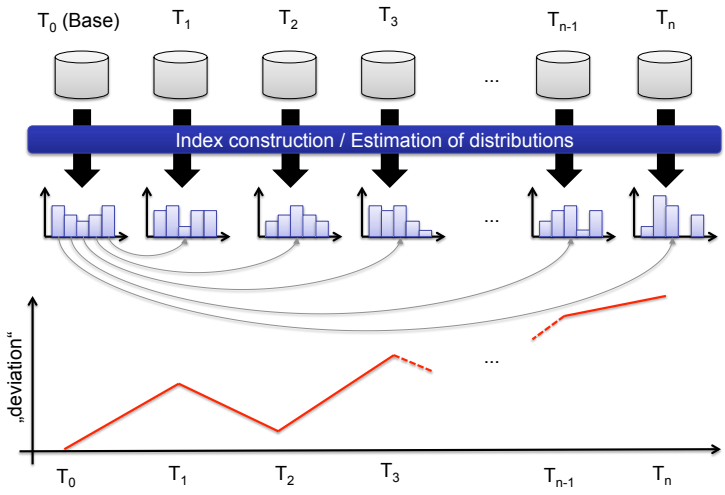


Figure 1. Evaluation of index stability: we built indices at different points in time over an evolving data set. The deviation in the distribution is measured by comparing the initial distribution to the distributions of the evolved data.

4 Experiments: Measuring Deviation of Index Models over Evolving Data

In our experiments, we empirically evaluate how accurate are density estimates obtained from implementations of index models over evolving data. To this end we build an index over the data at an initial point in time and obtain its distribution from the index. Then we compare this distribution to densities estimated over the same data at later points in time, when the data has potentially undergone changes. Figure 1 illustrates this process.

4.1 Metrics

We are comparing estimates of the density function for distributions of Linked Data items in an index structure. Common metrics to compare density functions are cross entropy, Kullback-Leibler divergence and perplexity. We briefly review the definitions of these metrics and explain their interpretation.

Assume we have estimated two probability distributions $P_1(X)$ and $P_2(X)$ at different points in time over an evolving data set. Then the cross entropy is defined as:

$$H(P_1, P_2) = -\sum_{k \in \mathcal{K}} P_1(X = k) \log(P_2(X = k)) \tag{2}$$

In the context of compression theory, cross entropy can be interpreted as the average number of bits needed to encode events following the distribution P_1 based on

an optimal encoding scheme derived from P_2. If the two distributions are equivalent, then cross entropy corresponds to the normal entropy $H(P_1)$. The entropy of P_1 also provides a lower bound for cross entropy. Based on this interpretation, the Kullback-Leibler divergence gives the deviation in entropy (or overhead in encoding) relative to the entropy for P_1 and is defined as:

$$D_{KL}(P_1, P_2) = H(P_1, P_2) - H(P_1) \tag{3}$$

Therefore, if two distributions are equivalent, they have a Kullback-Leibler divergence of zero. This is a desirable feature for our evaluation as it renders the comparison of distributions of evolving data independent from the different levels of the entropy observed for different index structures.

Perplexity, instead, provides an evaluation of a distribution by giving the number of events which under a uniform distribution would have the same entropy value. As such it is considered to be more easily interpretable by humans than the somewhat abstract entropy values. Perplexity itself is defined over entropy values, though. Here we formulate it directly on the basis of cross entropy:

$$PP(P_1, P_2) = 2^{H(P_1, P_2)} \tag{4}$$

Perplexity is a standard metric for evaluating probabilistic models. The lower the perplexity is, the better a model explains observed data and the more truthful are its estimates of the probabilities. When looking at perplexity over the cross entropy $H(P_1, P_2)$ in particular, there is also another interesting interpretation of the values. If perplexity is higher than the number of events considered, then using a simple uniform distribution instead of P_2 would correspond to a better approximation of the distribution P_1. Furthermore, the interpretation of perplexity relative to the event space allows for a normalisation. The normalised perplexity PP_{norm} is defined as $\frac{PP}{|X|}$, where $|X|$ denotes the size of the event space.

