Masters of Ceremony & Editors: Polo Chau Jilles Vreeken Matthijs van Leeuwen Dafna Shahaf Christos Faloutsos

Proceedings of the
ACM SIGKDD 2015 Full-day Workshop on

Interactive Data Exploration and Analytics

IDEA 2015

Sydney, Australia
Aug 10, 2015

poloclub.gatech.edu/idea2015

Proceedings of the
ACM SIGKDD Workshop on
Interactive Data Exploration and Analytics

These proceedings are not included in the ACM Digital Library.

IDEA'15, August 10, 2015, Sydney, Australia

ACM SIGKDD Workshop on Interactive Data Exploration and Analytics

General Chairs

Duen Horng (Polo) Chau (Georgia Tech)
Jilles Vreeken (Max Planck Institute for Informatics and Saarland University)
Matthijs van Leeuwen (KU Leuven)
Dafna Shahaf (Microsoft Research)
Christos Faloutsos (Carnegie Mellon University)

Program Committee

Adam Perer (IBM, USA)
Aditya Parameswaran (UIUC, USA)
Antti Ukkonen (Finnish Institute of Occupational Health, Finland)
Arno Knobbe (Amsterdam University of Applied Sciences, the Netherlands)
Arno Siebes (Universiteit Utrecht, the Netherlands)
B. Aditya Prakash (Virginia Tech, USA)
Cody Dunne (UMD, USA)
Danai Koutra (U. Michigan, USA)
Esther Galbrun (Boston University, USA)
Geoff Webb (Monash University, Australia)
George Forman (HP Labs)
Hanghang Tong (Arizona State University, USA)
Hoang-Vu Nguyen (Max Planck Institute for Informatics and Saarland University)
Jacob Eisenstein (Georgia Tech, USA)
Jaegul Choo (Georgia Tech)
Jefrey Lijffijt (University of Bristol, UK)
Kai Puolamäki (Finnish Institute of Occupational Health, Finland)
Leman Akoglu (Stony Brook University)
Lisa Singh (George Town, USA)
Marti Hearst (U. Berkeley, USA)
Nan Cao (IBM, USA)
Parikshit Ram (Georgia Tech, USA)
Pauli Mietinnen (Max Planck Institute for Informatics, Germany)
Saleema Amershi (Microsoft Research, USA)
Steffen Koch (U. Stuttgart, Germany)
Thomas Gärtner (U. Nottingham, UK)
Tijl De Bie (University of Bristol, UK)
Tim (Jia-Yu) Pan (Google, USA)
Tina Eliassi-Rad (Rutgers, USA)
Wouter Duivesteijn (TU Dortmund, Germany)
Zhicheng 'Leo' Liu (Stanford, USA)

Preface

Data, data everywhere; massive datasets of previously unthinkable sizes, surpassing terabytes and petabytes, have quickly become commonplace. They arise in numerous settings in science, government, and enterprises. While technology exists by which we can collect and store such massive amounts of information, making sense of these data remains a fundamental challenge. In particular, we lack the means to exploratively analyze databases of this scale. Currently, surprisingly few technologies allow us to freely "wander" around the data, and make discoveries by following our intuition, or serendipity. While standard data mining aims at finding highly interesting results, it is typically computationally demanding and time consuming, thus may not be well-suited for interactive exploration of large datasets.

Interactive data mining techniques that aptly integrate human intuition, by means of visualization and intuitive **human-computer interaction** techniques, and **machine computation** support have been shown to help people gain significant insights into a wide range of problems. However, as datasets are being generated in larger volumes, higher velocity, and greater variety, creating effective interactive data mining techniques becomes an increasingly harder task.

It is exactly this research, experiences and practices that we aim to discuss at IDEA, the workshop on Interactive Data Exploration and Analytics. In a nutshell, IDEA addresses the development of data mining techniques that allow users to interactively explore their data. We focus and emphasize on **interactivity** and effective **integration** of techniques from **data mining**, **visualization** and **human-computer interaction**. In other words, we explore how the best of these different but related domains can be combined such that the *sum is greater than the parts*.

Following the great success of IDEA at KDD 2013 and at KDD 2014, the main program of IDEA'15 consists of nine papers that cover various aspects of interactive data exploration and analytics. All papers will be presented both orally, as well as during the interactive poster and demo session. We express our most sincere gratitude to Microsoft Research for sponsoring our workshop and this session. These papers were selected from a total of 16 submissions after a thorough reviewing process. We sincerely thank the authors of the submissions and the attendees of the workshop. We wish to thank the members of our program committee for their help in selecting a set of high-quality papers. Furthermore, we are very grateful to Geoff Webb and Jure Leskovec for engaging keynote presentations on the fundamental aspects of interactive data exploration and visualization.

Polo Chau & Jilles Vreeken & Matthijs van Leeuwen & Dafna Shahaf & Christos Faloutsos

Saarbrücken, July 2015

Table of Contents

Invited Talk

The Knowledge Factory: A Retrospective

Geoff Webb
Faculty of Information Technology
Monash University, Australia
Geoff.Webb@monash.edu

Abstract

This talk revisits the first major program of research into interactive rule discovery. The Knowledge Factory is an interactive rule learning system developed in the 1990s. It has many novel features that still remain relevant today. The talk will cover the key techniques that the research developed, interpreting them in the light of subsequent developments in the field.

Bio

Geoff Webb is a Professor of Information Technology Research in the Faculty of Information Technology at Monash University, where he heads the Centre for Data Science. His primary research areas are machine learning, data mining, user modeling and computational structural biology. Many of his learning algorithms are included in the widely-used Weka machine learning workbench. A commercial implementation of his association discovery techniques, Magnum Opus, has been acquired by BigML, Inc. for inclusion in their cloud-based data mining solution.

He was editor-in-chief of the highest impact data mining journal, Data Mining and Knowledge Discovery from 2005 to 2014. He is co-editor of the Springer Encyclopedia of Machine Learning, a member of the advisory board of Statistical Analysis and Data Mining, a member of the editorial board of Machine Learning and was a foundation member of the editorial board of ACM Transactions on Knowledge Discovery from Data. He has been Program Committee Co-Chair of the two top data mining conferences, ACM SIGKDD International Conference on Knowledge Discovery from Data (2015) and the IEEE International Conference on Data Mining (2010) and General Co-Chair of the 2012 IEEE International Conference on Data Mining. He is a technical advisor to BigML, Inc. He is an IEEE Fellow and has received the 2013 IEEE ICDM Service Award and a 2014 Australian Research Council Discovery Outstanding Researcher Award.

Invited Talk

Machine Learning for Human Decision Making

Jure Leskovec
Department of Computer Science
Stanford University
jure@cs.stanford.edu

Abstract

In many real-life settings human judges are making decisions and choosing among many alternatives in order to label or classify items: Medical doctor diagnosing a patient, criminal court judge making a decision, a crowd-worker labeling an image, and a student answering a multiple-choice question. Gaining insights into human decision making is important for determining the quality of individual decisions as well as identifying mistakes and biases. In this talk we discuss the question of developing machine learning methodology for estimating the quality of individual judges and obtaining diagnostic insights into how various judges decide on different kinds of items. We develop a series of increasingly powerful hierarchical Bayesian models which infer latent groups of judges and items with the goal of obtaining insights into the underlying decision process. We apply our framework to a wide range of real-world domains, and demonstrate that our approach can accurately predict judge decisions, diagnose types of mistakes judges tend to make, and infer true labels of items.

Bio

Jure is an assistant professor of Computer Science at Stanford University and chief scientist at Pinterest. His research focuses on mining large social and information networks. Problems he investigates are motivated by large scale data, the Web and on-line media. This research has won several awards including a Microsoft Research Faculty Fellowship, the Alfred P. Sloan Fellowship and numerous best paper awards. Jure received his bachelor's degree in computer science from University of Ljubljana, Slovenia, and his PhD in in machine learning from the Carnegie Mellon University and postdoctoral training at Cornell University. Jure also co-founded a machine learning startup Kosei which was recently acquired by Pinterest.

Visual Interactive Neighborhood Mining on High Dimensional Data

Emin Aksehirli[•] Bart Goethals[•] Emmanuel Müller[•○]

[•] University of Antwerp, Belgium
{firstname.lastname}@uantwerpen.be

[○] Karlsruhe Institute of Technology, Germany
emmanuel.mueller@kit.edu

ABSTRACT

Cluster analysis is widely used for explorative data analysis, however, it is not trivial to select the right method and optimal parameters. Moreover, not all clustering methods can work with raw or dirty data. In this paper, we introduce an interactive data exploration tool, VINeM, which combines interactive mining with unsupervised tools by exploiting an intuitive neighborhood-based visualization technique. Local neighborhood based visualization is useful not only for analyzing multiple (dis-)similarity measures but also for effectively discarding noise. VINeM works well with high dimensional data and can be used to find subspace clusters.

Keywords

subspace clustering, clustering, interactive data mining, high dimensional data, subspace selection, data visualization

1. INTRODUCTION

Explorative data analysis is an important tool that is used in both academic and industrial research areas. In recent years computational discoveries have become more affordable than the actual experiments, thanks to Moore's law. Therefore, researchers try to infer as much as they can from a limited set of data and refer to experiments only for conclusive results [14].

In most data exploration scenarios, little is known about the data, which means that the data is not annotated, and hence, not a good fit for supervised learning tasks, e.g. classification. Unsupervised methods are more suitable for data exploration tasks since they can work with limited initial knowledge about the data.

Unsupervised methods have their own challenges, foremost of which is the selection of the method that best fits the data at hand. This is not easy considering the thousands of available clustering methods in the literature [17]. Moreover, the selection of the algorithm is the first step of the process. Finding its optimal parameters is the next challenge which requires an understanding of the data.

To get a grasp of the data, one can rely on the most general and the least complicated techniques, such as descriptive statistics or visualizations. Descriptive statistics depend heavily on assumptions about the data. Not only determining them is not easy, but also, statistics can be misleading if those assumptions do not hold. Data visualization techniques, however, are easy to use by non-expert users and provide significant information. Therefore, data visualisation has always been an attractive research area [19]. Furthermore, using suitable visualizations, the user can be involved in the critical steps to improve the discovery process [11, 7]. On the other hand, existing tools either (1) are not capable of visualizing different views on the data [26], (2) do not integrate interactive and unsupervised mining methods [13, 23, 8], (3) are not aware of clusters that exist in subsets of the dimensions [27], (4) or are not designed to find the clusters [29].

Data from observations are rarely usable as they are. Before starting the actual analysis, analysts should deal with additional data cleaning steps, such as removing noise and dealing with missing values. Methods that are robust enough to cope with dirty or unstructured data can significantly shorten this tedious process.

In this paper, we introduce an interactive high-dimensional data analysis tool that can visualise different views on the data in a unifying framework while seamlessly integrates unsupervised and interactive mining tasks. In summary, the contributions of this paper are as follows:

- Neighborhood-based unifying data visualization,
- Micro cluster-based relevant dimension detection,
- GUI application that combines interactive data exploration with automated tools,
- Use case scenarios for various data exploration tasks.

In Section 2, we discuss the properties of high-dimensional data space in terms of clustering. Then, we introduce a neighborhood-based data representation that can cope with high-dimensional and noisy data while being easily visualizable (Section 3). In Section 4, we present our software tool that exploits this representation to make the data more available to the user, both for understanding and for wrangling. We explore two use-case scenarios in Section 5 and conclude in Section 6.

2. CLUSTERING HIGH DIMENSIONAL DATA

As a result of the advances in the data gathering and data storing technologies, we can associate many attributes with a single data object. Although more data may provide us with new insights, it may also hinder the discovery process by cluttering the interesting relations with redundant information. Furthermore, since the data objects become more and more alike with an increasing number of dimensions [6], the traditional definition of similarity becomes meaningless in high-dimensional data, and hence, clustering methods that depend on the similarity between objects fail to cope with high-dimensional data.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

On the other hand, similarities of the objects according to a subset of attributes can still be meaningful. *For example, if a group of people have similar values for a specific bio-marker and they show a tendency for drug abuse, then other attributes such as the eye color, weight or sex are probably irrelevant.* Therefore, meaningful knowledge in high-dimensional data is often extracted as a tuple of similar objects and the attributes in which they are similar. Such information is called a *subspace cluster*.

Formally, a data object $\mathbf{o}$ is defined as a vector over a set of attributes $\mathscr{A}$. A dataset $\mathscr{DB}$ is a collection of data objects. A cluster is a set of objects $C \subset \mathscr{DB}$. A subspace cluster is defined as a tuple of an object set and an attribute set, $SC = (C, A)$ where $A \subseteq \mathscr{A}$.

High dimensionality poses a problem during the visualization as well. Feature selection techniques, such as PCA [24] and MDS [10], are used in the literature to represent data in a lower dimensional space. However, feature selection is done on the whole dataset and can therefore easily miss the subspace clusters [23]. Reducing the dimensions while keeping the cluster structures is also proposed [25, 30, 21], but requires a computationally expensive preprocessing which also depends on the assumptions on the data. Therefore, they are not suitable for interactive data exploration settings.

Scatter plot matrices show a 2D matrix of scatter plots for each pair of dimensions. Although they are useful to get a grasp of the relatively lower dimensional data, it gets harder to interpret them for the high-dimensional data because the number of charts is a combinatorial function of the number of dimensions. Parallel coordinates represents the relations of objects in different projections. They provide an interactive exploration environment but they cannot be used for non-univariate projections, i.e., they can represent only 1D projections [16].

3. NEIGHBORHOOD DATABASE

3.1 Object Neighborhoods

"Tell me who your friends are, and I will tell you who you are." The core concept of this famous phrase is successfully applied to many cases of data analysis. Neighborhoods, i.e. *friends*, of data objects provide robust assessment of the similarity. They can even be more accurate than the actual features in some cases [18]. Neighborhoods are a good estimator for determining class labels [12], or whether an object is an outlier [9]. As they are good at preserving the local relations, they are used to overcome the problems of high dimensionality [3].

