CognitiveDB: An Intelligent Navigator for Large-scale Dark Structured Data

Michael Gubanov, Manju Priya, Maksim Podkorytov
Department of Computer Science
University of Texas at San Antonio
{mikhail.gubanov, maksim.podkorytov}@utsa.edu,
manjupriya.harikrishnan@my.utsa.edu

ABSTRACT

Volume and *Variety* of Big data [Stonebraker, 2012] are significant impediments for anyone who wants to quickly understand what information is inside a large-scale dataset. Such data is often informally referred to as 'dark' due to the difficulties of understanding its contents. For example, there are millions of structured tables available on the Web, but finding a specific table or summarizing all Web tables to understand what information is available is infeasible or very difficult because of the scale involved.

Here we present and demonstrate `CognitiveDB`, an intelligent cognitive data management system that can quickly summarize the contents of an unexplored large-scale structured dataset, visualize it for interactive navigation, understanding, and support intelligent query processing.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: General.

Keywords

Web-search; Large-scale Data Management; Cloud Computing; Data Fusion and Cleaning; Summarization; Visualization.

1. INTRODUCTION

Increasing amounts of data are produced every year at a staggering rate, yet technological barriers to data access have limited their usefulness. For example, data scientists at large enterprises desire to enrich their data to do a better job in predictive analytics tasks. However, their previous experiences enriching this data from external data sources suggest that data acquisition costs are high, therefore they are reluctant to spend their time finding and acquiring all missing data pieces. Consequently, they do not take all relevant data into account, which leads to degraded performance of their predictive algorithms.

WWW 2017 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054735

Consider a data scientist who is trying to predict hard cider sales in the regions of interest to be able to restock accordingly for the next season. Since consumption of hard cider is correlated with weather, access to the complete weather data per region would improve prediction accuracy. However, such a dataset is not available internally, hence, the data scientist is considering enriching existing data with detailed weather data from an external data source (e.g. the Web). However, the cost to do ETL (Extract-Transform-Load) from a large-scale external data source like the Web in order to integrate weather data is expected to be high, therefore the data scientist never acquires the missing data pieces and does not take weather into account.

In response, we present `CognitiveDB` - a new interactive Web-based navigator that quickly provides meaningful insight into an unknown, large-scale structured data set, to help identify and locate information of interest. Figure 2 illustrates a view through `CognitiveDB` onto a large-scale structured dataset, containing *36 million* tables extracted from the Web. The left pane of the interface contains a bulleted list, which stores the main tables of objects sorted by their rank. The right pane displays a graph of these groups and the tables in each group. A sample of the data is displayed in the right pane when the user clicks on the table in the left pane.

To identify and rank the main tables amongst millions of tables in a large-scale structured dataset, we introduce *OBRank* (rank of an Object), an algorithm to measure table importance in the dataset. Similar to PageRank [Brin and Page, 1998] that operates on the Web graph, we construct a graph from tables in the dataset and account for the direction and degree of a node in this graph to calculate the node rank. Then, having calculated *OBRank*, we group the tables by their rank, and display a list of tables with the highest rank to the user (see Figure 1, left pane). With such a concise summary, the user can quickly grasp what a large-scale structured dataset is mostly about, without having to spend time fully exploring it. Without *OBRank* and `CognitiveDB` this task becomes prohibitively difficult and the user is faced with millions of tables and the problem of understanding what is inside the dataset. There is no concise summary of the dataset provided, such as the one that `CognitiveDB` generates, and a few random samples of the dataset are of little use to the user trying to discern the contents. Similar to a Web search engine, the user may type an object name into the search-box (see Figure 1) to obtain all possible matches in the dataset. Because keyword-searches of this type might provide inaccurate or incomplete search