In addition to the metrics for comparing the density estimates, we use the Jaccard-similarity over the set of key elements. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be the sets of key elements derived at two points in time. Then the Jaccard-similarity is defined as:

$$\textit{Jaccard}(\mathcal{K}_1, \mathcal{K}_2) = \frac{|\mathcal{K}_1 \cap \mathcal{K}_2|}{|\mathcal{K}_1 \cup \mathcal{K}_2|} \tag{5}$$

A higher similarity value indicates a larger overlap between the sets of key elements while a low value indicates a stronger deviation. In this way we can get an impression of how stable the set of elements used for indexing is in the different indexing approaches.

To summarise: cross entropy provides an impression of the evolution of the absolute density, Kullback-Leibler divergence the deviation from the initial density, perplexity gives a more human interpretable view on the changes in the entropy values and the Jaccard-similarity allows for an assessment of the stability of the set of key elements used for indexing.

4.2 Data Set

We use the Dynamic Linked Data Observatory [11,10] data set. The data set provides weekly crawls of LOD data sources starting from always the same set of seed URIs. The initial snapshot from the 6th of May 2012 contains 16,376,867 RDF triples and covers a wide range of data sources. The data is provided in the form of quads containing the RDF statement as well as the source URI, where the triple was crawled from. Thus, it suits our formal requirements for the index models. Details on the design considerations, implementation and crawling strategy for the data set can be found in the original publications.

We used 77 data snapshots from the 6th of May 2012 up to the 8th of December 2013 for our experiments. The data was fed as raw input to implementations of all the twelve indexing models, without any further pre-processing[3].

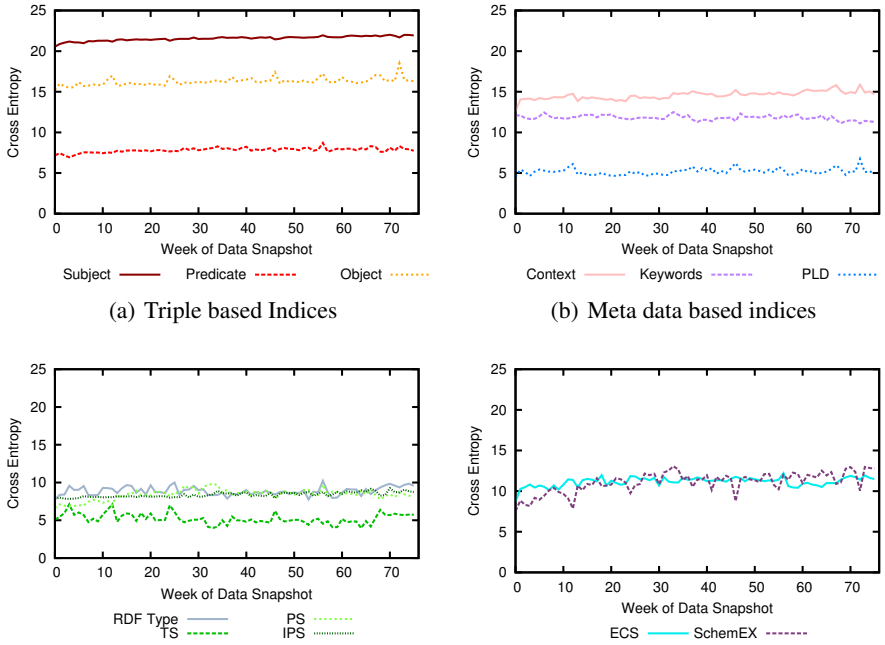
4.3 Results

We will now look at the performance of the different indexing models w.r.t. to the metrics measuring the ability to truthfully estimate density functions over evolving data.