DEFINITION 1 (NEIGHBORHOOD). *Neighborhood of an object* $\mathbf{o}$*, denoted by* $N(\mathbf{o})$*, is a set of objects that are similar to* $\mathbf{o}$*.*

DEFINITION 2 (ε-NEIGHBORHOOD). *The radius-based neighborhood,* ε*-Neighborhood, of an object* $\mathbf{o}$ *is defined as the set of all objects that are more similar to* $\mathbf{o}$ *than a certain scalar value. Formally, let* $\delta : \mathscr{DB}^2 \rightarrow \mathbb{R}_+$ *be a dissimilarity measure,* $\varepsilon \in \mathbb{R}_+$ *and* $\mathbf{o}, \mathbf{p} \in \mathscr{DB}$*,*

$$\varepsilon\text{-}N(\mathbf{o}) = \{\mathbf{p} | \delta(\mathbf{o}, \mathbf{p}) < \varepsilon\}$$

DEFINITION 3 (k-NEAREST NEIGHBORHOOD). *Let* $NN_k(\mathbf{o})$ *represent the* k^{th} *closest object to* $\mathbf{o}$*, the k-nearest neighborhood of* $\mathbf{o}$ *is:*

$$k\text{-}NN(\mathbf{o}) = \{\mathbf{p} | \delta(\mathbf{o}, \mathbf{p}) \leq \delta(\mathbf{o}, NN_k(\mathbf{o}))\}$$

Since the k-nearest neighborhood uses a relative similarity threshold, it is more robust for assessing the similarities in heterogeneous data than ε-neighborhood. Note that we use the concept of generic

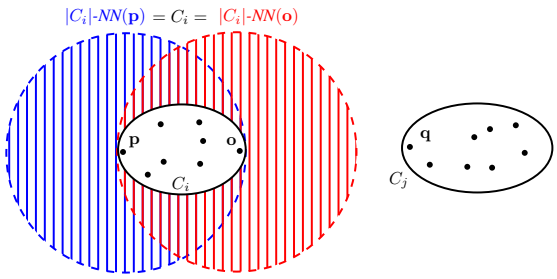


Figure 1: Case of separable clusters

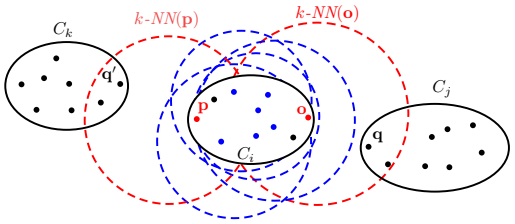


Figure 2: Case of non-separable clusters

neighborhood (Definition 1) if the type of the neighborhood is irrelevant.

DEFINITION 4 (NEIGHBORHOOD DATABASE). *The Neighborhood Database (*$\mathscr{ND}$*) is the collection of neighborhoods of all objects. Figure 3b shows the neighborhood database of the dataset in Figure 3a.*

Since *a cluster is defined as a group of similar data objects*, there is a clear connection between the cluster structures in the data and the neighborhoods. Figure 1 shows two clusters that are clearly separated, i.e., each object in a cluster is more similar to objects in the same cluster than to objects in the other clusters. For each object $\mathbf{o} \in C_i$, its k-nearest neighborhood with k equal to the size of its cluster, $|C_i|$-$NN(\mathbf{o})$, is equal to the cluster itself. Formally, if cluster C_i is clearly separated, then $|C_i|$-$NN(\mathbf{o}) = C_i, \forall \mathbf{o} \in C_i$. If the clusters are not separated, e.g., as shown in Figure 2, the most of the objects in C_i still include the whole cluster, provided that the neighborhoods are large enough.

Neighborhoods of the objects that are in the same cluster are either the same or share a large set of objects. Therefore, we can find

	A_1	A_2
$\mathbf{o}_1$	4000	3200
$\mathbf{o}_2$	5	6
$\mathbf{o}_3$	7	6
$\mathbf{o}_4$	3000	4000
$\mathbf{o}_5$	6	8
$\mathbf{o}_6$	5000	4200

(a) Data

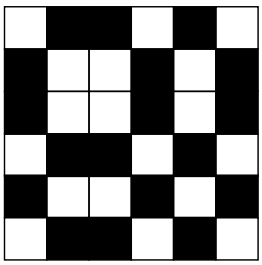
3-$NN(\mathbf{o}_1)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$
3-$NN(\mathbf{o}_2)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_3)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_4)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$
3-$NN(\mathbf{o}_5)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_6)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$

(b) Neighborhood database

Figure 3: An example dataset and its neighborhood database

	$\mathbf{o}_1$	$\mathbf{o}_2$	$\mathbf{o}_3$	$\mathbf{o}_4$	$\mathbf{o}_5$	$\mathbf{o}_6$
3-$NN(\mathbf{o}_1)$	1	0	0	1	0	1
3-$NN(\mathbf{o}_2)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_3)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_4)$	1	0	0	1	0	1
3-$NN(\mathbf{o}_5)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_6)$	1	0	0	1	0	1

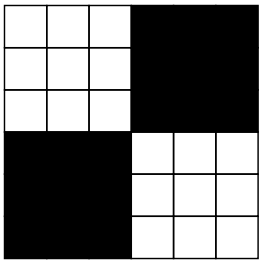
(a) Neighborhood matrix



(b) Visual Figure 4a

	$\mathbf{o}_2$	$\mathbf{o}_5$	$\mathbf{o}_3$	$\mathbf{o}_4$	$\mathbf{o}_1$	$\mathbf{o}_6$
3-$NN(\mathbf{o}_2)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_5)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_3)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_4)$	0	0	0	1	1	1
3-$NN(\mathbf{o}_1)$	0	0	0	1	1	1
3-$NN(\mathbf{o}_6)$	0	0	0	1	1	1

(c) Neighborhood matrix of A_1



(d) Visual Figure 4c

Figure 4: Neighborhood matrices of the example dataset

the cluster structures by finding the repetitive patterns in the neighborhoods [2, 3]. Although the repetitive patterns can be detected by unsupervised tools, selecting the correct parameters may not be trivial. On the other hand, through proper representation, a human can spot repetitive patterns even in noisy settings and provide the intuition that would substantially improve the accuracy and the speed.

3.2 Representation

DEFINITION 5 (NEIGHBORHOOD MATRIX). *A neighborhood matrix is a binary adjacency matrix where columns and rows respectively represent objects and their neighborhoods. If the object in column j is in the neighborhood of object i, then the corresponding cell at i^{th} row and j^{th} column is 1, and 0 otherwise.*

Figure 3a shows a small data set of 6 data objects with 2 attributes. We can identify two cluster structures: $\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$ and $\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$. Figure 4a shows the neighborhood matrix for the 3 nearest neighborhoods of the dataset. While the repetitive neighborhoods are present, it is hard to see and mine. On the other hand, by using the order in the data, we can create a better representation that is easier to interpret and compute. For example, Figure 4c shows the same neighborhoods, where objects and neighborhoods are ordered according to A_1.

The neighborhood representation is compatible with the pixel-based visualization [19]. A *neighborhood matrix* can be represented graphically as an $n \times n$ square, where a pixel is white if the value of the corresponding cell is 1 and black otherwise. Figures 4b and 4d are the pixel-based representation of the neighborhood matrices in figures 4a and 4c, respectively. Cluster structures in the data are stunningly visible in Figure 4d.

Advantages of this representation include: (1) The representation is intuitive. (2) Individual objects are visible, i.e., they can be differentiated from their surroundings, regardless of the local density. (3) Since the repetitive structures stand out, cluster structures are visible. (4) It allows interactive mining since the individual objects are visible. (5) The relations in the representation are explainable because the original attributes are not distorted, e.g., the matrix is not translated as in PCA. (6) Creating the representation is not computationally expensive compared to dimensionality reduction techniques [21]. (7) Compared to graph representation, it provides a scalable and extendible solution, e.g., pixel sizes can be determined by the screen and the data size.

3.3 Mining the Neighborhoods

The evaluation of clusters is an important aspect when finding meaningful clusters. As we discussed in Section 3, objects in the same cluster repetitively co-occur in the neighborhoods of other objects. In this regard, we use the *support* of a cluster as a measure of repetitiveness of an object set in the neighborhood database. Therefore, clusters can be detected by finding the object sets that have a high *support*.

DEFINITION 6 (SUPPORT(σ)). *The support of an object set, i.e. cluster, is the number of neighborhoods in which the objects occur together. Formally,*

$$\sigma(C) = |\{\mathbf{o} \mid C \subseteq N(\mathbf{o}), N(\mathbf{o}) \in \mathscr{ND}\}|$$

Support is a useful measure also for interactive mining. As shown in Section 4, supports of an object set in different neighborhood databases give an overview of the cluster formations. Moreover, this relation between the clusters and the support in the neighborhoods, can directly be mapped to frequent itemset mining [1], so that the whole literature of frequent itemset mining methods becomes available for cluster analysis [3].

One clear advantage of using neighborhood databases is their unifying representation. All of the $\mathscr{ND}$s, regardless of the underlying (dis-)similarity measure, share the same properties, which means less context switches and more clarity for the user. For example, consider these measures: Euclidian distances on subsets of attributes, cosine similarity on all numeric attributes, a measure for the boolean attributes, two different measures for the same categorical attribute. $\mathscr{ND}$s for each of them can be mined with the same

set of tools because we are looking for the same kind of information: the objects that frequently co-occur in the neighborhoods.

As we discuss in Section 2, for high-dimensional data, local similarities are more meaningful than similarities in the whole data space. Therefore, bottom-up search is a widely used strategy to find subspace clusterings [22, 20]. In this regard, we propose to start the interactive analysis with the neighborhoods in 1 dimensional projections and find the object sets that repetitively co-occur in the neighborhoods in different projections.

Our tool, cf. Section 4, includes two methods for unsupervised mining of neighborhood databases, both of which satisfy the time constraints of an interactive setting. ***Sampling Miner*** mines the dataset for a subset of all the frequently co-occurring object sets. It is based on a Monte Carlo process, and as such, it mines a predetermined number of maximally large object sets that has more support than a certain threshold. These object sets are counterparts of *maximal frequent itemset*s [5]. Although it is not guaranteed to be complete, it produces satisfactory results [3]. ***Fast Miner*** exploits the orders in an attribute to find the complete cluster structure [2]. It starts by mining the individual $\mathscr{ND}$s for a complete set of 1 dimensional clusters. Then, each cluster is refined further by checking whether any of its subsets are clusters in other dimensions. *Fast Miner* can only be used to mine neighborhoods of univariate measures.

Typically, subspace clusters overlap with each other both object-wise and attribute-wise. For example, a set of objects can form a cluster in dimensions 1 and 2 while another, but not necessarily disjoint, set of objects can form a cluster in dimensions 2 and 3. Therefore, the relation between the attributes should be investigated by assessing localities instead of the whole domain. We propose to use micro clusters to find the similarities between attributes. These kind of micro clusters are used to detect cluster structures [4, 15]. We mine a sample of micro clusters of size 5 in each attribute. The number of micro clusters shared by a pair of attributes becomes their similarity score. If a set of attributes are similar to each other, it can be worthwhile to investigate the neighborhoods of the combination of these dimensions. Even though the complete cluster structures are not visible in projections, micro clusters can effectively catch and aggregate local similarities, as we will show in Section 5.2.

Although sorted and non-sorted neighborhoods essentially contain the same information; in a visual setting, it is often easier to work with sorted neighborhoods. Unfortunately, sorted neighborhoods are possible only for univariate (1 dimensional) projections. For the interactive setting in VINeM, we approach the problem by partially sorting the $\mathscr{ND}$ by focusing on a subset of the objects. The partial sorting is done by selecting an object as a reference and sorting the remaining objects and neighborhoods according to their similarities to the reference object. As we show in Section 5.2, partial sorting provides enough visual information for the identification of cluster structures.

Even if there are no cluster structures in the data, there is a minimum amount of repetition in the neighborhoods. In the case of uniformly distributed data, each individual object appears in exactly k neighborhoods, while the consecutive object sets of size $\frac{k}{2}$ appear in exactly $\frac{k}{2}$ neighborhoods. Neighborhoods for a uniform dataset are shown in Figure 5. This observation is used as a key indicator to discard the projections that do not have cluster structures.

Outliers and noise objects exist in relatively sparse areas and do not occur frequently in neighborhoods of objects [28]. Therefore, noise can be easily identified in the neighborhood database as the objects with low support.

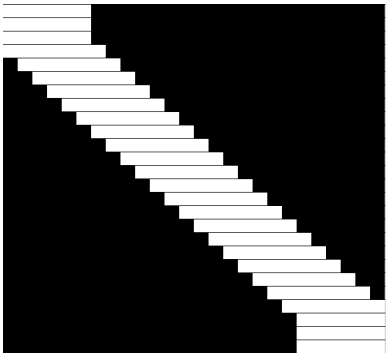


Figure 5: Neighborhoods of uniform data

If a set of objects form cluster structures in multiple attributes, then their values are similar to each other in these attributes. Therefore, we can use the dispersion of an object set in an attribute as an heuristic for cluster formation. Standard deviation and median absolute deviation (MAD) are widely used measures for dispersion. According to our observation, MAD gives more accurate results than the standard deviation.

4. VISUALIZATION

In this section we introduce VINeM, a data exploration tool that exploits the neighborhood database representation. An important design goal of VINeM is to provide a user-friendly interface that allows a user to easily blend unsupervised tools with instant decisions. Therefore, all of the features are designed for human interaction while they can be manipulated by the unsupervised tools.

VINeM is implemented in Java using Swing as GUI toolkit. It is accessible along with a detailed user manual on the supplementary website.[1]

4.1 Neighborhood

The main interaction area of VINeM is the neighborhood panel where the neighborhood matrix is shown, cf. Figure 6. As discussed in Section 3.2, columns represent the objects and rows represent the neighborhoods.

Currently, two kinds of neighborhoods are supported: ε-neighborhood and k-nearest neighborhood. The *kNN* representation is the default because of its robustness. VINeM starts by showing the *kNN* neighborhood databases for each of the individual attributes. Initially, the matrix and the objects are sorted according to the shown attribute. The order of the objects can be observed in the selection list where the ids, or the names, of the objects are shown, cf. 4 in Figure 7.

The matrix representation can be manipulated using the following means:

Dissimilarity measures for the projections. In the initial setting, there is one neighborhood matrix per attribute, each of which represents the neighborhoods according to Euclidian distances per dimension. Which $\mathscr{ND}$ to show can be selected by using dropdown, cf. 10 in Figure 7.