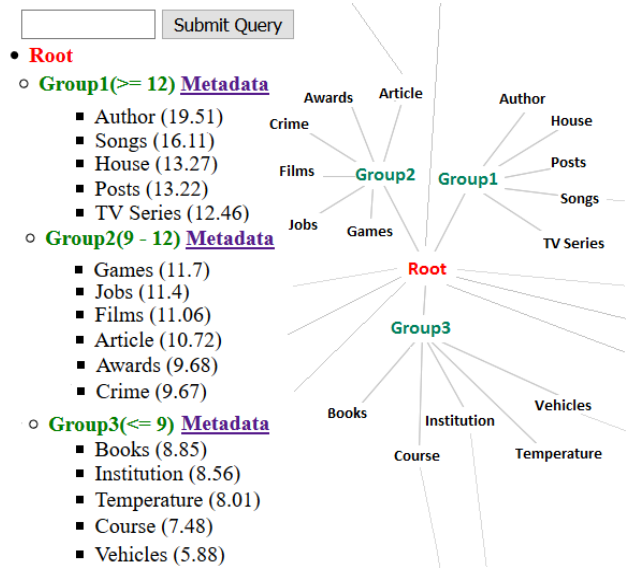
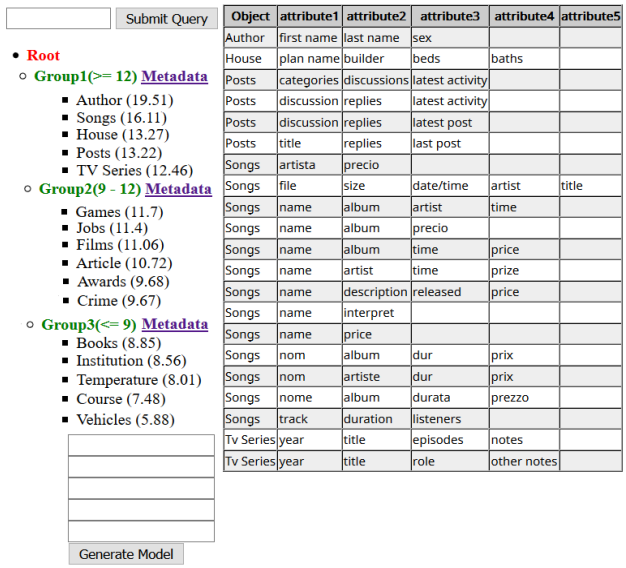


Figure 1: An easy to grasp summary of a large-scale 'dark' dataset having millions of Web tables.



Object	attribute1	attribute2	attribute3	attribute4	attribute5
Author	first name	last name	sex		
House	plan name	builder	beds	baths	
Posts	categories	discussions	latest activity		
Posts	discussion	replies	latest activity		
Posts	discussion	replies	latest post		
Posts	title	replies	last post		
Songs	artista	precio			
Songs	file	size	date/time	artist	title
Songs	name	album	artist	time	
Songs	name	album	precio		
Songs	name	album	time	price	
Songs	name	artist	time	prize	
Songs	name	description	released	price	
Songs	name	interpret			
Songs	name	price			
Songs	nom	album	dur	prix	
Songs	nom	artiste	dur	prix	
Songs	nome	album	durata	prezzo	
Songs	track	duration	listeners		
Tv Series	year	title	episodes	notes	
Tv Series	year	title	role	other notes	

Figure 2: A metadata sample of the most prominent group of Web tables (Group 1).

results[1], we have designed an alternative option. The user enters a few descriptive attributes for a table in the bottom left corner (see Figure 2), in response `CognitiveDB` automatically trains a large-scale machine learning classifier, capable of identifying such tables and uses it in the future to answer user queries relating to this table. The classifier is trained only on data in the table, so it does not depend on the table or attribute names. Such a classifier simulates human *cognitive* ability (object recognition in this instance), hence the system name `CognitiveDB`. Of course, human cognitive abilities are far more advanced, however, they are extraordinarily inefficient at processing large amounts of data, and therefore unable to grasp, understand, and use a large-scale dataset quickly.