We start to look at the development of cross entropy over time. The plots in Figure 2 show the result for the triple (Figure 2(a)), metadata (Figure 2(b)) and schema based index models. For the schema level index models we differentiate between the simpler models in Figure 2(c) and the ECS and SchemEX models making use of more extensive schema information in Figure 2(d). We can see nicely the different entropy levels for the individual indices. The explanation for this are more skewed distributions of the data in the index structures as well as different sizes of the key element sets. We can also observe some impacts of the evolving data on the entropy levels. The increase is not monotone, but shows some fluctuations over time. This can be attributed, on the one hand, to the data not shifting away homogeneously from its original distribution. On the other hand, such a behaviour can also be explained with the limited availability of certain data sources over time. As seen in previous analysis of the same data set, some data sources were not always available at all moments in time, causing a shift in the distribution due to the lack of the corresponding data and to peaks in the observed plots. Note, that this is not a flaw in the data set as it reflects a realistic scenario on the Web.

When looking at the Kullback-Leibler divergence in Figure 3, it can be seen how the deviations from the initial values develop. For most indices the development is behaving comparable in the sense that deviations appear approximately at the same points in time with the same direction of the deviations (this can be seen quite nicely for the triple and metadata based indices in Figure 3(a) and 3(b)). Two exceptions are the keyword based index, which has a much more stable behaviour in general, and the context based index, which exhibits an increase in Kullback-Leibler divergence around week 60. The stability of density estimates obtained from the keyword index are conceptual. As the

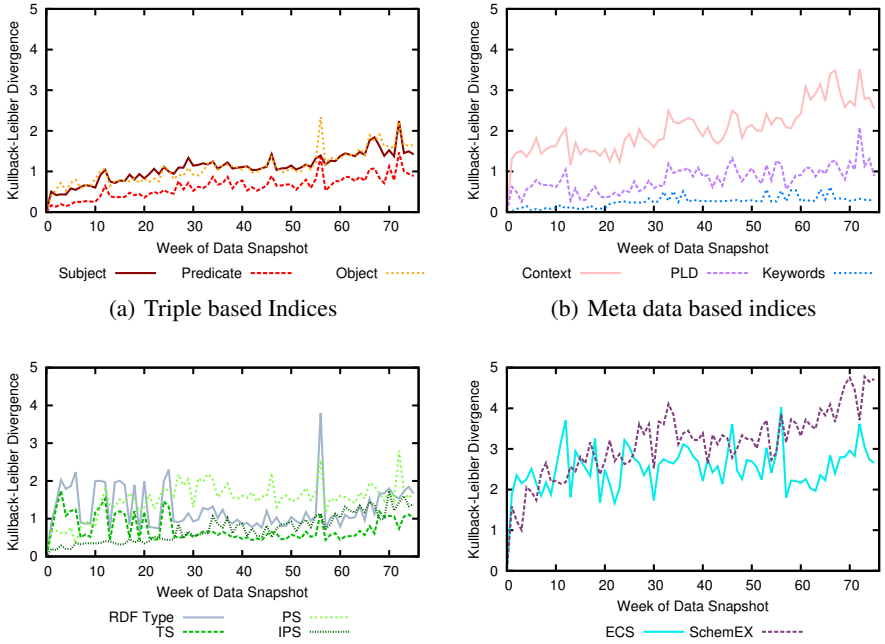
[3] Please note that SchemEX [9] is typically computed using an approximative, highly efficient stream-based approach. While in general the results of this approach are of high quality [6] we want to make sure it does not introduce a bias in the analysed index structures. Thus, we decided to compute precise, truthful and lossless SchemEX indices.



(a) Triple based Indices (b) Meta data based indices

(c) Schema based indices (d) ECS and SchemEX

Figure 2. Evolution of cross entropy for densities estimated over index structures.



(a) Triple based Indices (b) Meta data based indices

(c) Schema based indices (d) EC and SchemEX

Figure 3. Evolution of Kullback-Leibler divergence for densities estimated over index structures.