Object and neighborhood order. The order of the objects is not necessarily dependent on the dissimilarity measure. For example, it is possible to sort the neighborhoods in attribute 1 according to attribute 2. On the other hand, the order and the measure are synchronized by default for convenience. The dimension that is used

[1] `http://adrem.uantwerpen.be/vinem`

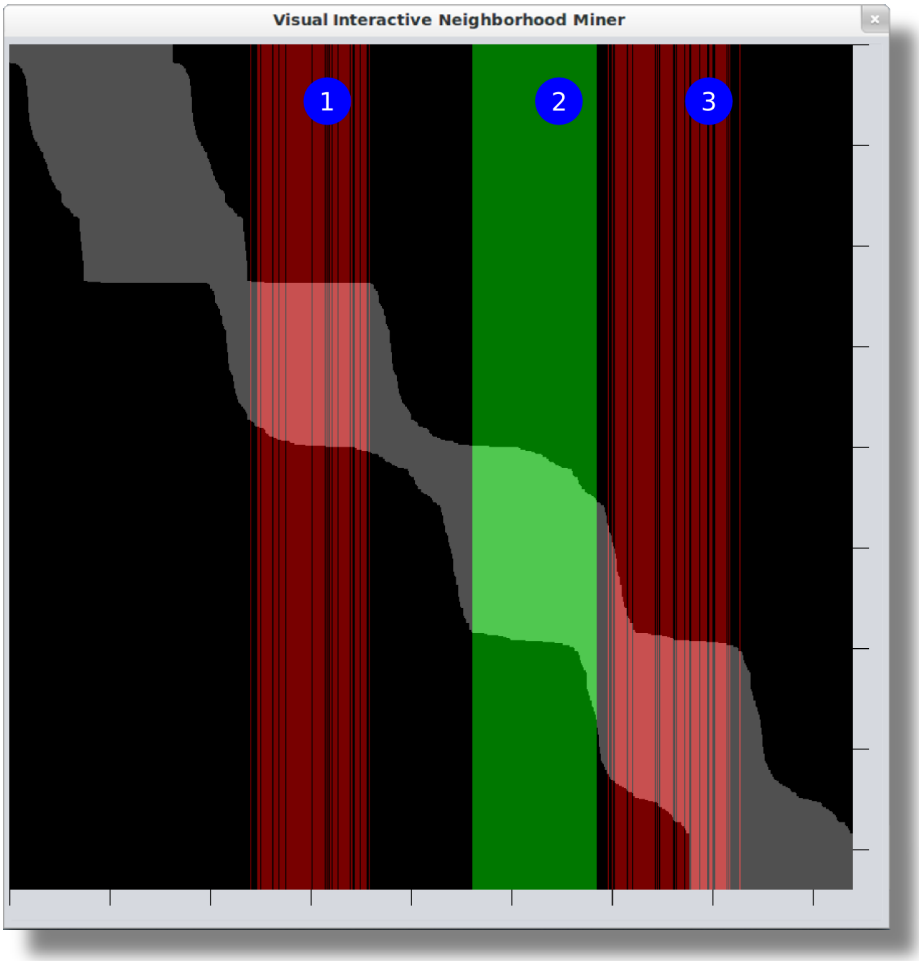


Figure 6: Neighborhood matrix

to order the objects can be selected using a slider, cf. 8 in Figure 7. If the synchronization is enabled, this will also change the dissimilarity measure.

The type of the neighborhood assessment, ε vs. kNN. The parameter for neighborhood size (k) can be set for *kNN*, while the parameter for neighborhood radius (ε) is required for the ε-Neighborhood. For convenience, possible radius sizes are pre-computed per similarity measure and user selects among them. The type of the neighborhood and its parameters can be selected on the control panel, cf. 7 in Figure 7.

4.2 Interactivity

Interactivity of VINeM starts with parameter selection. The interface and the representation are updated instantly after each parameter change, allowing the user to experiment with parameters on a responsive interface.

The selection of objects is the first step of the interactive analysis. There are two intuitive ways to select objects: (1) Dragging the left mouse button on the neighborhood matrix highlights the columns in green and selects the corresponding objects, cf. 2 in Figure 6. (2) Selecting objects from the selection list combo box by using standard list selection techniques, cf. 4 in Figure 7. The main selection method is dragging on the matrix, the list selection is for fine tuning. There are three modes for the selection on matrix, cf. 5 in Figure 7. In "Select" mode, only the objects under the mouse are selected. The "And" mode selects the objects if they are already selected while the "Or" mode adds the new selection to the already selected objects.

A separate frame, cf. Figure 8, shows the information about the selected objects, such as the size of the selection, their support and dispersion in other $\mathscr{ND}$s. This information is used to determine the next $\mathscr{ND}$ to investigate. Selection of the objects is an essential part of the data analysis in VINeM. During the analysis, the selection is iteratively refined by removing the objects that do not belong to the cluster in the additional attributes, which results in a selection of similar objects according to some attributes, i.e., a subspace cluster.

Selected objects can be identified as a cluster by clicking the button "Cluster Selected", cf. 5 in Figure 7. The clusters that are identified either by the user or by an unsupervised tool are shown in the cluster list window, cf. Figure 9. Any of the identified clusters can be visualised on the neighborhood matrix along with the selected objects, so that the selection can be compared with the known clusters. Clusters are highlighted in red on the neighborhood matrix, cf. 1 and 3 in Figure 6. Clusters can be manipulated by adding or removing objects. If a substantial refinement is

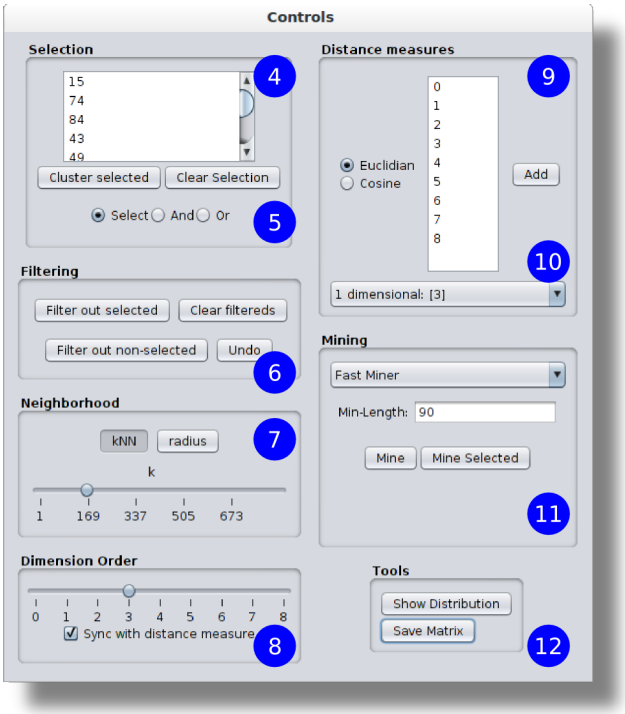


Figure 7: Control Panel

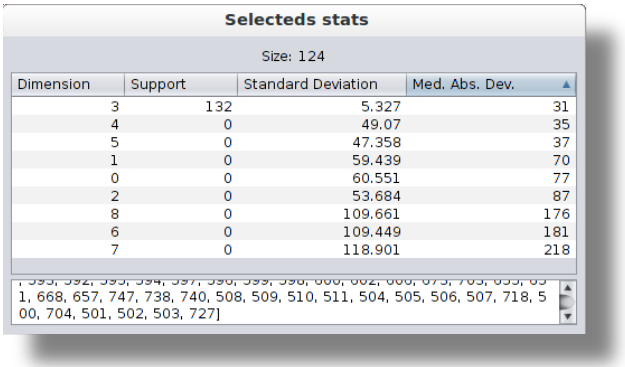


Figure 8: Basic information about the selection

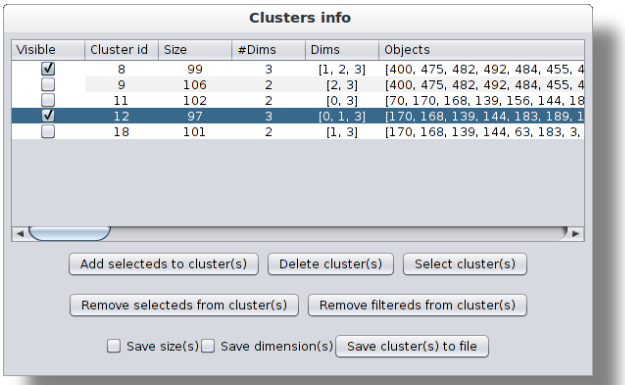


Figure 9: List of detected clusters

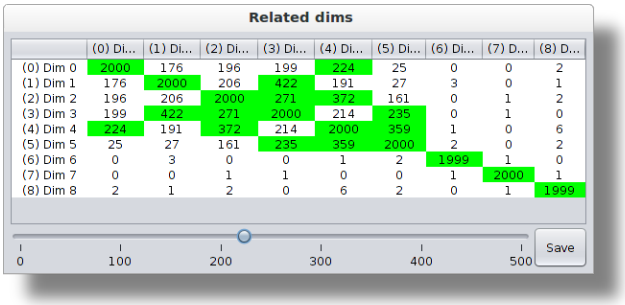


Figure 10: Relevancy score of dimensions

required, they can always be converted into selections. Any subset of clusters can be saved to a file for further study.

A set of objects can be filtered out, i.e. removed, from the view to focus on the objects under consideration, cf. 6 in Figure 7. Note that filtering removes the objects and their neighborhoods just from the view, i.e., the neighborhoods are not re-computed. Therefore, it is fast and it does not change the data on the fly. Filtering comes in handy while performing a cross measure analysis since it hides the noise caused by the projection.

Neighborhood databases for new measures can be added by just a few mouse clicks using the GUI. To add an $\mathcal{ND}$, a similarity measure is selected along with the dimensions on which it will be applied, cf. 9 in Figure 7. After adding the new $\mathcal{ND}$, it can be viewed by selecting from the drop-down menu, cf. 10 in Figure 7.

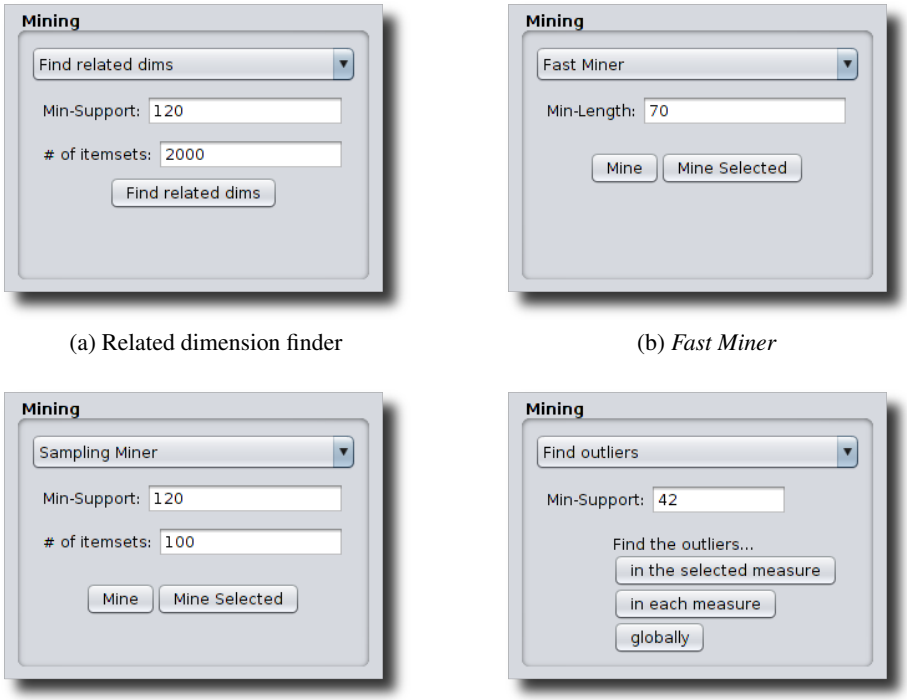
$\mathcal{ND}$s of non-univariate similarity measures can be partially sorted. Right clicking on a column selects the corresponding object as the reference and sorts the other objects according to their similarities to this object, cf. Section 3.3.

4.3 Unsupervised Tools

VINeM is bundled with unsupervised tools to support the user during the data analysis, cf. 11 in Figure 7. They are seamlessly integrated with the interactive tools where applicable.

Related dimension finder mines the $\mathcal{ND}$ of each individual attribute for micro clusters as explained in Section 3.3. Parameters are updated with suggested values when the $\mathcal{ND}$ is modified, cf. Figure 11a. Relevancy scores between pairs of attributes, i.e. the number of shared micro clusters, are shown as a table. A thresholding slider is provided for visual assistance on detecting the highly related dimensions, cf. Figure 10.

The two miners that are introduced in the Section 3.3, namely *Fast Miner* and *Sampling Miner*, can be used for unsupervised mining. While *Fast Miner* requires only one parameter which is the minimum length of a cluster, cf. Figure 11b; *Sampling Miner* requires two parameters: required minimum support of an object set to be identified as a cluster and number of samples, cf. Figure 11c. Both of the miners can be run either on the whole dataset or only on the selected objects, so that the objects that are not under consideration can be left out. The suggested values for the parameters are



(a) Related dimension finder

(b) *Fast Miner*

(c) *Sampling Miner*

(d) Outlier detection

Figure 11: Parameters for miners

provided for a quick start. After the run, all of the found clusters are added to the cluster list for further investigation.

Automated noise/outlier detection is also provided as a miner. The objects that have a support less than the threshold are identified as noise. The support of objects can be evaluated (1) *in the selected measure*, (2) *in each measure*, or (3) in *all measures*, cf. Figure 11d. Noise objects are added as a cluster. It is up to the user to filter them out of the view.

5. APPLICATION

5.1 Finding clusters in subspaces

Figure 12 shows the steps of an interactive mining process. We start by finding a neighborhood size that makes the clusters visible in one of the $\mathscr{N}\mathscr{D}$s. In this example, we can see the cluster structures in Dim(ension) 1 for a certain k value (Step 1). We identify a one dimensional cluster by selecting its objects (Step 2). The next step is to check other dimensions to find out whether any subsets forms a cluster there. When we switch to the $\mathscr{N}\mathscr{D}$ of the Dim 2, objects in the cluster are still marked with red (Step 3).

Since we are looking for subsets of our cluster, we remove the remaining objects by filtering them out (Step 4). We decide that the objects do not form cluster structures in Dims 2 and 3 because their filtered $\mathscr{N}\mathscr{D}$s look like the $\mathscr{N}\mathscr{D}$ of a uniform distribution, which is shown in Figure 5 (Steps 4 and 5). Note that, although there are some repetitive neighborhoods in the filtered $\mathscr{N}\mathscr{D}$ of Dim 2, they are too small to be identified as clusters. Filtered $\mathscr{N}\mathscr{D}$ of Dim 4 looks interesting (Step 6), there is a large set of objects that co-occur in neighborhoods. We select these objects and identify them as a cluster in Dimensions 1 and 4.

5.2 Finding relevant dimensions

Figure 13 shows the steps for an exploratory analysis on a 10 dimensional dataset. In this dataset, the clusters are not immediately visible in individual dimension projections. Step 1 of the figure shows 4 of the $\mathscr{N}\mathscr{D}$s. There are signs of structures in Dims 0 and 2, which can probably be enhanced by modifying the neighborhoods size, while Dims 4 and 6 look like they lack any kind of structure. It is possible that the whole cluster structures are not visible in one dimensional projections. Running the *related dimension finder* gives us the scores for each pair of dimensions. Then, we can interactively decide which of the dimensions are related to each other by examining the high scores (Step 2). It looks like the two sets of dimensions {0, 1, 2, 3} and {4, 5, 6, 7} share common structures.

With a few clicks, we add a new dissimilarity measure that assess the neighborhoods in the combination of dimensions 4, 5, 6, and 7 (Step 3). Although it is almost impossible to spot the structures in the $\mathscr{N}\mathscr{D}$ of the combined dimensions (Step 4), sorting the neighborhoods according to a reference object, i.e. partial sorting in Sec. 3.3, helps us to see the structure in the data (Step 5). The blob on the top left of the sorted $\mathscr{N}\mathscr{D}$ represents a large set of objects that co-occur in the neighborhoods. We select these objects for further analysis (Step 6). Note that, since this sorting is according to only one object, some objects that are not inside the cluster can be in the blob by mistake, and they can be removed by using the partial orderings of the objects in the blob. We investigate further by sorting the $\mathscr{N}\mathscr{D}$ according to an object that is not in the cluster candidate (Step 7), and now we can see some other structures in the $\mathscr{N}\mathscr{D}$. We identify the selection as a cluster (Step 8), and then we continue our analysis with selecting a new cluster candidate. We can still modify the neighborhood parameters to improve the view. For example, cluster structures are more visible in Step 8 compared to Step 7, because of using ε-neighborhoods instead of *kNN*.

