

READ: Rapid data Exploration, Analysis and Discovery

Udayan Khurana*
University of Maryland
College Park
udayan@cs.umd.edu

Srinivasan Parthasarathy
IBM Research
Yorktown Heights
spartha@us.ibm.com

Deepak Turaga
IBM Research
Yorktown Heights
turaga@us.ibm.com

ABSTRACT

Exploratory data analysis (EDA) is the process of discovering important characteristics of a dataset or finding data-driven insights in the corresponding domain. EDA is a human intensive process involving data management, analytic flow deployment and model creation, and data visualization and interpretation. It involves extensive use of analyst time, effort, and skill in data processing as well as domain expertise. In this paper, we introduce READ, a mixed initiative system for accelerating exploratory data analysis. The key idea behind READ is to decompose the exploration process into components that can be independently specified and automated. These components can be defined, reused or extended using simple choice points that are expressed using inference rules, planning logic, and reactive user interfaces and visualization. READ uses a formal specification of the analytic process for automated model space enumeration, workflow composition, deployment, and model validation and clustering. READ aims to reduce the time required for exploration and understanding of a dataset from days to minutes.

1. INTRODUCTION

Exploratory data analysis (EDA) is a critical statistical approach for discovering important characteristics of a dataset or finding data-driven insights into the corresponding domain. The core objective of EDA is to identify interesting phenomena in an application domain, and also help an analyst formulate interesting hypotheses that explain these phenomena. EDA involves the use of a variety of statistical and data visualization techniques to discover the core entities in an application domain, their characteristics, key features, interesting behaviors, and relationships. EDA has found wide spread acceptance across multiple domains including semiconductor manufacturing, analysis of communication networks, cyber security and threat analysis, fraud analysis, insurance, log analysis, and econometrics.

A key challenge with the EDA process is its human intensive nature, such as the need for extensive time and effort from highly skilled statisticians and domain experts as illustrated in Figure 1 and exemplified in Figure 2. The analyst starts with a collection of data sets, and deploys a variety of SQL or equivalent map-reduce analytic flows on the data in order to clean and create useful intermediate representations. This step is followed by a model creation step where the analyst typically creates analytic flows, either

via SQL, or via statistical packages such as R, Mahout, SPSS, and SAS. This results in the generation of one or more models that are visually explored by the analyst: an example of a model could be a univariate feature computed for each entity within the data set, with the anomalous entities marked out in a box plot based on this feature. If the models are domain relevant and of interest to the analyst, then further annotations or labels are created based on the model, which leads to new knowledge that is used in subsequent iterations of EDA.

During the EDA process, not all the models constructed by the analyst might lead to interesting insights. This implies that multiple iterations are usually required before the discovery of a domain relevant insight. The iterative nature of EDA and its heavy reliance on the analyst skill and effort contributes to the *high time to discovery*. Further the number of models explored is limited by the human in the loop, which might mean that some non-trivial and potentially interesting analysis paths leading to insightful models and discoveries may never get explored. Another significant challenge in this process is the extent of skills demanded from the analyst. This may include data management, data analysis, query construction and statistical and visualization expertise. Our system, READ, is aimed at these complex challenges in data exploration and discovery through an unprecedented level of automation.

READ is a mixed initiative system for data exploration and stands for Rapid Exploration Analysis and Discovery on Big Data. The key contributions and innovation of READ are described below:

1. READ decouples the description of data sets within the system from the description of analytics and queries deployed on the data. Analytics are represented within READ using a set of choice points: for instance, a histogram query on a data set can be represented using the table ingested in the query, the attributes by which the rows of the table are grouped for aggregation, the statistical function and the attributes on which the function is applied for histogram model generation. We employ automated reasoning for generating a space of potentially interesting models that can be derived from data: this is achieved by reasoning about the various possible ways in which analytic choice points can be instantiated concretely from the data specification.
2. READ uses automated planning technologies for workflow and analytic composition that result in the creation of models. While the analyst still has the option of pruning and filtering the set of models generated from the data, the capabilities described thus far largely frees up the analyst from mundane data management, analytic flow creation, and coding tasks that are involved in model building.
3. Automated model generation in READ yields the capability

*The author was an intern at IBM during the time of this work.