Table 1. Average and maximal perplexity of indices over evolving data

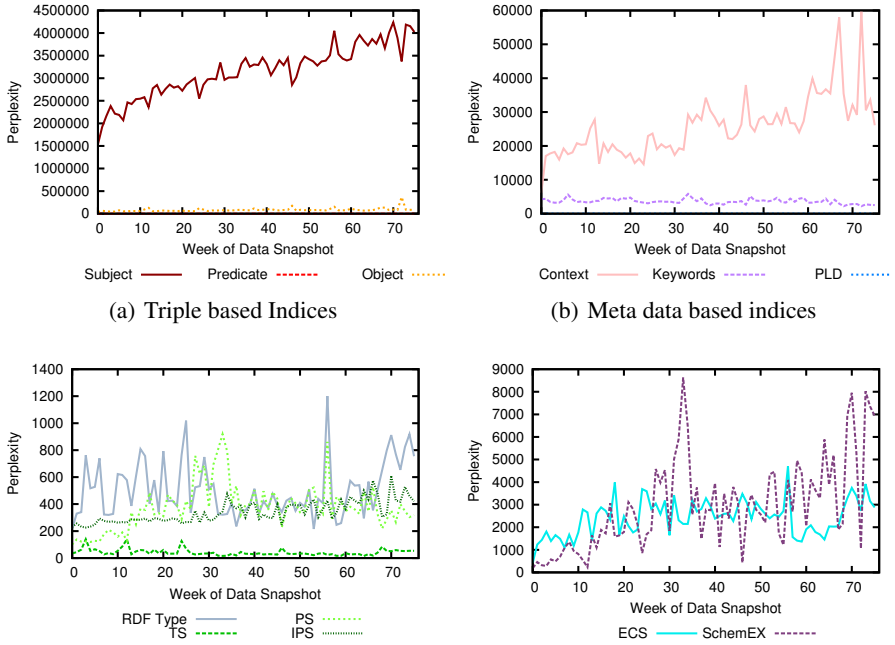
Index	S	P	O	C	PLD	Keyword	T	TS	PS	IPS	ECS	SchemEX
Size Key Set	3,665,267	11,554	2,887,357	68,665	746	1,057,790	11,154	25,727	35,985	12,555	89,252	118,473
Max. PP	4,237,601	421	368,459	59,888	107	5,837	1,201	140	919	611	4,699	8,632
Max. PP_{norm}	1.156	0.036	0.128	0.872	0.143	0.006	0.108	0.005	0.026	0.048	0.053	0.073
Avg. PP	3,148,052	230	83,438	25,758	38	3,660	506	42	380	342	2,430	2957
Avg. PP_{norm}	0.860	0.020	0.029	0.375	0.051	0.003	0.045	0.002	0.011	0.027	0.027	0.025

index effectively operates over single words as key elements, the density corresponds to a unigram language model. Given that the domain of the data did not change, we can hypothesise that the language style and domain remained relatively stable. The explanation for the increase in divergence of the context index around week 60, instead, lies in the data set. Investigating the data closer, revealed that at this point in time one particular data source (`taxonconcept.org`) started to contribute a significantly higher amount of triples than before. Thus, the strong impact on the context index.

The quality and stability of the density estimations can best be seen in the plots of the perplexity values in Figure 4. Again, we can observe a big difference in the absolute values. The overall highest values are observed for the subject based index. We can see in Figure 4(a), that the perplexity of this index increases relatively homogeneously and reaches it maximal value around week 70. Also several other indices show a more or less steady increase in perplexity. The simpler schema level indices in Figure 4(c), however, are relatively stable—with the exception of a few high peeks. The peeks align again with the unavailability of some data sources. In Figure 4(d) it is interesting to observe, that the perplexity of SchemEX is comparable to the one of the ECS index. This is surprising as SchemEX conceptually is more complex than the ECS and uses more extensive schema patterns. However, it seems with the more fine-grained schema level model it can better distinguish between the parts which have evolved and those which have remained stable. Therefore, also the density estimation is more reliable.