6. CONCLUSION

While visualisation and interactiveness are very important for exploratory data analysis, available tools fall short to satisfy all of its challenges. In this paper we show that local neighborhoods provide the means for both intuitive visualization and interactive mining of subspace clusters. We introduce VINeM, a platform-independent, visual and interactive data analysis tool that exploits the intuitive neighborhood-based representation to seamlessly combine a user friendly interactive interface with the unsupervised tools. We introduce a micro cluster based tool to find relevant dimensions and we provide example scenarios of exploratory data analysis to show the usefulness of our application.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, volume 1215, pages 487–499, 1994.

[2] E. Aksehirli, B. Goethals, and E. Müller. Efficient cluster detection by ordered neighborhoods. In *2015 17th International Conference on Big Data Analytics and Knowledge Discovery (DaWaK) (accepted)*, 2015.

[3] E. Aksehirli, B. Goethals, E. Müller, and J. Vreeken. Cartification: A neighborhood preserving transformation for mining high dimensional data. In *2013 IEEE 13th International Conference on Data Mining (ICDM)*, pages 937–942, Dec. 2013.

[4] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(6):937–965, 2005.

[5] R. J. Bayardo, Jr. Efficiently mining long patterns from databases. *SIGMOD Rec.*, 27(2):85–93, June 1998.

[6] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Database Theory–ICDT'99*, pages 217–235. Springer, 1999.

[7] M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. Wrobel. One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD*, IDEA '13, pages 27–35, New York, NY, USA, 2013. ACM.

[8] S. Bremm, T. von Landesberger, M. He\s s, T. Schreck, P. Weil, and K. Hamacherk. Interactive visual comparison of multiple trees. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 31–40. IEEE, 2011.

[9] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000.

[10] T. F. Cox and M. A. Cox. *Multidimensional scaling*. CRC Press, 2010.

[11] B. Goethals, S. Moens, and J. Vreeken. MIME: a framework for interactive visual pattern mining. In *Proceedings of the 17th ACM SIGKDD*, pages 757–760. ACM, 2011.

[12] M. Goldstein. k_n-nearest neighbor classification. *Information Theory, IEEE Transactions on*, 18(5):627–630, 1972.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[14] T. Hey, S. Tansley, and K. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Redmond, Wash., 1 edition edition, Oct. 2009.

[15] P. Hu, C. Vens, B. Verstrynge, and H. Blockeel. Generalizing from Example Clusters. In J. Fürnkranz, E. Hüllermeier, and T. Higuchi, editors, *Discovery Science*, number 8140 in Lecture Notes in Computer Science, pages 64–78. Springer Berlin Heidelberg, Jan. 2013.

[16] A. Inselberg and B. Dimsdale. Parallel coordinates. In *Human-Machine Interactive Systems*, pages 199–233. Springer, 1991.

[17] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, June 2010.

[18] R. Jarvis and E. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, C-22(11):1025 – 1034, Nov. 1973.

[19] D. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, Jan. 2002.

[20] H.-P. Kriegel, P. Kröger, M. Renz, and S. Wurst. A generic framework for efficient subspace clustering of high-dimensional data. In *Fifth IEEE International Conference on Data Mining*, pages 8 pp.–, Nov. 2005.

[21] L. Maaten. Learning a parametric embedding by preserving local structure. In *International Conference on Artificial Intelligence and Statistics*, pages 384–391, 2009.

[22] G. Moise and J. Sander. Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 533–541, New York, NY, USA, 2008. ACM.

[23] E. Müller, S. Günnemann, I. Assent, and T. Seidl. Evaluating clustering in subspace projections of high dimensional data. *Proceedings of the VLDB Endowment*, 2(1):1270–1281, 2009.

[24] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[25] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec. 2000. PMID: 11125150.

[26] J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results [gene identification]. *Computer*, 35(7):80–86, July 2002.

[27] J. Seo and B. Shneiderman. A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 65–72. IEEE, 2004.

[28] B. Sluban, M. Juršič, B. Cestnik, and N. Lavrač. Exploring the power of outliers for cross-domain literature mining. In *Bisociative Knowledge Discovery*, pages 325–337. Springer, 2012.

[29] A. Tatu, F. Maas, I. Farber, E. Bertini, T. Schreck, T. Seidl, and D. Keim. Subspace search and visualization to make sense of alternative clusterings in high-dimensional data. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 63–72. IEEE, 2012.

[30] J. B. Tenenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 2000. PMID: 11125149.

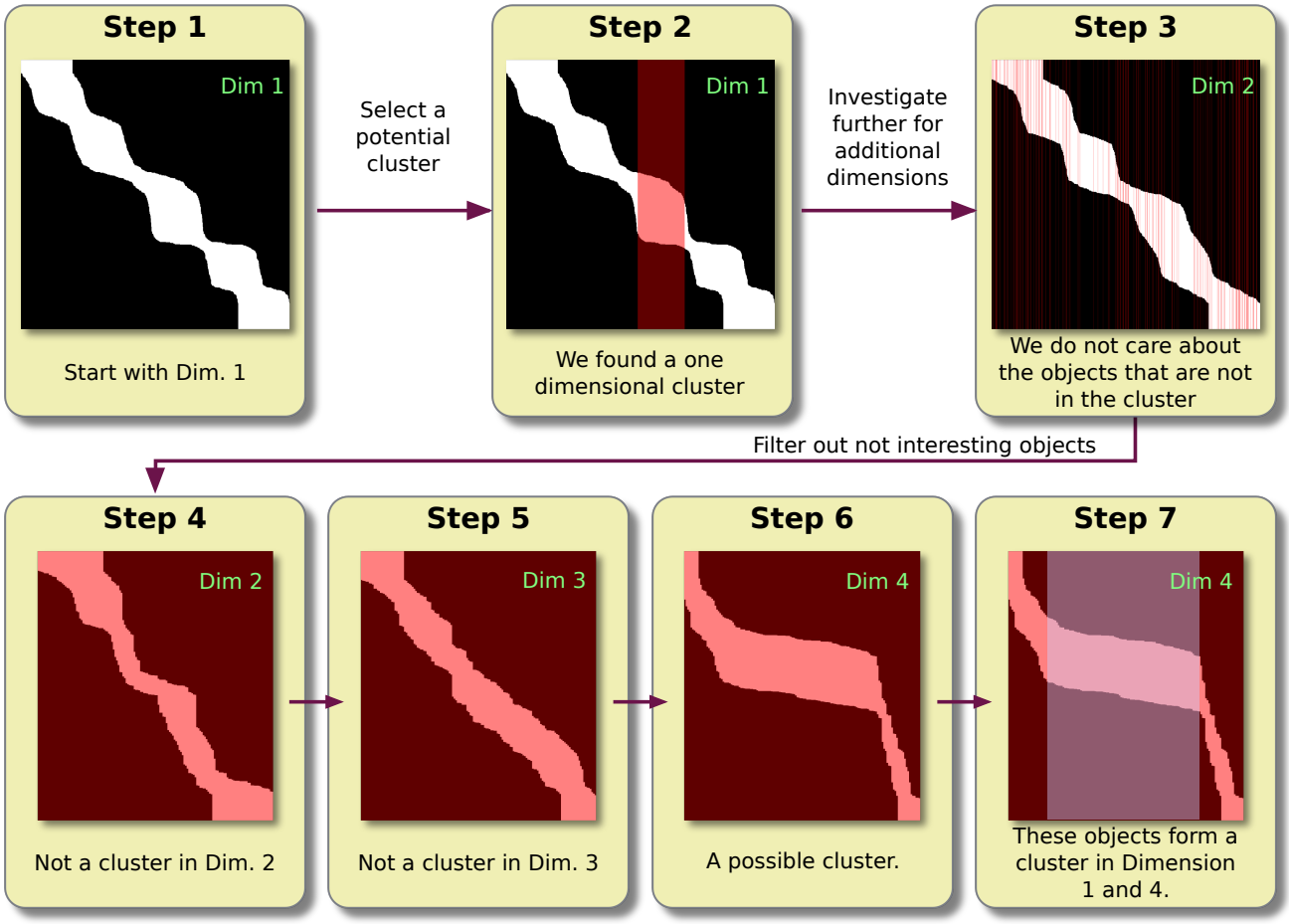


Figure 12: Subspace cluster detection

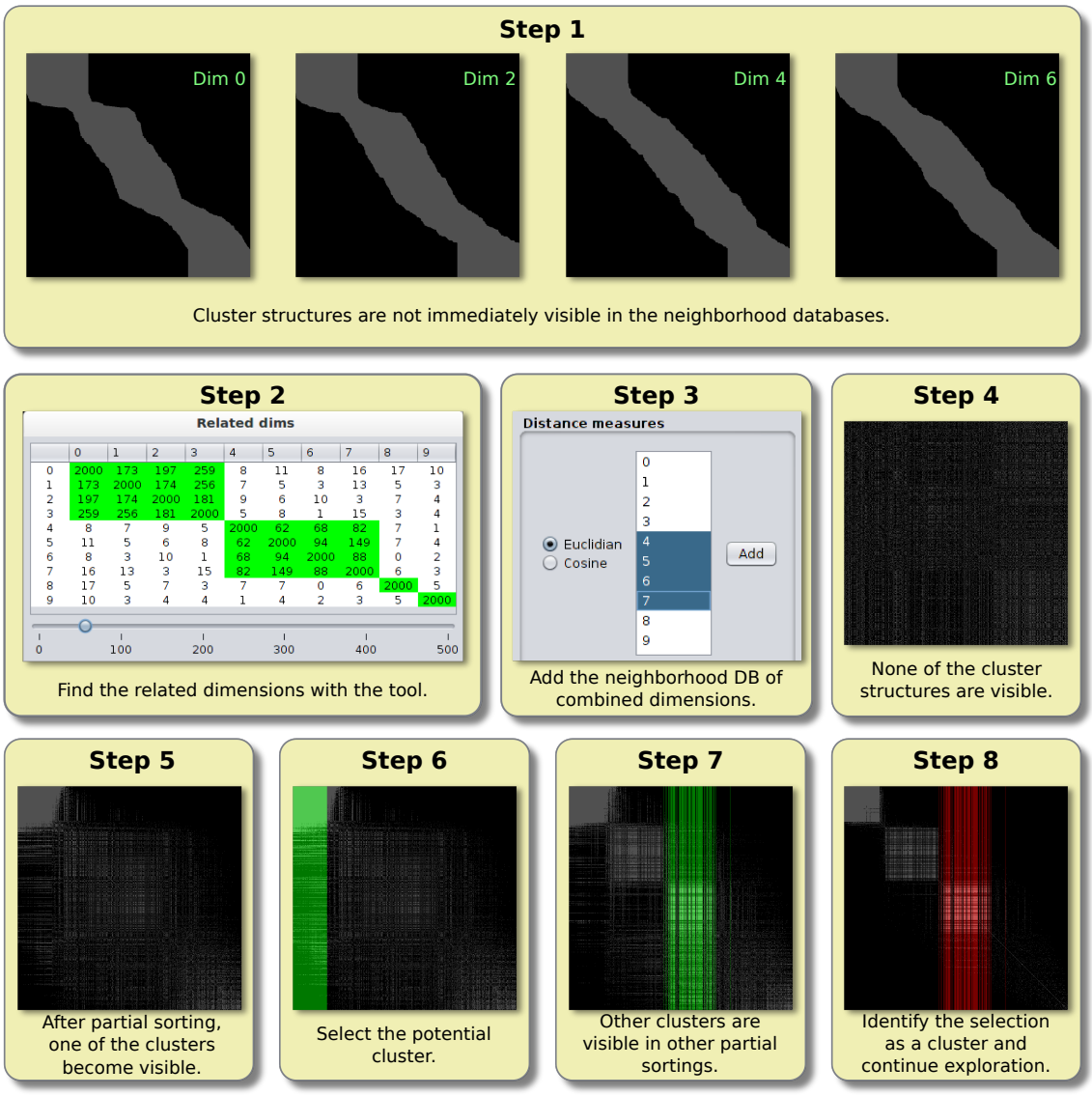


Figure 13: Finding relevant dimensions for cluster detection

Creedo—Scalable and Repeatable Extrinsic Evaluation for Pattern Discovery Systems by Online User Studies

Mario Boley, Maike Krause-Traudes, Bo Kang and Björn Jacobs
University of Bonn and Fraunhofer IAIS
Schloss Birlinghoven, Sankt Augustin, Germany
mario.boley@gmail.com

ABSTRACT

We present Creedo—a conceptual framework along with an accompanying software implementation for empirically evaluating knowledge discovery systems. Creedo provides a language for the creation of study designs that test how well different test systems support real users to perform certain data analysis tasks. These designs are scalable and repeatable, i.e., after their creation, a study can be carried out any number of times and with an arbitrary high number of participants without consuming valuable resources such as the time of supervision personnel. Building on the conceptual framework, the accompanying web application, which is freely available at Bitbucket, supports data mining researchers in all central tasks for conducting a user study: in embedding their ideas into an operational data analysis environment, in assigning and monitoring tasks for study participants, and in evaluating the results. The implementation as web application enables large scale and geographically distributed studies, in which, nevertheless, all study participants essentially have an identical user-experience.

1. INTRODUCTION

In this paper we present Creedo—a software-supported framework for conducting empirical user studies with the purpose of evaluating scientific contributions in pattern discovery [Hand, 2002] and related areas of knowledge discovery from data. Although the user plays a central role already in the earliest scientific definitions of the knowledge discovery process [Fayyad et al., 1996], contributions to the field are traditionally supported by formal or empirical evaluations that replace the user by a set of abstract assumptions about her—most often by the single simple assumption that users like to see analysis results as fast as possible. Hence, the vast majority of research articles focus on computation speed as evaluation metric while the usefulness of the computation output is a mere postulate.

There is, however, an increasing trend to investigate claims that directly contain the data mining user as a prime subject. Examples include the work of De Bie and Spyropoulou [2013] which, beyond just asking "how one can formalize a nugget of information", also deals with the question of "how one can formalize how interesting such a nugget of information is to *a particular user*", the work of Dzyuba et al. [2014] that aims at enabling the use of pattern discovery algorithms to *non-expert users*, as well as other, including our own, work on the same subject [Xin et al., 2006, Boley et al., 2013]. It is often hard to support direct claims about user satisfaction solely based on a certain set of axioms about the user: there might be no widely accepted set of such axioms relevant to the specific claim, and, moreover, usually the very same assumptions used to evaluate a contribution were also used to develop it in the first place. Where such intrinsic evaluations reach their limits, empirical studies that involve real users could be a powerful alternative, because they constitute an evaluation *extrinsic* from the simplifying development assumptions. Nevertheless such studies are conducted only rarely (see Ke et al. [2009], Shahaf and Guestrin [2010], Li et al. [2012] for some notable exceptions).