In the rest of the paper, we will review related work (Section 2), give an overview of the system architecture (Section 3), describe how the user can interact with `CognitiveDB` (Section 4), and conclude with a brief summary of the technical problem that the paper addresses (Section 5).

2. RELATED WORK

We studied a variety of relevant systems in large-scale information extraction, search, and data management. [Lugmayr et al., 2017], [Lugmayr et al., 2016] are complete, comprehensive surveys of contemporary research in Big data. Authors discuss what is really new in Big data, and more importantly introduce a new *five-Trait framework for Cognitive Big data* applicable to many real-world domains. [Cafarella et al., 2007] investigated the problem of schema amendment in a structured data corpus with present data tuples, but incomplete or missing attribute names, and described an algorithm (`TGEN`) to perform this task. In another work, [Cafarella et al., 2008] create a database engine `AcsDB` (attribute correlation statistics database) that ingests Web tables for use in different scenarios. They describe several use cases (in context of the same dataset): schema auto-complete (given a set of attributes, suggest more relevant attributes), attribute synonym finding (given an attribute, suggest synonymous attributes), and join-graph discovery. [Balmin et al., 2004] describe an algorithm to calculate rank of nodes, resulting from keyword-search over a database. The authors apply it to rank publications in DBLP. They use the intuition similar to Pagerank to propagate authority through the graph of publications in DBLP. Our graph is constructed from tables in the large-scale dataset and their connections and differs from the graph in [Balmin et al., 2004]. The algorithm to calculate and propagate rank is also different. [Pimplikar and Sarawagi, 2012] introduce a Web table search engine which allows for the specification of a set of keywords, then consolidates a corpus of tables to retrieve a single table with these keywords as column descriptions. The problem of matching the column in the table to the query keyword was addressed by exploiting the knowledge about web table context (i.e. parent HTML document) and the page element styling information to infer table relevance to the query. However, they do not suggest any ranking functions to help the user find the most relevant object. [Adelfio and Samet, 2013] also attempted to infer metadata for Web tables. They distinguished more data rows (instead of just table metadata and data) and used this information to improve attribute name inference from the data. They also used page element styling as a hint to assist with solving the problem of schema extraction. Finally, they applied these techniques to spreadsheet data, another source of structured data. [Balakrishnan and et al, 2015] discuss the problem of web table corpus construction and created a search engine to query these web table corpuses, presenting the query result in a reusable format. They use supervised Machine Learning as well as simple heuristic rules to wipe out uninformative web tables. They also use Machine Learning to match tables to queries by detecting relationships between columns.

Unfortunately, these recent contributions, while very important in their own ways, do not provide an easy to grasp

[1]The same table might be referred to by different names, moreover many extracted Web tables do not have names at all - just attributes and data rows.

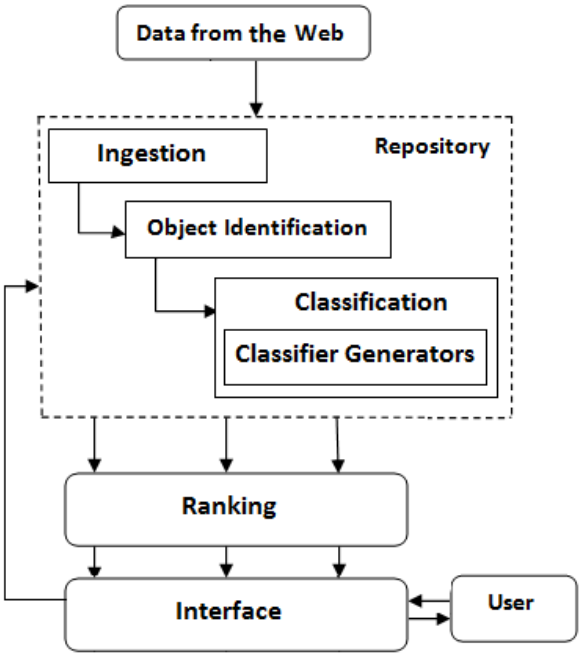


Figure 3: `CognitiveDB` Software Architecture

summary of a large-scale structured dataset that can be used to quickly understand what is inside the 'dark data' dataset. `CognitiveDB` summarizes the dataset contents and outputs a concise list of main tables ranked by their footprint in the dataset and connectivity to other tables. The object-identifying classifiers, generated by `CognitiveDB` are trained on data, hence do not depend on table name, and do not suffer from mismatches to 'dirty' metadata that is usually the case for any table search engine matching the table/attribute names to retrieve query results.