Table 1 provides an aggregated view on perplexity. The table lists information about the maximal and average perplexity values as well as normalised perplexity. There, we can see, for instance, that the maximal value of the subject based index corresponds to a perplexity value of 4,237,601. Given the initial size of the key element set of 3,665,267, the normalised entropy at this point in time reaches a values of 1.156. Thus, assuming a simple uniform distribution of the key elements would provide a more accurate estimation of the distribution. It remains to be said, though, that the distribution of the subject key elements in the analysed data set is in fact relatively close to a uniform distribution.

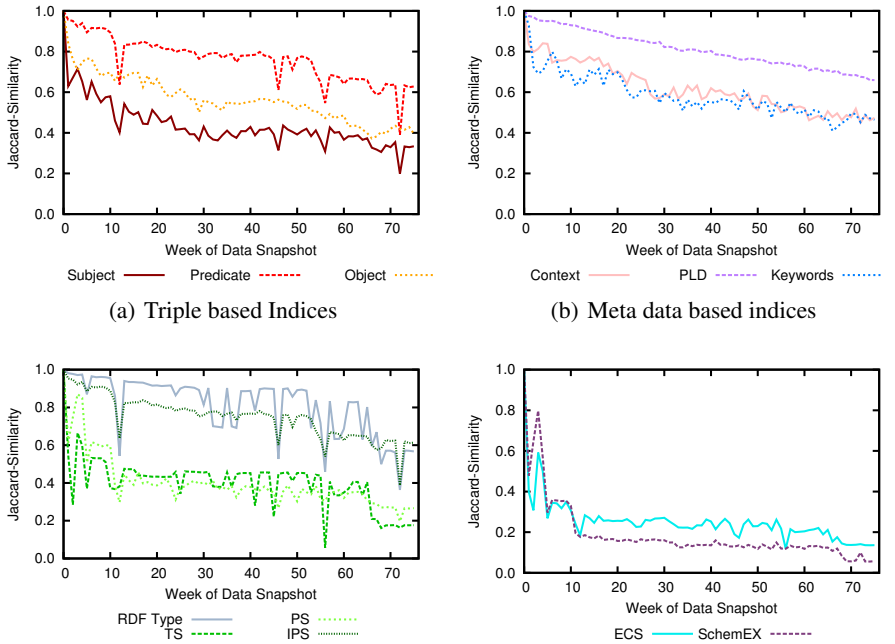
More interesting is the insight which can be obtained from the average normalised perplexity values in Table 1. Here, the single peeks of perplexity due to unavailable data sources have a lower impact. Furthermore, the normalisation renders the values comparable over all indices. We observe, that all schema level indices have very low values. This underlines the stability of these indices. Also, we can see again the keyword based index to have a low average normalised perplexity. Both observations align with the impression we obtained from the analysis of the Kullback-Leibler divergence.



(a) Triple based Indices (b) Meta data based indices

(c) Schema based indices (d) ECS and SchemEX

Figure 4. Evolution of Perplexity for densities estimated over index structures.



(a) Triple based Indices (b) Meta data based indices

(c) Schema based indices (d) ECS and SchemEX

Figure 5. Evolution of the Jaccard-similarity over the index element sets.

Finally, when looking at the evolution of the set of key elements over time in Figure 5, we can get some additional insights. While the similarity between the sets of key elements decreases more or less steadily for the triple and meta data based index models, it drops to relatively low values for several schema based models quite immediately. Most models drop to a Jaccard-similarity of 0.5 and below already after 10 weeks. The SchemEX index even drops to values of approximately 0.2. However, from a low overlap in the set of key elements we can not judge the overall quality of the index, as it is not clear how many data items are affected by the changed key elements. As we have seen before when looking at perplexity, SchemEX seems to cope very well in capturing and distinguishing patterns that are more or less stable. Thus, even if some of the key elements with very few associated data entries disappear, the effect on the estimates of densities is relatively low.