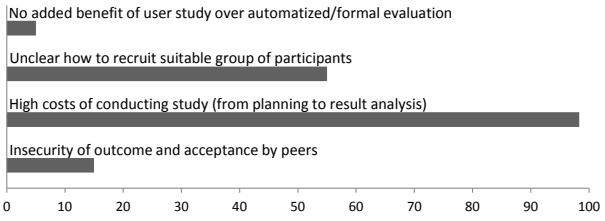


Figure 1: Reasons reported by ECMLPKDD authors to not conduct a user study in cases where such a study could have been beneficial (in percent of respondents).

One might argue that this reluctance to engage in user studies is due to the fact that the vast majority of data mining authors does not share the feeling that such studies could be beneficial to support the value of their contributions. However, in a recently conducted online poll among ECMLPKDD authors [Boley et al., 2015], 50% of 135 respondents reported that they decided not to conduct a user study within the last two years *although* their work could have benefited from it[1]. Moreover, those respondents al-

[1]This result held almost uniformly across different subfields: Machine Learning authors had only little less "yes"-answers (49%) than the (non-disjoint) group of authors who work in pattern mining and visual analytics (56%).

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

most unanimously pointed to the high costs involved as at least one of the reasons for their decision (see Fig. 1). Indeed, even after the already challenging task of gathering a group of relevant study participants, there are several high-cost issues that remain to be addressed: The work has to be embedded into an operational data analysis environment that a) has a user interface that the participants can operate and that b) allows to perform all measurements necessary for performance assessment. Then, the trials that form the basis for performance measurements have to be carried out, which includes splitting participants into several groups, e.g., corresponding to test and control, assigning the right participants to operate the right data analysis environment under the right instructions, and, most importantly, recording all variables relevant to the hypothesis that abound from these trials. Finally, study hypotheses can refer to humans not only in an acting capacity, i.e., as subject operating a data analysis tool, but also in an evaluating capacity, i.e., as providing certain quality assessments, e.g., whether patterns are interestingness or useful within the context of a specific task. Also these evaluations have to be carried out.

Presently, if a researcher wants to tackle all of these steps, she finds herself only insufficiently supported by current software tools. The accessibility aspect of embedding could be reached by creating a plug-in for a general purpose data mining suite like Rapid Miner [Mierswa, 2009] or WEKA [Hall et al., 2009]. However, the resulting software system would not be connected to a study infrastructure that can automatically extract relevant measurements. Some aspects of testing could be addressed by using a general purpose online survey framework like Survey Monkey [sur] or KwikSurveys [kwi]. However, these frameworks are oblivious to the semantics of data analysis and would only allow performance measurements based on indirect questionnaires rather than directly tracking user actions. Ultimately, all current solutions lack a conceptual framework for defining user studies that tie together data analysis systems, analysis tasks, and performance measurements.

To the best of our knowledge, such a framework is given for the first time by Creedo. Creedo provides a language for the specification of study designs that test how well certain data analysis systems support real users to perform certain data analysis tasks based on certain evaluation criteria. Studies based on these designs can be repeated any number of times as long as a suitable pool of study participants is available. Corresponding to the goal of extrinsic evaluations, Creedo studies can not only involve human test participants, but also the evaluation critera can involve human evaluators. In order to enable scalable studies, Creedo allows users to act in the double role of participant and evaluator while providing assignment schemes that allow to control potential biases that could otherwise arise from such double roles. Building on this conceptual framework, the Creedo web application supports data mining researchers in performing embedding, testing, and evaluation as defined above in a convenient and integrated way. It provides simple reusable UI elements that can easily be extended and can be combined to form interactive data analytics dashboards. Moreover, it allows to rapidly design, deploy, and conduct Creedo studies involving those dashboards as test systems. The application automatically performs all task assignments to participants and evaluators, who can then carry out their tasks using any standard web browser.

2. FRAMEWORK OUTLINE

Creedo allows to specify an interaction process during which a group of users jointly provides evidence in favor or against a test hypothesis. Particularly, the specific kind of **hypothesis** we are interested in can be paraphrased as follows:

> *"Users can solve a certain class of analysis tasks better with a specific target system than with other control systems."* (1)

In order to break down the rather complex interaction process evolving around testing such a hypothesis, let us begin by distinguishing between different roles in which users are acting within this process and different time periods in which certain actions are performed. Chronologically, the hypothesis validation starts with the **study design time** where the **study designer** translates the test hypothesis into an executable study design. This is followed by the **study execution time** where all the actual measurement data is accumulated. This happens for once through study **participants** who engage in trial **sessions** with data analysis systems, in which they produce some tangible output that we refer to as session **results**. And this also happens through **evaluators** who review those results and produce **evaluations** of them. Finally, there is the **study conclusion time** where the data is processed in order to provide evidence in favor of or against the hypothesis to the study designer. Note that it is a central feature of Creedo studies that users can act in more than one role within a given study. In fact it can be very crucial for a study to be scalable that all the participants also act as evaluators.

As we can see, the central element of a Creedo study is the **study design**, which corresponds to a specific test hypothesis. More specifically, it can be considered as an *operationalization* of the natural language version of the hypothesis. That is, a study design gives a measurement procedure that can be practically carried out in order to assess all quantities necessary to evaluate the hypothesis. Once created, a study design can be used for an arbitrary number of actual studies, each of which corresponds to a measurement according to the design or, from an alternative point of view, to the realization of a random variable defined by the design (where the randomness would typically involve the selection of users). Thus, by "repeating a study" we actually refer to the process of running a new study with an identical study design as used by a previous study but with a new set of users. A typical motivation for this would be to increase the level of confidence in a result.

A Creedo study design translates an hypothesis following the blueprint given in (1) by specifying the following set of components (see also Fig. 2):

1. a set of system specifications, corresponding to the "target system" and the "control systems" referred to in the hypothesis,

2. a set of task specifications that represent the "certain class of analysis tasks", what it means to "solve" them as well as what are the basic qualities of task results,

3. an assignment logic that specifies how assignments are issued in a study to its users in order to generate all required measurements, and finally

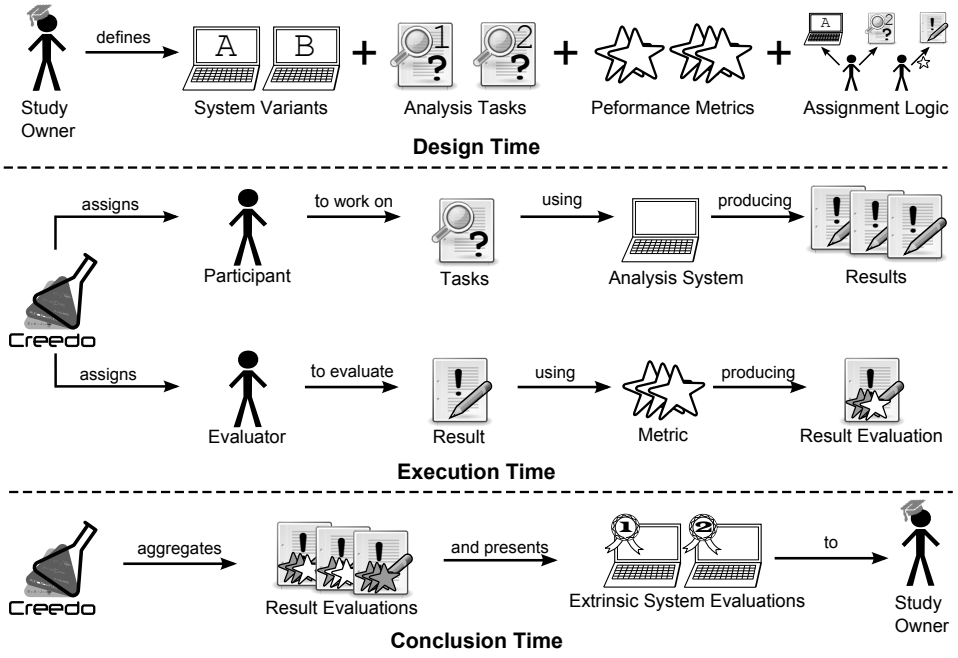


Figure 2: Outline of elements and process of a Creedo study.

4. system performance metrics that formally define the meaning of "better" in terms of the basic result qualities given by the task specifications.

While the conceptual framework of Creedo is more abstract than its implementation in the Creedo web application, some of the concrete options that are available for the design components above are determined by the implementation. Hence, before we investigate the detailed composition of study designs (Sec. 4), we will first review the Creedo web application (Sec. 3). After that, in order to illustrate all previously introduced concepts, we will present a study design that was developed to evaluate the FORSIED framework (Sec. 5).

3. CREEDO WEB APPLICATION

The Creedo web application [cre] is a Java server application that allows to define and execute study designs. It can be installed to run on a central web server, from which study participants and evaluators can receive their assignments in the form of interactive web pages. These pages are very light-weight. All actual computation is carried out on the server and all content-relevant state like the input data is also kept there. The motivation for this architecture is two-fold: Firstly, it enables large scale and geographically distributed studies, in which, nevertheless, all study participants essentially have an identical user-experience: Every computer with access to the server can potentially be used as an input terminal, as long as all used computers have the same input/output devices and their hardware is otherwise sufficient to fluently display the pages in one of the standard web browsers. Secondly, all study-related definitions and statistics are persistent in one centralized location from which study execution can be controlled consistently.

The data analysis systems that are examined in a study executed by the web application correspond to Creedo analytics dashboards. These are a visual user front-end of an interactive and iterative knowledge discovery process that builds on our open-source Java library realKD [rea]. This library is designed to allow easy and straightforward implementations of pattern discovery algorithms by providing a wide range of pattern discovery primitives and an expressive data model. Thus, extending realKD provides a convenient way to embed a novel algorithmic idea into Creedo analytics dashboards. The server side state of the dashboard is given by a realKD data workspace, which contains all the data artifacts available for analysis, and by the state of optional UI components that can be added to a dashboard. These components allow to perform certain data analysis activities like data inspection, data mining, and post-processing. The state of all components can be accessed to define concrete study concepts (e.g., result definitions, performance metrics).

The components that are implemented at the time of writing this paper support an interactive pattern discovery process, in which the user can run mining algorithms, inspect their results in the context of the given input data, potentially choose some of the produced patterns in a result collection, and then re-iterate. In the following we introduce some of these components together with the part of their state that is relevant to study definitions (see Fig. 3 for a screenshot of an analytics dashboard containing all components presented here).

For the inspection of data artifacts, the **data view container** can aggregate a number of data view sub-components, each of which is able to represent a different kind of data or to provide an alternative view on the same data. These subcomponents currently include: A data table view for rendering tabular data composed of numeric, ordinal, and categorical attributes, a point-cloud, which renders a 2-dimensional

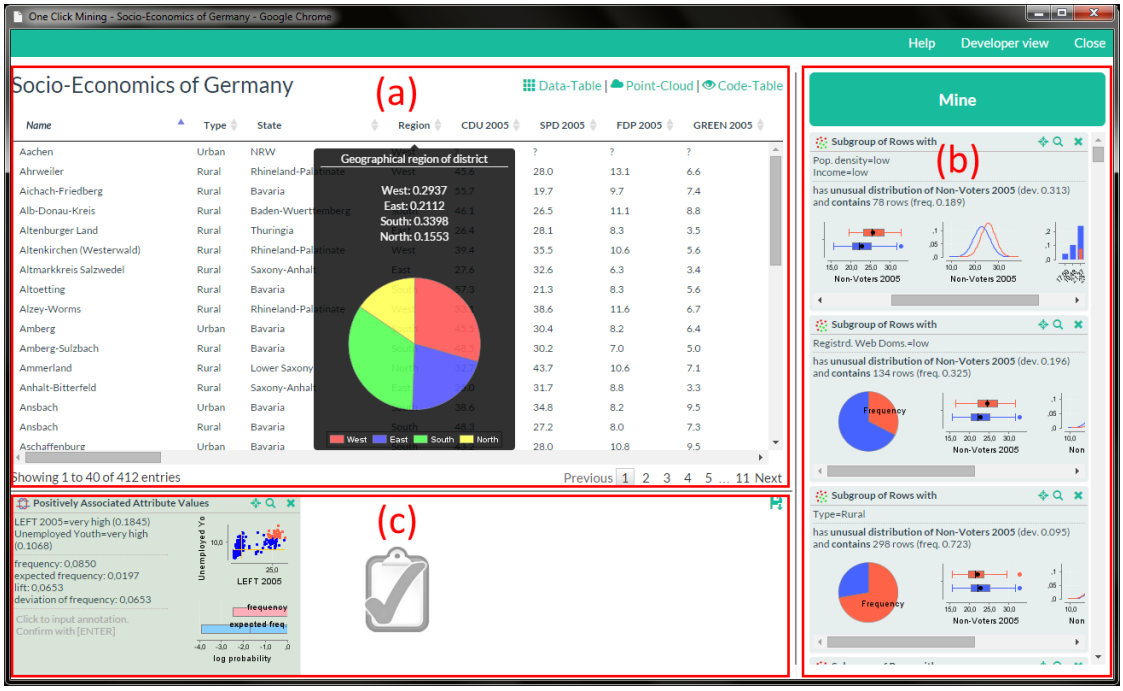


Figure 3: Creedo analytics dashboard with data view container (a), pattern mining launcher (b), and KD result container (c).

embedding of the data, and a propositional logic view, which renders a collection of propositional statements defined on rows of a data table (as required by pattern discovery algorithms that work on binary data).

The component that allows to perform the actual data mining is referred to as the **pattern mining launcher**. It enables the user to run pattern discovery algorithms on all data-artifacts present in the system and to display their results. The set of available algorithms can be chosen by the system designer from all realKD mining algorithms. In order to launch one of the algorithms, the user first has to click on a mine button, then choose the specific algorithm she wants to use, and finally select valid settings for all the parameters of that algorithm (the last two steps are only required when more then one algorithm is available and the selected algorithm exposes user parameters, both of which can be avoided by the system designer in order to provide a non-expert user-experience). After the termination of the algorithm, the component displays all patterns found in a scrollable list below the mine button. This list is referred to as the **candidate area** because patterns *can* be taken over from here into the user's result set should she consider them suitable. For the purpose of defining study designs, the accessible state of the pattern mining launcher includes, in addition to the content of the candidate area, also the algorithmic settings that have been used by the user to find the patterns in it.

Finally, the **knowledge discovery result container** allows users to incrementally assemble result sets for their data analysis task. Results are received by means of drag-and-drop from the candidate area of the pattern mining launcher. They also can be deleted, re-arranged dynamically, e.g., in order to express a (linear) result prioritization, and annotated in order to express a result interpretation. The accessible state of this component is the ordered set of results along with all annotations as well as the time and number of mining rounds that have passed until each pattern was found.

In summary, the components of analytics dashboards are all designed towards simplicity and a strictly defined user-purpose. This serves two goals: the resulting data analysis systems are relatively accessible also to inexperienced users and the components allow for an unambiguous interpretation of user interaction—both of which are useful properties for test systems in user studies.