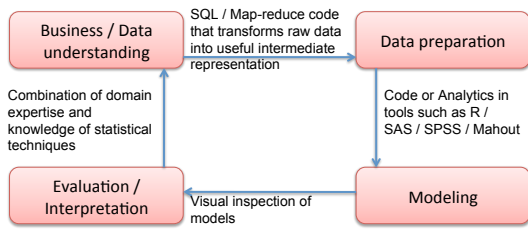


Figure 1: The iterative nature of exploratory data analysis as exemplified by the Cross Industry Standard Process by Data Mining (CRISP-DM) [1]. Notice the manual human-intensive nature of this process during the data preparation and modeling tasks.

to rapidly generate hundreds or thousands of models via analytic flow deployment over datasets. In order to scale up the interpretation of these models by the analyst, READ uses novel visual and statistical techniques for model exploration. Specifically, READ provides a mechanism to plug in validation metrics, that determine which of the automatically generated models is the most interesting one. For instance, a histogram model can be interesting due to its entropy, skewness, or sparsity. READ also provides a mechanism to plug in similarity or distance metrics in order to find the models that are fairly similar to an interesting model. Clustering such a set of models under one category helps remove redundancy in the space of models. By generalizing the analysis for a representative model from the cluster, the analyst saves time on redundant inspection and interpretation. For instance, if five hundred models out of a pool of two thousand models mark out the same set of entities as anomalies, they could all be clustered together. In this case, the similarity is based on a Jaccard coefficient metric.

Audience Experience with READ Demo: We believe the combination of automated model enumeration, workflow composition, and validation and similarity metrics for rapid model interpretation makes for a novel, exciting, and accelerated process of knowledge discovery on massive amounts of data. During this demonstration of READ, users will be able to play with pre-loaded data sets, or plug in their own data sets and descriptions and watch READ auto-compute and enumerate a variety of model building options. Users can prune or filter from this enumeration (this is an optional human input). Once the subset of models to be built is generated, through a single click of a button, users will also be able to witness automated workflow composition and deployment mechanisms at work leading to mass production of models that have been enumerated in the earlier step. Finally, users will be exposed to the READ model interpretation subprocess where interesting anomalies and trends are easily identified via validation metrics, and model redundancy is eliminated through similarity metrics. Specifically, the amount of time spend by users in *driving* READ would be proportional to the number of insights (e.g., distinct type of anomalies) they discover within the data, leading to a broad coverage of analysis paths explored and at the same time, fruitless analysis paths being eliminated.

2. RELATED WORK

John Tukey pioneered the use of exploratory data analysis nearly four decades ago [10]. Various packages and languages that support exploratory data analysis have been developed since. This in-

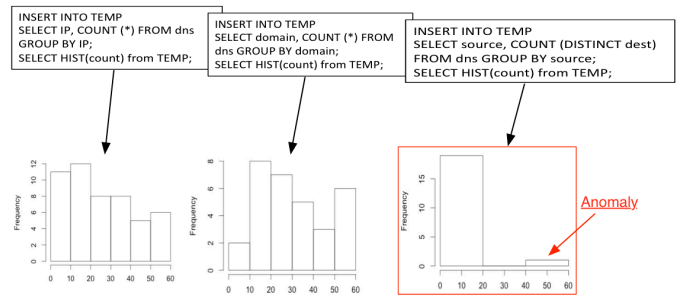


Figure 2: The analyst is interested in generating cyber situational awareness through a dataset of Domain Name Service(DNS) transactions. the analyst creates a variety of models which in this case are univariate features for various entities in the network; the models are generated by issuing multiple SQL queries and visualized as histograms. The third model is a discriminative feature which segregates DNS servers as anomalous hosts as opposed to DNS clients which are regular hosts.

cludes, S, S-Plus, R, SPSS, SAS, and Matlab [7, 2]. A recent view of modern data exploration practices is available from Behrens and Yu [5]. SeeDB is a different line of work focused on finding the appropriate visualizations for a database query [9].