Quite interesting is also the observation that the deviations in the key elements set in IPS index seem much lower than for the IP index. This, however is an artefact of the data set and how it is generated. Given that the data set essentially corresponds to a crawl of a fixed set of seed URIs, we naturally get a change of all the modelled subject entities and their properties. For the objects, instead, we can only observe changes in the incoming relations from the considered subjects in the seed set. We cannot observe changes for those objects on the rest of the LOD cloud. On an unbiased data set, we would expect the two types of indices to behave comparable. This, however, remains to be verified.

5 Related Work

The changes and dynamics of Linked Data have been investigated in several publications. Umbrich et al. [15] give a good survey and provide a distinction between dynamics on the document and entity level. The Dynamic Linked Data Observatory data set has been introduced in [10]. It was analysed for changes in the data regarding triples, USUs, volume per data source, availability of the data on the Web (i.e. reachability of URIs) etc. The analysis showed a varying degree of changes in the data depending on the features considered. An extension of the analysis towards the schema level of LOD was presented by Dividino et al. [1]. A more detailed analysis of schema information revealed that also the schema level is heavily affected by the change in the data. The notion of schema elements in [1] is based on an ECS index model. However, all analytics focus on changes and dynamics on the side of the data. To the best of our knowledge, the impact of the changes on the accuracy of index models has not been analysed so far.

Index models and index structures for Linked Data or RDF in general are discussed in various contexts. Driven by the obvious need to index, cache and query distributed data, a wide range of solutions and applications have been proposed. We covered relevant publications in Section 2 when introducing the index models. However, when analysing index models most work considers mainly the efficiency of index structures for retrieving data or their effectiveness in a specific application context [13,14]. In these scenarios it is usually considered normal, that the index is aware of all changes in the data and is updated accordingly. Few publications consider index structures that are not always accurate. The motivation is either a more efficient computation of the

index over large scale data [9] or a distributed scenario where not all data is under the control of the authority managing the index, e.g. when federating SPARQL queries [2]. However, also in these cases the loss of accuracy has so far only been analysed for static data sets [9,6].

6 Conclusions and Future Work

In this paper we addressed the impact of evolving Linked Data on the accuracy of index models in providing reliable density estimations. Answering this question plays an important role given that, on the one hand, density estimations are central to several applications and that, on the other hand, Linked Open Data has been shown to be quite dynamic and evolving under several aspects. We formalised and implemented twelve prototypical index models from related work and evaluated their accuracy in estimating the density over evolving data. Employing metrics for comparing probability distributions we empirically analysed how far the densities obtained from an index diverged from the actual distributions in the evolving data. We observed that all densities estimated from implementations of the index models diverge from the densities of the evolving data. The divergence increases over time and particular events in the data caused stronger deviations for specific index models. For instance, models based on the data source were affected stronger by a burst and increase in the data volume provided by one specific data source. Finally, we also observed that models based on schema information seem to provide relatively stable estimations.

As future work we will investigate index specific strategies for performing evaluations of their own accuracy and corresponding update plans. This will include sampling strategies to identify the degree of data changes without considering the full data set. Furthermore, we will investigate more detailed methods for analysing the accuracy of index models when it comes to responding to concrete queries. The challenge in this case will be to provide representative queries, which cover all aspects of the data and how to deduce the overall change rate of the data from the divergence of query results.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013), REVEAL (Grant agree number 610928). The authors would like to thank Julius Gottron for a fruitful and lively discussion on indexing strategies.