4. STUDY DESIGN COMPONENTS

After having an overview over the possible user-experiences that can be provided by the Creedo web application, we are now ready to turn to the more abstract aspects of study definitions. As discussed earlier, the central notion of Creedo's study domain language is the study design. In this section we now present in more detail the individual components of those designs: system and task specifications, performance metrics, and assignment schemes.

4.1 System and Task Specifications

A **system specification** within a study design is a specification of how to construct an analysis system given certain input parameters. When using the Creedo web application, this means how to construct an analytics dashboard from the components presented in Sec. 3. More generally, system specifications have the role of determining all aspects of the user-experience that are relevant to the study hypothesis. Typically, this means that different specifications within the same study design are identical except for a specific test component, the effect of which is to be determined by an

extrinsic evaluation. For example, the test system might contain a specific data visualization (e.g., PCA-based point cloud) whereas the control system does not, or the test system replaces specific algorithm parameters (e.g., support/lift in association mining) by an automatic choice whereas the control system exposes those parameters to the user.

The input parameters of the system specifications are then used during study execution to inject task specific content provided by the task specifications into the analysis systems. In the Creedo web application, the current default implementation of a system specification accepts a collection of **data loaders** that determine the data artifacts available in the dashboard and a collection of **component builders** that shape its visual appearance.

While system specifications define tools, **task specifications** describe the purpose and evaluation context, in which these tools are supposed to be used within a study. Looking back to our hypothesis blueprint (1), we are interested in assessing how well human users can solve specific tasks with certain analysis systems. That is, we are interested in how well humans can operate a software-tool with the purpose of producing results (a solution) for some task.

Naturally, this purpose has to be communicated to the user in a human-interpretable form. At the same time, the data analysis system expects a machine-readable input. Then, the evaluation of the hypothesis requires actual solution entities, for which certain qualities can be assessed. Hence, the state of the tool, in which the participant considers the task as solved, has to be interpreted as one or more of these result entities. Finally, also the human evaluators, if they are involved, need to have an understanding of the task in order to carry out their evaluations. Thus, another set of human-understandable instructions, yet from a different perspective, is required in studies which involve evaluators. In summary, task specifications in Creedo essentially have to be able to translate an abstract concept of a task between several ontological realms and between several perspectives. For the Creedo web application they consist of

1. human-understandable **instructions**, i.e., a sequence of HTML-documents that may refer to embedded media files,

2. **input values** of parameters accepted by all of the system specifications in the study design (e.g., the input data for which the task has to be carried out and other auxiliary input that is task specific),

3. a **result definition**, i.e., a rule that describes how to extract result entities from a state of the dashboard, in which the participant has declared that she is done working (this also includes determining whether a given state of the dashboard can be declared as "done" in the first place), and

4. a set of **evaluation schemes** that are used by human evaluators to evaluate task results.

At the moment, all evaluation schemes in Creedo are **rating schemes** that consist of a name, a set of natural language evaluation instructions, and a finite set $V \subseteq \mathbb{Z}$ referred to as the **scale** of the rating scheme. For a result r, a single evaluation according to such a rating scheme is then simply the value $v \in V$ that is assigned to r by one evaluator.

4.2 System Performance Metrics

Creedo formalizes the concept of one analysis system being "better" than another by comparing them through a system performance metric. These metrics are supposed to capture intuitive concepts like the "average quality of results produced (with a specific system)" or "average time needed by participants to produce a good result". The formal definition of such metrics for systems can involve other metrics defined on the different types of artifacts that are produced during study execution. That is, system metrics can rely on session metrics, session metrics on results metrics, and result metrics on evaluation metrics. On each of these levels, Creedo allows the definition of metrics according to certain production rules, but also provides a set of useful predefined metrics. Before we introduce the general language for metric definitions, let us consider some of the predefined metrics in order to get a sense for their general flavor.

Let c be an evaluation scheme given in the task specification of the study, and E_x denote the set of all c-ratings that have been performed for a result $x \in X$. A basic result metric is then the **average c-value** defined by $\hat{c}(x) = \operatorname{avg}\{c(e)\colon e \in E_x\}$. A slightly more complex variant of this metric abounds from applying a z-score transformation to all rating values assigned to results by a specific evaluator. This gives rise to the **average evaluator-normalized c-value** defined by $\hat{c}_*(x) = \operatorname{avg}_{e \in E_x}(c(e) - \mu_{u(e)})/\sigma_{u(e)}$ where $u(e)$ denotes the evaluator that has provided evaluation e and μ_u and σ_u denote the sample mean an standard deviation of all c-ratings provided by a particular evaluator u. This metric can be useful when the study designer suspects that some evaluators might interpret the rating scale of c differently than others. As an example for a full system performance metric consider the **average maximal f-value** that is defined for any result metric $f\colon X \to \mathbb{R}$ by

$$a \mapsto \operatorname{avg}\{\max_{x \in X_s} f(x)\colon s \in S_a, |X_s| > 0\}$$

where S_a denotes the set of all sessions that have been performed using system a, and X_s the set of all results that have been produced in session s. Another example is the **median time until success** that is again defined with respect to some result metric f and a success threshold $\tau \in R$ by

$$a \mapsto \operatorname{med}\{\min\{t(x)\colon x \in X_s, \hat{c}(x) > \tau\}\colon s \in S_a, |R_s| > 0\} \tag{2}$$

where $t(x)$ denotes the total session time until the result x was stored by the participant.

Creedo's general **language for metric definitions** is based on a set of elementary metrics (which can vary depending on context and implementation) and a set of production rules from which more advanced metrics can be defined. Let us first review some **elementary metrics**, which are defined in most contexts. For rating evaluations, elementary metrics contain:

- `Value`, i.e., the rating value chosen by the evaluator,
- `EvaluationTime`, i.e., the time taken by the evaluator to choose value.

For results, some elementary metrics are:

- `SessionTimeUntilStored`, i.e., the total time since the trial session started at the moment a result is added to the result area.

- `RoundsUntilFound`, i.e., how often the participant had to start a mining algorithm until the result was shown in the candidate area.
- `TimeBetweenFoundAndStored`, i.e., how long it took the participant to find and store the result after it has been produced by a mining algorithm.

The set of elementary session metrics includes:

- `TotalSessionTime`, i.e., the total time between the moment the participant first sees the analysis system until the system is closed,
- `RoundsInSession`, i.e., the total number of times the participant has pressed the mine button within the session,
- `NumberOfResults`, i.e., the number of results that have been submitted by the participant at the end of the session.

Now we can turn to the **production rules** for building more complex X-metrics, where X can be one of the sets of analysis systems, analysis sessions, results, or rating evaluations, i.e., $X \in \{A, S, R\} \cup \{E_c : c \in C\}$ (the rules also involves X-**constraints**, which are boolean function $q\colon X \to \{\text{true}, \text{false}\}$):

- If f and g are X-metrics then $x \mapsto f(x) + g(x)$, $x \mapsto f(x) - g(x)$, $x \mapsto f(x)g(x)$, $x \mapsto f(x)/g(x)$, and $x \mapsto f(x)^{g(x)}$ are also X-metrics.
- The trivial constraint $x \mapsto \text{true}$ is an X-constraint.
- If f is an X-metric then for all thresholds $\tau \in \mathbb{R}$ the functions $x \mapsto \delta(f(x) > \tau)$, $x \mapsto \delta(f(x) = \tau)$, and $x \mapsto \delta(f(x) < \tau)$ are X-constraints where δ evaluates to true if the condition given in brackets is true and false otherwise.
- If q and r are X-constraints then $x \mapsto q(x) \wedge r(x)$ and $x \mapsto \neg q(x)$ are also X-constraints.
- If g is a Y-metric, q is a Y-constraint, and $X \subseteq 2^Y$ then

$$x \mapsto \text{aggr}\{g(y)\colon y \in x, q(y)\}$$

  is an X-metric for all aggregation functions

$$\text{aggr} \in \{\max, \min, \text{avg}, \text{med}, \text{mode}, \text{count}\} \ .$$

Here, as a slight abuse of notation, we identify analysis systems with the set of analysis session that have been performed with them, i.e., $A \subseteq 2^S$, analysis sessions with the set of their results, $S \subseteq 2^X$, and result with the set of rating evaluations that they have received $X \subseteq 2^{E^c}$ for all rating schemes c in the task specification.

4.3 Assignment Schemes

The system performance metrics of a study design define a lot of measurements that can be taken based on the actions of study users (participants and evaluators). What these metrics do not determine is how an actual pool of study users should be assigned to perform these actions. This is the role of **assignment schemes**. In particular, they have to solve the following problems:

- *User workload has to be limited.* Each user has a limited amount of time and attention. If the study is demanding more of these resources than the user is willing to provide, she will not respond, and the user is essentially lost for the study.
- *Biases have to be controlled.* For instance, a participant working on the same task with two different systems, is likely to improve the second time independently of the system due to an increased understanding of the task itself. Another important source of biases abounds if users act in the double role of participants and evaluators: the study designer has to take into account how the knowledge that a participant gains when working on a task will effect her judgment when she is assigned to evaluate results of that task afterwards.
- *Dependencies of assignments have be resolved.* Trivially, evaluation tasks can only be assigned when the corresponding results have been produced in trials. On top of that, if a uniform distribution of evaluations is required for all results, then it makes sense to hold back evaluation assignments even further until the total number of results is known.

In summary, assignment schemes shape the behavior of a study at execution time by answering the following questions: Which participant is supposed to work with which system on which task, which evaluator is supposed to evaluate which result, and when, i.e., in what sequence, are these actions to be performed?

Currently, two schemes are available: `TwoPhaseAssignment`, which first issues all trial assignments to all participants and then, when those are all completed, issues all evaluation assignments, and `CombinedAssignment`, which generates a combined trial/evaluation assignment for users that have both roles. Both schemes have the following parameters in common:

- Number of trial assignments per participant.
- Constraints for valid trial assignments which can be a subset from
  - `IdenticalSystems`, i.e., all systems in all trial assignments for a given participant have to be identical,
  - `IdenticalTasks`, i.e., all tasks in all trial assignments for a given participant have to be identical,
  - `NoIdenticalTasks`, i.e., in no two tasks assignments of a given participant she is allowed to work on the same task,
  - `NoIdenticalSystems`, i.e., in no two tasks assignments of a given participant she is allowed to work with the same systems.
- Number of results per evaluator.
- Constraints for valid evaluation assignments which can be a subset from
  - `NoEvaluationOfOwnResults`, i.e., no result given for evaluation has been produced by the evaluator herself in one of her own trial assignments,

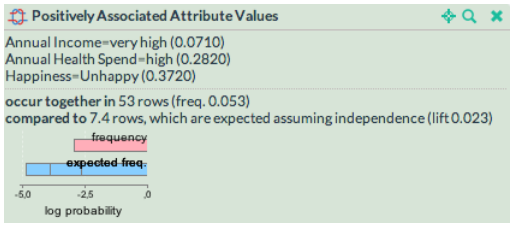


Figure 4: Association pattern stating that the joint frequency of three attribute/value combinations is "higher then expected". Intuitively, this statement is less interesting for someone who already knows that "annual health spend" is correlated with "annual income". FORSIED aims to quantify this effect.

- `NoEvaluationOfOwnTasks`, i.e., no result given for evaluation is of the same task as a task that the evaluator worked on herself,
- `ResultsStratifiedBySystems`. i.e., the results within an evaluation assignment have to come from all different systems at approximately equal amounts.

Both schemes assure that all system/task combinations will receive an approximately equal number of sessions and all results will receive an approximately equal number of evaluations.

5. USE CASE: EVALUATING FORSIED

In order to illustrate the various concepts of the Creedo framework that were introduced in the previous sections, we now turn to an exemplary case study. Namely, we present the design of a study that we conducted to evaluate the framework for "Formalizing Subjective Interestingness in Exploratory Data Mining" (FORSIED) [De Bie, 2013, De Bie and Spyropoulou, 2013]—a recently proposed pattern mining technique with a particularly user-centric approach. Note that a detailed discussion of this study and its results is out of scope of this paper, and will be published separately. Here we focus on the mapping of a real study hypothesis with theory-specific requirements to Creedo components.

5.1 Background and Hypothesis

FORSIED aims to quantify the interestingness of a pattern for a particular user depending on the prior knowledge that this user already has about the dataset, in which the pattern was found. Prior knowledge can come, e.g., in the form of certain summary statistics of the data, like row and column averages, or by other patterns that have been previously discovered. Importantly, FORSIED is an attempt to capture a universal notion of interestingness that in an ideal implementation (where all prior knowledge of a user can be assessed) coincides with the intuitive natural-language notion of interestingness. In order to make this very general concept more tangible, we focus here on an embodiment where we use a FORSIED-based pattern ranker as post-processing step in a round-based discovery of association patterns (see Webb [2010] and Fig. 4). This process works as follows: every round starts with the production of a random set of association patterns that, subsequently, is ordered by a ranker, before it is presented to the user. Then the user can pick from that ordered list patterns according to her interest and start over into the next round until she is done.

The FORSIED-based ranker, in particular, orders patterns according to their subjective interestingness considering as prior knowledge all the patterns that have already been discovered and stored by the user in previous rounds as well as the univariate distributions of all data attributes. Based on the design claim of FORSIED, in every round, the FORSIED-based ranker should point the user directly to those new patterns that are still interesting to her, and consequently allow her to save time and attention while browsing the result set compared to when a traditional static measure is used for ranking. The longer the discovery process proceeds the more this advantage should be emphasized. This gives rise to the following **hypothesis**:

> *"A FORSIED-based association discovery process allows users to discover a set of interesting patterns from a dataset faster than a conventional association discovery process (based on a static interestingness measure that is oblivious to prior and gained knowledge)."*

Translating this hypothesis into a useful operational study design implies defining a sufficiently robust objective measurement about subjective interestingness. In the next section, we will see how this apparent conundrum can be attacked by using Creedo components in order to control the knowledge of participants and evaluators throughout study execution.

5.2 Design Requirements

In addition to developing an executable study design that captures as precisely as possible our hypothesis, we also aim for a design that meets the following general requirements:

- It should evaluate the claim about "user interestingness" *extrinsically*, i.e., this notion should not just be captured by simplifying formal assumptions. In particular, this means that we want to employ human evaluators for assessing whether a result is interesting or not, instead of relying, e.g., on the intra-theoretic assumption that FORSIED's formalization of interestingness does indeed correspond to the intuitive notion thereof.
- Moreover, the design should be *robust*, i.e., the study result should be affected as little as possible by the potentially outlying behavior of individual participants or evaluators. This means that we want to average over different trials and result evaluations.
- Finally, we aim for a design that leads to *scalable* studies, i.e., the amount of time and attention required by the study owner should not depend on the number of participants. This means that we want to assign to users the double role of trial participant and result evaluator.