Online Analytic Processing (OLAP) is a key technology that supports exploratory data analysis and advanced predictive mining and modeling tasks that could be deployed on the data [8]. OLAP supports a variety of data models and operations including multidimensional conceptual views of data, query performance, intuitive data manipulation interfaces (e.g., slicing and dicing, drilling down), and cross dimensional operations. Another line of research that is related to READ is the use of automated planning algorithms for workflow composition and task execution within the EDA process [3, 4, 11]. READ and these works can mutually benefit each other as their design goals are orthogonal. Specifically, in READ: First, the enumeration and generation of the various analytics and queries that could be deployed on the data are computed by READ automatically (as opposed to providing optimized performance or workflow composition assistance for user initiated analytic tasks); Second, analytic automation and automated model generation with READ results in an explosion in the number of machine generated models and views of the data; READ supports intuitive and rapid exploration of these auto generated models through a novel collection of visual analytics for model validation and similarity clustering, which is not a consideration in the approaches outlined above.

3. READ DESIGN AND IMPLEMENTATION

Data analysis in READ is decomposed into four key subprocesses: 1) Data description, 2) Model space enumeration, 3) Model generation, and 4) Model interpretation. This is illustrated in Figure 3. We now describe the design and implementation of READ components that support these subprocesses.

3.1 Components

Data Description: We illustrate READ's data description methodology in Figure 4. Data description in READ is intended to provide the same *look and feel* as schema description in a database with a few notable enhancements. There are opportunities within this description to identify specific attributes of a dataset as representing entities within a domain. Entities are the *things* of importance to an

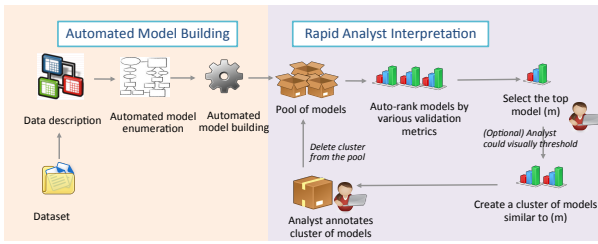


Figure 3: The READ discovery methodology: Automated model building enumerates and generates all possible models or analytic flows for any given collection of data sets. Analyst interpretation is aided by pluggable validation and similarity metrics. Analyst interaction helps select the most interesting model and the ones similar to it, thereby eliminating redundancy. Together, this methodology ensures wide coverage by slicing and dicing data, at the same time reducing the analyst time to be proportional to the number of distinct interpretations that could be drawn from the data.

Meta Data Editor								
Tables: dns								
[isSourceable: true]								
Att Name	Time	Trans_ID	Src	Dest	Query	D_name	IP	Q_Type
Att Type	Timestamp	String	String	String	Boolean	String	String	Enum
Is Entity?	false	false	true	true	false	true	true	false
Entity			IBMHost	IBMHost		Domain	Ext_IP	
Range								{ssh,http}
<input type="button" value="Reset"/> <input type="button" value="Add Column"/> <input type="button" value="Save Changes"/>								

Figure 4: Data specification in READ is similar to specifying a data base schema with a few enhancements. An attribute marked as an entity is of special interest - specifically, models and features could be built for this entity. Additionally, the metadata about the range and type of entity help define rules for model building as well.

analyst in the domain, i.e, things for which models could be built. For example, in cyber security analysis, entities may correspond to hosts, web domains, IP addresses, applications, ports, and subnets. In a business intelligence domain, entities may include users, products, and stores. Additional layers of specification could include descriptions of enum attributes (what range do they span). These descriptions are utilized by READ’s model enumeration logic for automated enumeration of data preparation and model creation options.

READ users supply data specification through a web-based front-end which stores them as JSON objects within the file system. In our implementation, the data specification front-end is also connected with Hadoop/HIVE’s meta store which maintains the schema of the various data sets stored that are stored in a HIVE database.