3. SOFTWARE ARCHITECTURE

The architecture of `CognitiveDB` is depicted in Figure 3, the components of which are described below in more detail.

Web Data: We use a focused crawl of a portion of the Web (open to crawlers as declared by robots.txt) using *Nutch* [Khare and Cutting, 2004], index the crawl with *Lucene* inverted indexing [luc, 2011], then use a part of the crawl to extract Web tables by iterating over the indexed corpus. A detailed description of these steps is not a focus of our discussion here and is omitted due to space constraints [Cafarella et al., 2008, Cafarella et al., 2007, Etzioni and et al, 2004, Gubanov et al., 2009, Paton et al., 2016, Gubanov, 2017b, Gubanov et al., 2014, Gubanov et al., 2016].

Ingestion: The tables extracted in the previous step are cleansed and ingested into a large-scale parallel column store for further analysis. The data coming from the Web are usually 'dirty' (examples include empty tables, junk advertisements, typos and misspellings) and requires cleaning before ingestion. Detailed description of these steps is not a focus of our discussion here and is omitted due to space constraints [Cafarella et al., 2008, Chu and et al, 2015, Gubanov and Pyayt, 2013, Gubanov and Bernstein, 2006b, Gubanov et al., 2011, Gubanov and Pyayt, 2012, Gubanov and Shapiro, 2012, Gubanov et al., 2009, Gubanov and Stonebraker, 2014, Gubanov and Bernstein, 2006a, Gubanov and Pyayt, 2016, Gubanov, 2017a, Priya et al., 2017, Gubanov, 2017b].

Object retrieval with SQL: Once all tables have been cleaned and ingested into a parallel column store, a scalable algorithm is required to identify and group similar tables. For example, *course* and *class* might be used for University course data, while *vehicle* and *car* might be used for automobile research. We started by designing and optimizing SQL queries to run over a large-scale parallel column store to retrieve groups of similar objects.

Generating Scalable Machine Learning Classifiers: To improve accuracy of retrieval, we train an ensemble of large-scale Machine Learning classifiers, based on several descriptive keywords entered by the user. Figure 5 demonstrates automatic generation of a classifier detecting *Album* tables given several descriptive keywords from the user. Entering a few keywords is easier for anyone who wants to train a classifier compared to labeling the training data needed for training any supervised Machine learning model. The user empirically identifies a set of attributes that an object of interest is likely to possess. Then we take these attributes and generate a parallel SQL query extracting objects with these attributes from the dataset, similar to what a decision-tree Machine Learning classifier would do [Mitchell, 1997]. We observe a wide range of Precision/Recall (65-90%) for this type of extraction, based solely on attributes. To improve accuracy, we train an ensemble of C4.5 trees with a Naive Bayes Multinomial using the previously mentioned extraction as positively labeled training data [Mitchell, 1997]. We intentionally create a high-precision decision tree extractor with lower recall to get cleaner training data. We then amend it with automatically generated negative examples.

Interface: We designed an easy to use user interface to navigate and display a summary of a structured, large-scale dataset. The left frame (Figure 1) shows a ranked list of main data objects identified in the dataset with their assigned groups, the right frame displays a graph of relationships. This high-level structured summary of the dataset allows the user to quickly understand and navigate the dataset, an otherwise challenging task considering the scale involved. Based on their footprint in the dataset and their relationships to other objects, the objects are ranked. Section 4 describes the interface and user interactions in more detail. Object Ranking is described below.