These three requirements have two practical implications for our study design. First, since we want to be able to meaningfully average over results, it must be possible for all task results to evaluate them on an identical scale. That

System	$\tau = 1$	$\tau = 2$	$\tau = 3$
a_{test}	**563.63**	**563.63**	**505.5**
a_{control}	694.25	803.96	inf

Table 1: Median time in seconds until success corresponding to the two systems and different success thresholds.

is, interestingness of a result should mean the same for all participants and evaluators. Thus, our task definitions (in particular instructions and dataset) have to control the prior knowledge among all users, because, as per our hypothesis, prior knowledge is what determines interestingness. Since we also want to put users into the double role of participants and evaluators, this creates the further difficulty that when a user is asked to evaluate a result, she must have the same prior knowledge at that moment as the participant when she created the result. Hence, we have to define two task variants such that performing one task as a participant does not change the prior knowledge for the other variant (and hence still allows an unbiased evaluation of it).

5.3 Study Design

The **system variants** for the study design are already more or less defined by the considerations in Section 5.1. In particular we have a_{test} corresponding to an analytics dashboard equipped with the FORSIED-based ranker and a_{control} corresponding to one with a conventional ranker based on the lift measure. As actual association discovery algorithm we fix a controlled pattern sampling algorithm [Boley et al., 2012] where all parameters (such as number of patterns per round, pattern pruning strategy) are fixed to feasible values. This makes the study design accessible also for participants that are not pattern discovery experts.

As stated above, we need two **task variants** in order to meet our requirements. In order to control the prior knowledge, the **input datasets** consist of randomly generated population data of two fictitious lands called "Lakeland" and "The Plain". Each dataset contains 1000 rows, each of which corresponding to a fictitious inhabitant that is sampled according to a joint probability distribution of the following variables:

- **Race** $\in$ {Griffin, Diricawl, Werewolf},
- **Region** $\in$ {east, west, north, east} (of residency),
- **Income** $\in \{0, 1, 2, \ldots, 1000\}$ (in gold coins),
- **Health spending** $\in [0, 1]$ (as fraction of annual income), and
- **Happiness** $\in$ {happy, unhappy}.

The probability distributions for each of the two lands have different multi-variate dependencies and model parameters in order to "inject" certain patterns into the datasets such as, e.g., $P[\textbf{Race} = \text{Diricawl}, \textbf{Region} = \text{north}]$ being much larger than the product $P[\textbf{Race} = \text{Diricawl}]P[\textbf{Region} = \text{north}]$. In addition to the input dataset the input values also contain the univariate statistics of all attributes for the FORSIED-based ranker. Corresponding to the two input data generators, the two variants of **task instructions** are:

> "You see a sample of the population of **(Lakeland/The Plain)**. Get familiar with the summary statistics of each attribute by [...]. Then click the mine button in order to find patterns in the population. You can save patterns you consider interesting by [...] and delete patterns by [...]. Repeat these steps to refine your pattern collection until you think you have discovered the most interesting collection consisting of exactly 3 patterns."

The **result definition** is then the collection of patterns present at the result container of the analytics dashboard considered as *a single result set* (and a result can only be submitted if there are exactly three patterns in this area). This corresponds to our hypothesis, in which we conjecture that the FORSIED-based system should have an advantage when considering the time it takes to construct a whole set of result patterns. In contrast, since the FORSIED-based system and the conventional system behave essentially identical before the first pattern is stored, we would not expect to see a substantial advantage on the single pattern level. Finally, this also has to be reflected in the **evaluation scheme** used for result evaluation. Here, we use a single elementary rating metric called "joint interestingness" with the instructions:

> "How interesting is this set of patterns as a whole (taking into account the elementary statistics of the data attributes)?"

and scale {0 (not), 1 (almost not), 2 (a little), 3 (somewhat), 4 (very)}.

Corresponding to our initial considerations, we choose `TwoPhaseAssignment` as **assignment scheme** with the parameter values of 1 for the number of trial assignments per participant, 3 for number of results per evaluator, and `NoEvaluationOfOwnTask` as constraint for the evaluation assignments. Using the constraint assures that evaluators will see results only from the task variant they did not work on themselves. Thus, as required, they will have the same prior knowledge when evaluating the task as the participant had when producing it.

As **system performance metrics** we can then define the median time until success (see Eq. (2)) for different thresholds, e.g., $\tau \in \{1, 2, 3\}$ corresponding to different requirements to the average ranking of a result to be considered a success. As a relatively strict criterion, we can then say that our hypothesis is supported by a study, if for all $\tau \in \{1, 2, 3\}$, we have $f_\tau(a_{\text{test}}) < f_\tau(a_{\text{control}})$ where f_τ denotes the system performance metric with respect to τ.

5.4 Results and Experiences

The authors conducted a study based on the above design with 16 participants/evaluators who followed an open invitation among our department members and master students. While this population was recruited from a friendly environment, note that based on the study setup, there was no reliable way for participants to tell with what system variant they were working or what system was used to produce the result they were evaluating. Table 1 contains the aggregated results of this study for the different strictness levels of the system performance metric. As we can see, it was indeed the case that for all three levels the median time until success was smaller for the test system than for the target system.

Using the Creedo web application, the whole study process could be administrated conveniently from one location, while participants and evaluators where able to fulfill their assignments from wherever they wanted within a previously fixed six day period (the first half of which was reserved for trial participation, the second half for result evaluation). Moreover, the study design is readily available for re-use with a different population of users—either unmodified in order to increase the confidence in our initial result or in a modified form in order to address specific doubts or to investigate refined follow-up questions.

6. SUMMARY AND OUTLOOK

As we have seen, Creedo can support researchers in defining and conducting repeatable studies with real users in order to extrinsically evaluate novel contributions in data mining. The focus of this support is at the moment on the easy definition of analysis systems, tasks, measurements, and assignment logic in order to control biases and to reduce the cost of the actual study organization and performance measurement. In particular, Creedo allows to employ participants also in the role of result evaluators in order to provide scalable study designs.

The most important limitations of Creedo's current state are perhaps as follows. While a relatively rich set of performance metrics can be expressed in the system, Creedo currently does not provide any support for the statistical interpretation of those metrics. That is, in case the study authors do know that their participants are a representative subset of a certain target demographic, there is no support for testing whether metric measurements are significant for that demographic. Moreover, reflecting current restrictions of the realKD library, the data sources that can be injected into analytics dashboards of the web application are limited to rather small-scale tabular data. Finally, the available visual mining components are somewhat specific to pattern discovery. However, as our survey among ECMLPKDD authors revealed, there appears to be a high demand for user studies also in other subfields of data mining, especially of course in visual analytics. Hence, extending the components available towards these areas is another major direction of growth.

Among the many directions for potential improvement, the authors believe that the extension of Creedo and its implementation should be mainly driven by the requirements emerging from actual practical attempts of extrinsically evaluating novel data mining techniques. If there is a developing culture in data mining research of performing studies with real users, then the community as a whole will understand better than today, what are central requirements for supporting this task. The organization of the Creedo web application as an open source project is a suitable basis for an organic growth that follows such a development. The authors are excited to engage in collaborations with other data mining research groups to jointly gain more experience in performing user studies and foster their growth.

Acknowledgment

This work was supported by the German Science Foundation (DFG) under the reference number 'GA 1615/2-1'.

References

Creedo web application. `https://bitbucket.org/realKD/creedo/`.

Kwiksurveys. URL `https://kwiksurveys.com/`.

realKD Java library. `https://bitbucket.org/realKD/realkd/`.

Surveymonkey. `https://www.surveymonkey.com`.

M. Boley, S. Moens, and T. Gärtner. Linear space direct pattern sampling using coupling from the past. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 69–77, 2012.

M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. Wrobel. One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD Workshop IDEA*, pages 27–35. ACM, 2013.

M. Boley, B. Kang, and T. Gaertner. Poll among ECML-PKDD authors on user studies. `http://www.realkd.org/dm-userstudies/ecmlpkdd-authorpoll-march2015/`, 2015. [Online; accessed 2-April-2015].

T. De Bie. Subjective interestingness in exploratory data mining. In *Advances in Intelligent Data Analysis XII - 12th International Symposium, IDA 2013, London, UK, October 17-19, 2013. Proceedings*, pages 19–31, 2013.

T. De Bie and E. Spyropoulou. A theoretical framework for exploratory data mining: Recent insights and challenges ahead. In *Machine Learning and Knowledge Discovery in Databases*, volume 8190 of *LNCS*, pages 612–616. Springer Berlin Heidelberg, 2013.

V. Dzyuba, M. v. Leeuwen, S. Nijssen, and L. De Raedt. Interactive learning of pattern rankings. *International Journal on Artificial Intelligence Tools*, 23(06), 2014.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3): 37, 1996.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

D. J. Hand. Pattern detection and discovery. In *ESF Exploratory Workshop on Pattern Detection and Discovery*, pages 1–12. Springer, 2002.

W. Ke, C. R. Sugimoto, and J. Mostafa. Dynamicity vs. effectiveness: Studying online clustering for scatter/gather. In J. Allan and J. Aslam, editors, *Proc. of the 32nd int. ACM SIGIR conference on Research and development in information retrieval*, pages 19–26. ACM, New York, 2009.

Y. Li, L. Chiticariu, H. Yang, F. R. Reiss, and A. Carreno-fuentes. Wizie: A best practices guided development environment for information extraction. In *Proc. of 50th Ann. Meeting of the Ass. for Comp. Linguistics*, pages 109–114. ACM, 2012.

I. Mierswa. Rapid miner. *KI*, 23(2):62–63, 2009.

D. Shahaf and C. Guestrin. Connecting the dots between news articles. In *Proc. of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632. ACM, 2010.

G. I. Webb. Self-sufficient itemsets: An approach to screening potentially interesting associations between items. *TKDD*, 4 (1), 2010.

D. Xin, X. Shen, Q. Mei, and J. Han. Discovering interesting patterns through user's interactive feedback. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 773–778. ACM, 2006.

ISPARK: Interactive Visual Analytics for Fire Incidents and Station Placement

Subhajit Das, Andrea McCarter, Joe Minieri, Nandita Damaraju, Sriram Padmanabhan, Duen Horng (Polo) Chau
Georgia Tech
Atlanta, GA, USA
{das, andream, jminieri, nandita, sriramp, polo}@gatech.edu

ABSTRACT

In support of helping to reduce the response time of firefighters, and thus deaths, injuries, and property loss due to fires, we introduce ISPARK. The ISPARK system determines where fire stations should be located, analyzes the primary causes of fires, the existing infrastructure, and response times, by using visualizations which show the GIS mapping of fire stations on a dashboard. Incidents and response times are shown as additional layers, with clustering of fire incidents to determine predicted fire station locations, forecasting of fire incidents using regression, causal, infrastructure, and personnel analysis, creating an interactive, multi-faceted method for locating fire stations. A comparison of urban and rural fire incident response times is another dimension of this study. We demonstrate ISPARK's usage and benefits using a publicly available dataset describing 300,000 fire incidents in the states of Massachusetts and Maine. ISPARK is generalizable to other geographic areas and domains, such as police stations, schools, hospitals.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human factors; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

General Terms

Visual Analytics, Data Mining, Human-computer Interaction, Design, Human Factors

Keywords

Fire incidents, GIS, clustering, regression, response time, mapping, NFIRS, FEMA, GeoJSON, leaflet, D3

1. INTRODUCTION

In 2013 [14], deaths, injuries, and property losses due to fire were extensive (Figure 2). If the response time can be reduced by just one minute, fewer injuries and deaths should occur, and the cost of reconstruction will be reduced. The goal is to reduce response time for fire stations to aid in a fire, thus reducing injuries, deaths, and property damage from fires. Determining where fire stations should be located to minimize driving distance and response time (Figure 1), analyzing the causes of fires, the existing infrastructure and personnel, and comparison of response times will be beneficial in reaching this goal. There are no federal laws on fire incident response time, but the National Fire Protection Association (NFPA) has detailed standards which most communities use [3]. Response time includes: (1) Dispatch time: 1 min.; (2) Turnout time: 1 min.; (3) Travel time: 4 min.; (4) Setup time: 2 minutes. Since fire grows exponentially in the first 10 min, doubling every second, before flashover, response time is critical.

Past approaches to the problem of fire station location have used GIS [8], optimization, classification, regression, and satellite imagery. GIS, primarily as an historical and descriptive tool, has been used in Oregon [10], Oklahoma City [22], Nevada [9], Moscow [16], and Turkey [18]. Baltimore [7] and the Open Data Institute [12] went beyond

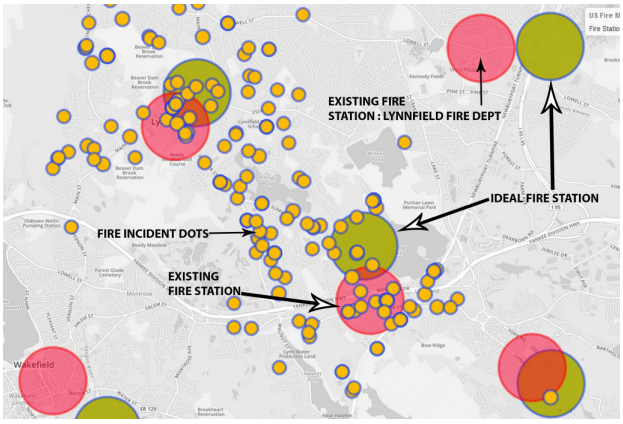


Figure 1: Screenshot of ISPARK showing actual (pink) and predicted (green) fire station locations in Maine determined by our approach, using coordinates with actual driving distances from fire stations to actual fire incidents. Fire incidents are shown as small yellow dots. ISPARK reduces the average driving distance between the fire stations and the fire incidents by about 1/3.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

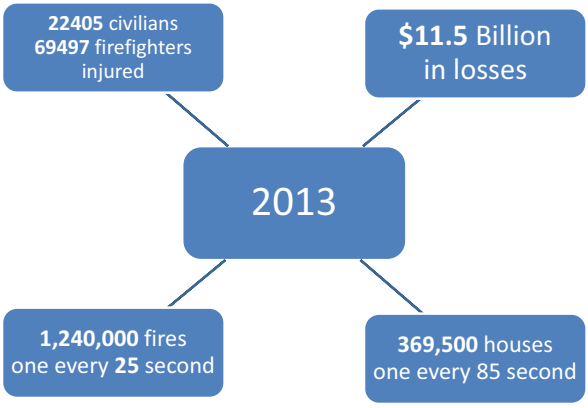


Figure 2: Impact of fire loss for the entire United States in 2013.

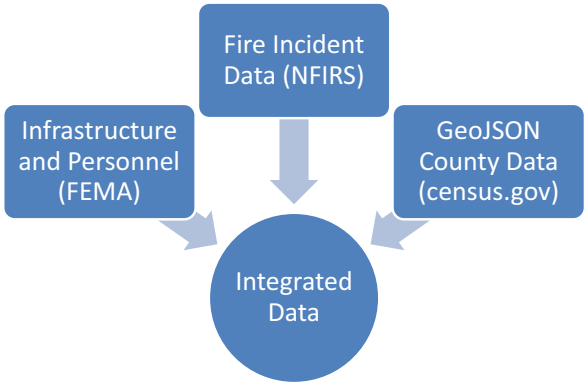


Figure 3: Schematic diagram showing the ISPARK data sources. County and state level detailed infrastructure and personnel data were available from FEMA, fire incident data from NFIRS, and geographic coordinates and population data from the U.S. Census.

these studies to create interactive maps, so the user could click on a fire station, close it down, and see the impact to response times. Malik et al created a system to visualize the impact of closing Coast Guard stations on search and rescue operations [1]. Interactive filtering and linked views were used by Maciejewski et al to visually detect hotspots [19]. Karafyllidis [13], Sen et al [24], and Liu [15] all used optimization combined with GIS to maximize map grid coverage and minimize cost. Sitanggang [20] studied physical data to classify 2693 objects using Naïve Bayes, relating them to fire incidence and location. Hernández-Leal et al [11] developed a fire risk index for forest fires using regression, combining variables such as satellite sensing data. In the city of Boston, Massachusetts (MA) an enterprise GIS system (ESRI based) is available to all staff, but has not yet been applied to locating fire stations, other than showing them on a map [6]. Maine (ME) also has an enterprise GIS system (ESRI) which has not yet been applied to this area [21]. In regard to interactive word clouds, previous work in this area has been done by Viegas [23].