Model Enumeration: An example of an inference rule is presented in Algorithm 1. This rule reasons on the data description and enumerates univariate features or descriptive statistical aggregates (based on pre-specified statistical aggregation functions). Model enumeration logic such as this for a variety of data preparation steps (e.g., joins and filters) as well as univariate feature creation are specified in the JBoss Drools Rule engine within our current implementation. Algorithm 9, iterates through the attributes in the data spec, identifying each attribute that is marked as entity. Entities are the groups within a group by aggregate query: for each group, the rule

Database	TableName	GroupBy	GroupBy/Entity	StatFunc	Attrib
Database: cyberlite (746)					
cyberlite	firewall...	externaladdress	IPAddress	count	null
cyberlite	firewall...	address	Host	count	null
cyberlite	firewall...	externaladdress	IPAddress	distinctCount	ibmaddress
cyberlite	firewall...	externaladdress	IPAddress	max	messagenumber
cyberlite	firewall...	externaladdress	IPAddress	min	messagenumber
cyberlite	firewall...	externaladdress	IPAddress	distinctMean	messagenumber
cyberlite	firewall...	externaladdress	IPAddress	median	messagenumber
cyberlite	firewall...	externaladdress	IPAddress	mean	messagenumber

Figure 5: Example of a subset of the space of possible univariate features and histogram models that are computable from DNS data set by applying automated enumeration logic for model creation.

computes a statistical or aggregate function using a second attribute from the dataset. Our intention with Algorithm 9 below is merely to showcase the art of the possible, and not to prescribe a final logic for univariate feature creation. The key here is the decoupling of data specification from analytic reasoning, which allows us to plug in a variety of data sets and automatically enumerate the space of models computable for them. A browsable view of the model space that is automatically enumerated by such inferencing is illustrated in Figure 5.

Algorithm 1 Example of an inference rule for enumerating univariate features from data specification

```

if  $D$ : Data Set with set of Attributes  $A$  then
  if  $a \in A$  and  $a$  is an entity then
    if  $b \in A$  and  $b \neq a$  then
      if  $sf$  is a statistical function applicable to a variable
        of type  $b$  then
          Compute feature  $f := sf(b)$  from  $D$  for per
          group  $a$ 
        end if
      end if
    end if
  end if

```

Model Building: Once the space of models has been enumerated, queries are automatically composed and deployed using the MARIO automated composer [6]. We omit further details due to lack of space.

Model Validation: The pool of generated models, potentially in thousands, need to be interpreted by the analyst. Instead of providing each constructed model to the analyst for interpretation, READ provides a visually interactive mechanism for the analyst to select the top most interesting model as per one (amongst many) validation metrics. Figure 6 shows the selection of the most important box plot amongst approximately 1800 models using the minimum entropy metric. As per the box plot, the median value is zero, there are several non-zero entities, while one has an exceptionally high value. The analyst could choose from several other options such as max skewness, min zero count to name a few.

Model Clustering: Often, many different models suggest the same or similar facts. For instance, box plots for counts or perhaps, those using a different statistical function such as distinct count or entropy for the same entity are likely to point the same result (anomaly and more precisely, a primary or secondary DNS server in our cyber-security example). Similar observations can be derived for other entities which exhibit anomalous behavior as well. Hence, for a model that seems to be of interest, the analyst can choose other similar models to make a collective observation. The crite-