Dataset: We used a parallel relational column-store, running in a distributed environment [Kornacker and et al, 2015, Stonebraker et al., 2005] to store 36 million Web tables, and then ran summarization, ranking, and classification experiments. The dataset had a total of 86 million data instances.

Object Ranking: The objects, displayed in the left frame of the interface (Figures 1,2,4,5) are ranked using the OBRank formula below. This ranking function has two components. The first one is the number of instances of an object in the dataset, the second component depends on relationships of an object with other objects. Hence, a highly connected object having fewer instances in the dataset might have a better ranking compared to an object with more instances, but no connections. Hence, we define *OBRank* as:

$$OBRank(P) = log(w(P)) + \sum_{P_j \in B_P}^{n} log(R_{P_j}) \qquad (1)$$

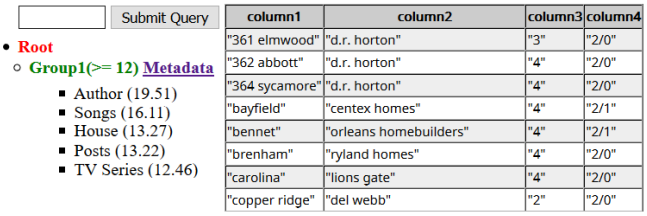


Figure 4: A data sample from one of the largest objects in the datset (*House*, Group 1)

where OBRank(P) is the rank of object P; n is the number of objects referring to object P: $j = 0 \dots n$; B_P is the set of objects that P is referring to. For example, if the *Article* has an attribute *Author*, we define that *Author* refers to *Article*. R_{P_j} is the number of references from P to P_j. w(P) is the weight of object P equal to the total number of instances of P in the dataset. For example, *Author* has 13,564 instances and 21,879 references to *Article*, hence the rank is 19.51. *Artist* having 255,357 references to *Songs* has rank 24.9. *Studio* having 16,168 references to *Films* has rank 19.38. These are examples of *OBRank* calculation, based on the connectedness of our dataset. It would be different for another dataset having different connectivity among the objects. If an object in the dataset is orphaned, the second component of the *OBRank* formula will be zero. However, the first component will be non-zero and still meaningfully reflect the importance of an object in the dataset [Brin and Page, 1998, Gubanov et al., 2009, Gubanov and Bernstein, 2006b].

4. DEMONSTRATION

The online demo screencast of `CognitiveDB` in action is available on YouTube[2]. Figures 1, 2, 4, 5 illustrate the interactive Web interface. In all of them, the left frame displays a *summary* of the large-scale structured dataset in a form of a treeview of groups of main objects sorted by their OBRank. This is similar to how a Web-search engine would rank Web pages using static ranking like PageRank [Brin and Page, 1998] and relative similarity to the user query. All objects with the highest rank are assigned to Group 1, lower ranked to Group 2, etc. The right frame content changes depending on the user actions. We demonstrate the working `CognitiveDB` prototype via the following use cases.

Summarization: The user provides a new large-scale (millions of tables) structured dataset in specific pre-defined format, suitable for rapid ingestion. The system ingests the dataset, automatically identifies, groups, and ranks the main tables by *OBRank* and displays a summarizing treeview in the left pane (Figure 1).

Connectivity: The user selects a group of objects in the left pane and the system generates a graph of objects with relationships in the right pane (Figure 1).

Data sampling: The user selects a specific object in the left pane, then a small sample of its instances is displayed in the right pane (Figure 4).

Metadata sampling: The user clicks on the *Metadata* link adjacent to the cluster name in the left pane. The system outputs in the right pane the list of objects with attributes from the selected group (Figure 2).