References

1. Dividino, R., Scherp, A., Gröner, G., Gottron, T.: Change-a-LOD: Does the Schema on the Linked Data Cloud Change or Not? In: COLD'13: International Workshop on Consuming Linked Data (2013)
2. Görlitz, O., Staab, S.: Splendid: Sparql endpoint federation exploiting void descriptions. In: Proceedings of the 2nd International Workshop on Consuming Linked Data. Bonn, Germany (2011)
3. Görlitz, O., Thimm, M., Staab, S.: Splodge: Systematic generation of sparql benchmark queries for linked open data. In: Cudre-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat,

J., Hauswirth, M., Parreira, J., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web – ISWC 2012, Lecture Notes in Computer Science, vol. 7649, pp. 116–132. Springer Berlin Heidelberg (2012)
4. Gottron, T., Knauf, M., Scheglmann, S., Scherp, A.: A systematic investigation of explicit and implicit schema information on the linked open data cloud. In: Cimiano, P., Corcho, O., Presutti, V., Hollink, L., Rudolph, S. (eds.) The Semantic Web: Semantics and Big Data. Lecture Notes in Computer Science, vol. 7882, pp. 228–242. Springer Berlin Heidelberg (2013)
5. Gottron, T., Knauf, M., Scherp, A.: Analysis of schema structures in the linked open data graph based on unique subject uris, pay-level domains, and vocabulary usage. Distributed and Parallel Databases pp. 1–39 (2014)
6. Gottron, T., Pickhardt, R.: A detailed analysis of the quality of stream-based schema construction on linked open data. In: Li, J., Qi, G., Zhao, D., Nejdl, W., Zheng, H.T. (eds.) Semantic Web and Web Science, pp. 89–102. Springer Proceedings in Complexity, Springer New York (2013)
7. Gottron, T., Scherp, A., Krayer, B., Peters, A.: LODatio: Using a Schema-Based Index to Support Users in Finding Relevant Sources of Linked Data. In: K-CAP'13: Proceedings of the Conference on Knowledge Capture. pp. 105–108 (2013)
8. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.U., Umbrich, J.: Data summaries for on-demand queries over linked data. In: Int. Conf. on World wide web. pp. 411–420. ACM (2010)
9. Konrath, M., Gottron, T., Staab, S., Scherp, A.: Schemex—efficient construction of a data catalogue by stream-based indexing of linked data. Web Semantics: Science, Services and Agents on the World Wide Web 16(0), 52 – 58 (2012), the Semantic Web Challenge 2011
10. Käfer, T., Abdelrahman, A., Umbrich, J., O'Byrne, P., Hogan, A.: Observing linked data dynamics. In: Cimiano, P., Corcho, O., Presutti, V., Hollink, L., Rudolph, S. (eds.) The Semantic Web: Semantics and Big Data, Lecture Notes in Computer Science, vol. 7882, pp. 213–227. Springer Berlin Heidelberg (2013)
11. Käfer, T., Umbrich, J., Hogan, A., Polleres, A.: DyLDO: Towards a Dynamic Linked Data Observatory. In: Workshop on Linked Data on the Web (LDOW) (2012)
12. Neumann, T., Moerkotte, G.: Characteristic sets: Accurate cardinality estimation for rdf queries with multiple joins. In: Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany. pp. 984–994 (2011)
13. Neumann, T., Weikum, G.: Scalable join processing on very large rdf graphs. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. pp. 627–640. ACM (2009)
14. Stocker, M., Seaborne, A., Bernstein, A., Kiefer, C., Reynolds, D.: Sparql basic graph pattern optimization using selectivity estimation. In: Proceedings of the 17th international conference on World Wide Web. pp. 595–604. ACM (2008)
15. Umbrich, J., Hausenblas, M., Hogan, A., Polleres, A., Decker, S.: Towards Dataset Dynamics: Change Frequency of Linked Open Data Sources. In: LDOW (2010)
16. Völker, J., Niepert, M.: Statistical schema induction. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., Leenheer, P., Pan, J. (eds.) The Semantic Web: Research and Applications, Lecture Notes in Computer Science, vol. 6643, pp. 124–138. Springer Berlin Heidelberg (2011)