Histogram based Network Anomaly Detection

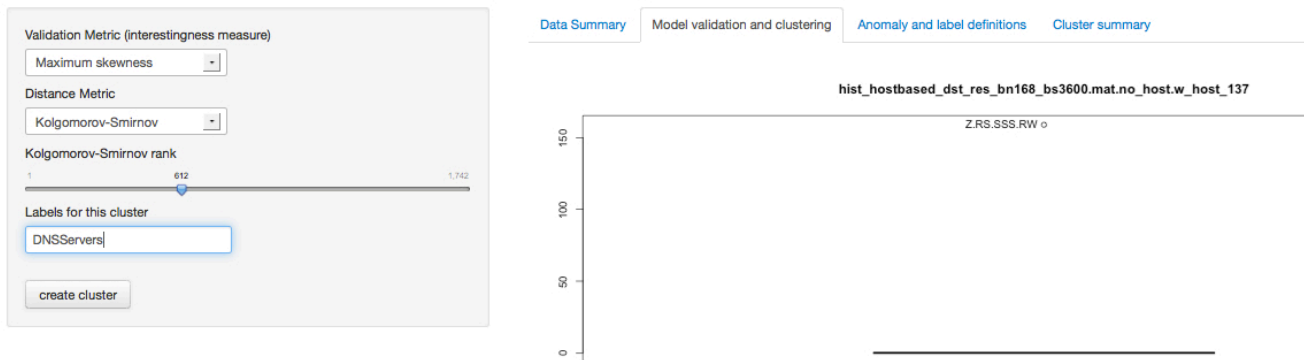


Figure 6: A novel visual analytics user interface that enabling rapid analyst interaction for interpretation of models. This interaction provides a variety of pluggable validation metrics - each metric ranks and decides on the most interesting model in the collection and a variety of distance metrics, which eliminate model redundancy by identifying other models close to the most interesting model. The user can choose the threshold at which this model cluster is created. The interpretation of this model cluster is represented using a cluster label: in this instance, the user has grouped 612 models out of 1742 models using the similarity threshold and called this cluster `DNSServers`, as this group of models provides a way to discriminate between DNS servers and clients.

ria of similarity can be selected through drop-down menus (e.g., the Kolmogorov-Smirnov distance metric in the menu in Figure 6). The threshold of similarity can be selected from a slider in an interactive manner, i.e., selecting 612 will display the 612th model by the similarity metric, apart from the top ranked model itself.

Interpretation and Annotation: If the analyst finds the observation from a cluster of models interesting, the corresponding entities in the model may be labeled. In our example, the analyst may give a label of “server” or “dns server” to the anomalies. The labels will be stored in a *non-sourcable* table by default.

4. DEMONSTRATION PLAN

During the demonstration, our focus will be on showing the following key features to the audience: (a) Enumerating the model space using data description and model rules, (b) Analyst interpretation including model validation and clustering. The primary objective will be to discover and label anomalies in the respective datasets. Once demonstrated, the audience will be encouraged to change description/rules and immediately see the effect on model creation. Apart from the DNS dataset, we plan to include certain datasets from healthcare¹ and cyber-security², ready to be used. The audience will also have the option to plugin a reasonable sized dataset, which may be uploaded fairly quickly to the cluster. We will also illustrate the details of the process involved, starting from data specification to the implicit and explicit choices made through out the analytic workflow. The process will also highlight the analyst effort and the artifacts such as the code generated in the process.

5. CONCLUSION

In this paper, we presented a demonstration of READ, a mixed initiative system for rapid exploration, analysis and discovery on large datasets. We are currently investigating the design of an algebraic framework which formally models the process of exploratory data analysis. This is intended to meld elements of relational al-

gebra which has revolutionized the field of databases, as well as process algebra which has revolutionized the field of concurrent system design. We are also investigating the use of hypothesis generation and planning technologies with the goal of designing an automated READ analyst assistant.

6. REFERENCES

- [1] Cross industry standard process for data mining. http://en.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining.
- [2] Exploratory data analysis. http://en.wikipedia.org/wiki/Exploratory_data_analysis.
- [3] Robert St. Amant and Paul R. Cohen. Interaction with a mixed-initiative system for exploratory data analysis, 1997.
- [4] Robert St. Amant and Paul R. Cohen. Intelligent support for exploratory data analysis. *Journal of Computational and Graphical Statistics*, 7:545–558, 1998.
- [5] J. Behrens and C. Yu. *Exploratory Data Analysis*. 2003.
- [6] Eric Bouillet, Mark Febowitz, Hanhua Feng, Anand Ranganathan, Anton Riabov, Octavian Udrea, and Zhen Liu. Mario: middleware for assembly and deployment of multi-platform flow-based applications. In *Middleware (Companion)*, page 26, 2009.
- [7] J. Chambers, T. Hastie, et al. *Statistical models in S*. Chapman & Hall London, 1992.
- [8] Codd E.F., Codd S.B., and Salley C.T. Providing olap (on-line analytical processing) to user-analysts: An it mandate. <http://www.fpm.com/refer/codd.html>, 1993.
- [9] Aditya Parameswaran, Neoklis Polyzotis, and Hector Garcia-Molina. Seedb: Visualizing database queries efficiently. *PVLDB*, 7(4):325–328, 2013.
- [10] J Tukey. Exploratory data analysis. *Reading, Ma*, 231, 1977.
- [11] M. Zakova, P. Kremen, F. Zelensky, and N. Lavrac. Automating knowledge discovery workflow composition through ontology-based planning. *Automation Science and Engineering, IEEE Transactions on*, 8(2):253–264, 2011.

¹<http://physionet.org/mimic2/>

²<http://www.caida.org/data/>