[2]https://youtu.be/ovdscf6evHk

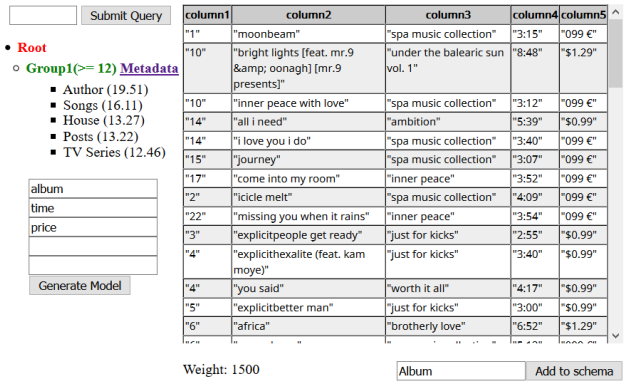


Figure 5: Automatic generation and training of a large-scale machine learning ensemble recognizing an object of interest, given just three descriptive attributes from the user and no additional training data.

Classifier generation: The bottom part of the left pane in Figure 5 consists of a set of text forms, where the user can enter the attributes best describing an object and have the system generate and train the classifier to identify such objects. Once the classifier is generated, a sample of newly classified objects is shown to the user in the right pane. If the user is satisfied with the results (i.e. the classified instances in the right pane are what the user has expected), s/he can click the *Add to schema* button, which saves the new classifier in the system and includes it in the left pane's treeview.

5. CONCLUSION

`CognitiveDB` is a working navigator and search engine that allows the user to visualize, understand, traverse, and query a large-scale structured dataset. The system automatically *summarizes* the dataset and generates a ranked list of main tables from millions available. It also allows the user to describe an object with a few keywords and automatically train a scalable classifier recognizing such objects. Without `CognitiveDB` it is very difficult to quickly get an idea of what is inside a large-scale structured data set with millions of tables.

Author biographies

Michael Gubanov is a Cloud Technology Endowed Assistant Professor at the University of Texas at San Antonio. He earned his PhD from the University of Washington and did postdoctoral research at Massachusetts Institute of Technology. Most of his research is in Large-scale Data Management, Data Integration, and Web-search. During his PhD studies he worked at Google, IBM Almaden Research Center, and Microsoft Research.

Manju Priya is an M.Sc. student at the University of Texas at San Antonio in the Computer Science department. She earned her B.Tech. in Information Technology from Anna University, India. She is interested in Large-scale Data Management, Web-search, and Large-scale Machine Learning. She is working on Web-search and Large-scale Data Integration.

Maksim Podkorytov is a Ph.D. student in the Computer Science department at the University of Texas at San Antonio. He earned his M.Sc. in Computational Mathematics from Lomonosov Moscow State University, Russia. He is working on Web-search, Large-scale In-memory Storage Engines, and Data Integration.

6. REFERENCES

[luc, 2011] (2011). online: http://lucene.apache.org/core/.

[Adelfio and Samet, 2013] Adelfio, M. D. and Samet, H. (2013). Schema extraction for tabular data on the web. *VLDB*.

[Balakrishnan and et al, 2015] Balakrishnan, S. and et al, A. H. (2015). Applying webtables in practice. In *CIDR*.

[Balmin et al., 2004] Balmin, A., Hristidis, V., and Papakonstantinou, Y. (2004). Objectrank: Authority-based keyword search in databases. In *In VLDB*.

[Brin and Page, 1998] Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *WWW*.

[Cafarella et al., 2008] Cafarella, M. J., Halevy, A., Wang, D. Z., Wu, E., and Zhang, Y. (2008). Webtables: exploring the power of tables on the web. *VLDB*.

[Cafarella et al., 2007] Cafarella, M. J., Suciu, D., and Etzioni, O. (2007). Navigating extracted data with schema discovery. In *WebDB*. Citeseer.

[Chu and et al, 2015] Chu, X. and et al, J. M. (2015). Katara: Reliable data cleaning with knowledge bases and crowdsourcing. In *VLDB*.

[Etzioni and et al, 2004] Etzioni, O. and et al, M. C. (2004). Web-scale information extraction in knowitall. In *WWW*.

[Gubanov, 2017a] Gubanov, M. (2017a). Hybrid: A large-scale in-memory image analytics system. In *CIDR*.

[Gubanov, 2017b] Gubanov, M. (2017b). Polyfuse: A large-scale hybrid data fusion system. In *ICDE DESWeb*.

[Gubanov and Bernstein, 2006a] Gubanov, M. and Bernstein, P. A. (2006a). Structural text search and comparison using automatically extracted schema. In *WebDB*.

[Gubanov et al., 2016] Gubanov, M., Jermaine, C., Gao, Z., and Luo, S. (2016). Hybrid: A large-scale linear-relational database management system. In *MIT Annual DB Conference*.

[Gubanov and Pyayt, 2012] Gubanov, M. and Pyayt, A. (2012). Medreadfast: Structural information retrieval engine for big clinical text. In *IRI*.

[Gubanov and Pyayt, 2013] Gubanov, M. and Pyayt, A. (2013). Readfast: High-relevance search-engine for big text. In *ACM CIKM*.

[Gubanov and Pyayt, 2016] Gubanov, M. and Pyayt, A. (2016). Type-aware web search. In *EDBT*.

[Gubanov et al., 2011] Gubanov, M., Pyayt, A., and Shapiro, L. (2011). Readfast: Browsing large documents through ufo. In *IRI*.

[Gubanov and Shapiro, 2012] Gubanov, M. and Shapiro, L. (2012). Using unified famous objects (ufo) to automate alzheimer's disease diagnostics. In *BIBM*.

[Gubanov and Stonebraker, 2014] Gubanov, M. and Stonebraker, M. (2014). Large-scale semantic profile extraction. In *EDBT*.

[Gubanov et al., 2014] Gubanov, M., Stonebraker, M., and Bruckner, D. (2014). Text and structured data fusion in data tamer at scale. In *ICDE*.

[Gubanov and Bernstein, 2006b] Gubanov, M. N. and Bernstein, P. A. (2006b). Structural text search and comparison using automatically extracted schema. In *WebDB*.

[Gubanov et al., 2009] Gubanov, M. N., Popa, L., Ho, H., Pirahesh, H., Chang, J.-Y., and Chen, S.-C. (2009). Ibm ufo repository: Object-oriented data integration. *VLDB*.

[Khare and Cutting, 2004] Khare, R. and Cutting, D. (2004). Nutch: A flexible and scalable open-source web search engine. Technical report.

[Kornacker and et al, 2015] Kornacker, M. and et al, A. B. (2015). Impala: A modern, open-source sql engine for hadoop. In *CIDR*.

[Lugmayr et al., 2017] Lugmayr, A., Stockleben, B., and Mailaparampil, M. (2017). Cognitive big data: survey and review on big data research and its implications. what is really 'new' in big data? *Journal of Knowledge Management*, 21(1).

[Lugmayr et al., 2016] Lugmayr, A., Stockleben, B., and Scheib, C. (2016). A comprehensive survey on big data research and its implications – what is really 'new' in big data? – it is cognitive big data! In *PACIS*.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA.

[Paton et al., 2016] Paton, N. W., Belhajjame, K., Embury, S. M., Fernandes, A. A., and Maskat, R. (2016). Pay-as-you-go data integration: Experiences and recurring themes. In *ICCTPI*.

[Pimplikar and Sarawagi, 2012] Pimplikar, R. and Sarawagi, S. (2012). Answering table queries on the web using column keywords. *VLDB*.

[Priya et al., 2017] Priya, M., Podkorytov, M., and Gubanov, M. (2017). ilight: A flashlight for large-scale dark structured data. In *MIT Annual DB Conference*.

[Stonebraker, 2012] Stonebraker, M. (2012). Big data means at least three different things... In *NIST Big Data Workshop*.

[Stonebraker et al., 2005] Stonebraker, M., Abadi, D., and et al, A. B. (2005). C-store: A column-oriented dbms. In *VLDB*.