Maine, with a population density of only 43 people per square mile, versus Massachusetts, with a population density of 12,793 people per square mile, were selected [5] so that an urban vs rural comparison could be made. Both MA and ME contribute data to the National Fire Incident Reporting System (NFIRS), a publicly available voluntary database used in our project [17]. One of the members of our project team is a volunteer firefighter in Maine, so his domain knowledge in this area is a significant help to us.

The central theme of our visualization is a GIS-level view of the data, followed by visualizations such as geographic patterns, parallel coordinates for infrastructure, word clouds examining causes, recommended fire station location, response time comparisons in urban and rural areas, and predictions of response times in the future. Previous work lacked interactivity, used expensive tools, was not easily extensible to other areas, and used outdated methods. We introduce ISPARK, which provides the following contributions:

- An interactive, integrated dashboard using open source tools, for implementation at low cost for fire departments across the United States, using ME and MA as the starting points.
- Prediction of the recommended location of fire stations using K-means clustering.
- Prediction of the response times for future years, and comparison of the actual and predicted response times of the firefighters.
- Determination of differences in urban and rural response times.

2. DATA SOURCES AND PREPARATION

Three data sources were used: (1) FEMA, for the infrastructure and personnel; (2) NFIRS for the fire incident data; and (3) the U.S. Census for the GeoJSON county level coordinates and population data (Figure 3). The U.S. Fire Administration collects data via the National Fire Incident Reporting System (NFIRS) system, which is the world's largest national, annual database of fire incident information. NFIRS is a reporting standard that fire departments use to uniformly report on the full range of their activities, from fire to emergency medical services (EMS) to equipment involved in the response. The database comprises about 75 percent of all reported fires that occur annually. Participating fire departments report about 22,000,000 incidents and 1,000,000 fires each year. For this study, approximately 300,000 records and 50 fields were extracted and cleaned for Maine and Massachusetts for 2010 through 2012, which was 500 MB. Two formats were created: csv (comma-separated values) and JSON (JavaScript Object Notation). Most of the data analysis was performed using csv format files, except for shape files and GeoJSON files to overlay shape layers on top of the Leaflet Map.

An additional data source was Microsoft Bing, which was used for geocoding fire station, fire incidents, driving distance (distance from the fire station to the fire incident), and predicted driving duration (time to get from the fire station to the fire incident). The latter was done both for the existing and the recommended fire stations as the originating point. We chose Bing due to their high query threshold, allowing us to issue as many API calls per day as were needed for this project.

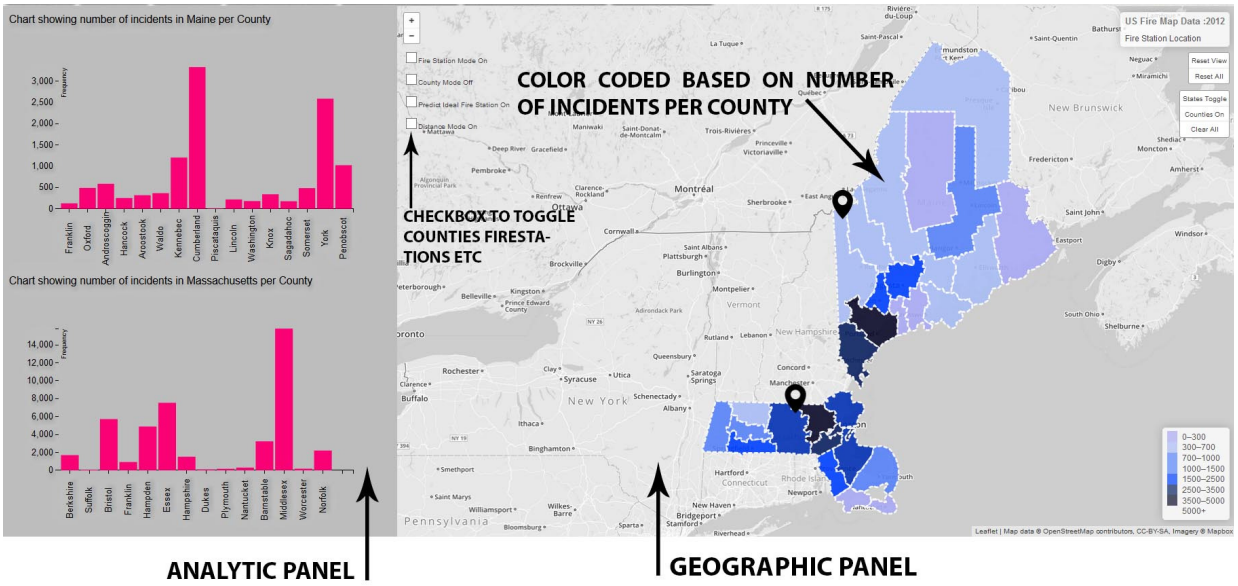


Figure 4: ISPARK's dashboard when opening the application. Left: fire incidents by county in Maine and Massachusetts for 2012 shown as histograms. Right: showing the same fire incident distributions geographically on a map (a darker county means having more fire incidents).

Perl scripts were written to refine the data for Maine and Massachusetts from the NFIRS national database for 2010, 2011, and 2012 in both JSON and csv formats. The latitudes, longitudes, predicted duration and driving distances from the fire station to the fire incident were also obtained from Bing using Perl scripts.

3. ISPARK: DESIGN & CAPABILITIES

3.1 Overview of The ISPARK System

Our approach includes integration of the data, visualization of the data on a dashboard, and various analyses of the data (Figure 4). The dashboard was divided into two sections, an analytics and a geographic panel (Figure 5). The analytics panel, based on D3.js, a JavaScript library for manipulating documents based on data [4], changes depending on the selected mode of the geographic panel. The geographic panel reveals the visual patterns of incidents, fire stations, and recommended fire stations using GIS mapping with Leaflet.js.[2] The analytics information include trends and statistical regressions, infrastructure analytics using D3.js parallel coordinates, K-means clustering to determine the recommended fire station locations, comparison of predicted with actual response times, and word clouds based on the causes of fires.

3.2 Dashboard Summary

The two dashboard panels are: (1) the left analytic panel, in which multiple histograms, word clouds, and other analytics are displayed, including the default 2012 county level data for Maine first, and then Massachusetts underneath; (2) the right geographic panel, which allows the user to zoom to any level desired using the "+" and "-" keys in the upper left section of the panel, with the default display shown as Maine and Massachusetts divided by county and shaded according

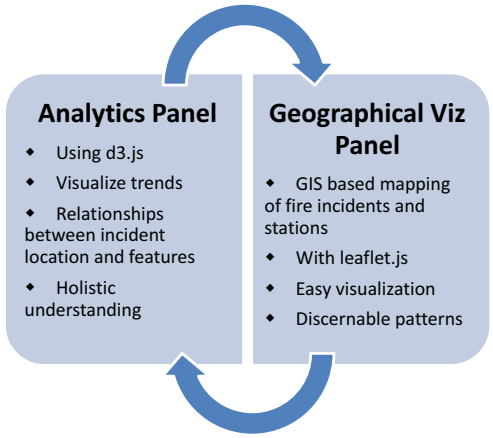


Figure 5: Schematic diagram showing the proposed data visualization toolset UI.

to the 2012 density of fire incidents. The right panel also offers multiple check boxes for "Fire Station Mode", "County Mode Off', "Predict Ideal Fire Station On", and "Distance Mode On". Additionally, prominent teardrop shape markers on each state can be clicked to show state level data. The user can also switch to another year through a menu option offering the years 2012, 2011, or 2010. These features will be described in more depth as we proceed through the design description.

The user will see the opening visualization with the map of the United States on the right, and the defaults shown as indicated (Figure 4). By clicking the buttons for each year, the county incident histograms on the left panel and density of shading in the counties in the right panel will change

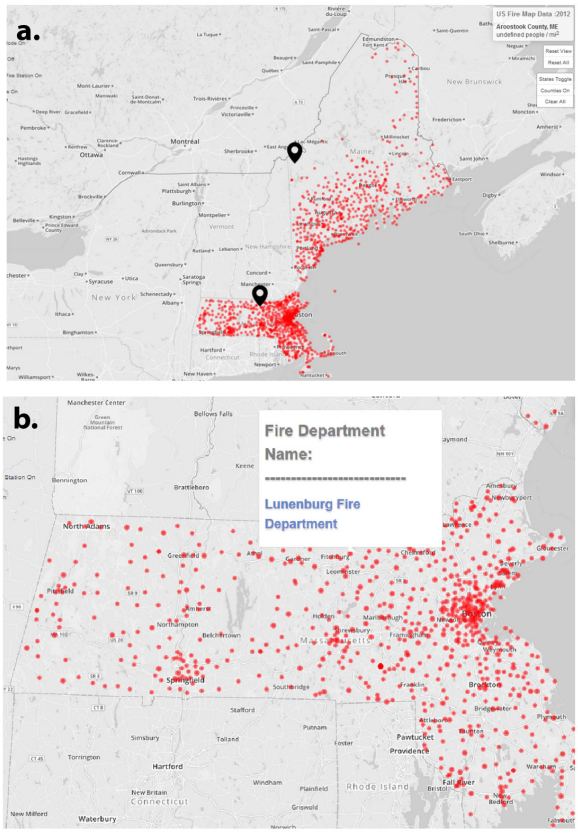


Figure 6: (a) Screenshot showing the fire stations (red dots) mapped over the states of MA and MN; (b) Fire stations zoomed in over the state of MA, and pop-up of fire department name, Lunenberg, when hovering over fire station.

appropriately. Since the fire incidents are too numerous to show individually, the counties are shaded according to the density of fire incidents, thus avoiding overplotting .

One of the options on the right panel offered to the user is to go into the "Fire Station Mode" (Figure 6a). Checking this box on the right panel will add the fire station location overlays on the states of MA and ME. By also checking the "County Mode Off" box, the fire station display will be cleaner looking. By clicking the "+" sign in the upper left of the screen, the user can zoom in as far as desired, and then, by hovering over a fire station node (red), see the name of any of the individual fire stations (Figure 6b). Note that the infrastructure, i.e., the number of various types of staff is shown (career, volunteer, paid per call, and so on) is also shown for that station.

3.3 Mapping Fire Incidents: Techniques & Design

Open source web technologies have been used, with a two panel interface, the right panel containing the Leaflet map with layers, and the left panel containing a JavaScript enabled interface, with multiple D3 visualizations. These two panels are designed to communicate with each other, so that

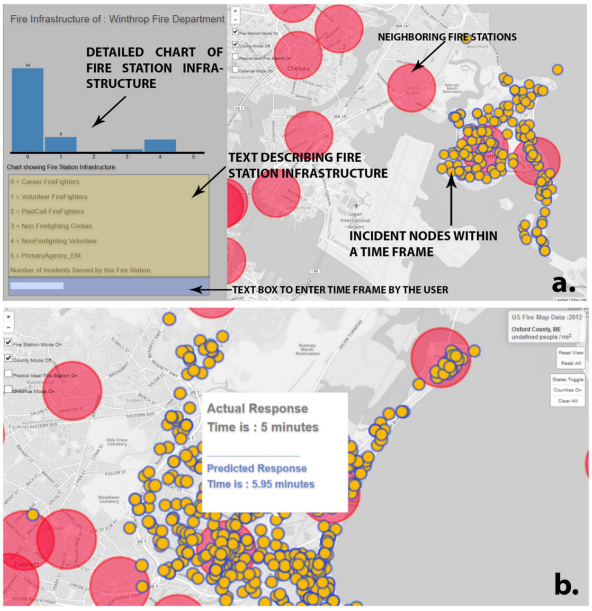


Figure 7: (a) Fire stations (red discs) zoomed in with left panel showing infrastructure for Winthrop fire station being hovered over; (b) Fire incidents (yellow dots) within a certain response time interval (e.g., 5 min) for the Winthrop fire station.

the related charts show up as the user selects various sets of information options on the visualization.

This map showcases incident data from 2010, 2011 and 2012, and provides a basis for future fire station locations for the entire country. The longitude and latitude values are used to retrieve the counties for each incident, and then shade them based on the number of incidents per year. The dashboard includes popups which load up as soon as the user clicks on any of the fire incidents or fire stations, providing more detailed information about the fire department, its capacity, and average response time. For the fire incidents, popups provide the actual response time compared to the predicted response time for that incident.

Multiple modes are provided, including the fire station and the incident modes. The fire station mode allows the user to zoom in to see the locations of all of the fire stations, and to see the number of different types of personnel (professional, volunteer, EMS staff) for each fire station. Selection buttons at the top by year provide the data for those time frames. Markers on top of each state provide various types of state level data when clicked.

3.4 Visualizing & Predicting Response Times

In Figure 7a, one of the fire stations has been selected by the user, and a response interval entered in the left panel, with the fire incidents served by that fire station within the specified time interval (5 min.) shown as yellow nodes. If the user now hovers over an incident, both the actual and the predicted response time will be shown (Figure 7b). Most often, the actual response time will be faster than the predicted response time, as in the example below, since the fire

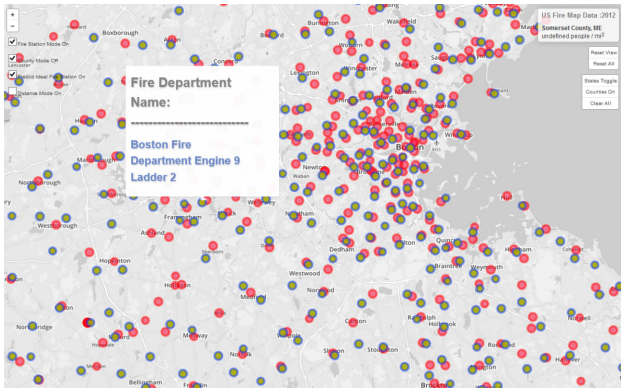


Figure 8: View showing predicted fire station dots in green color against existing fire station dots in red color.

truck can speed to the fire.

Statistical regressions using SAS were used to explore the relationships between variables. The fire location arrival times for each incident were subtracted from the time of the fire alarm. 3-5 percent of the records had either 0, nothing, or were negative, so they were removed from the analysis. Unrealistically long response times (over one hour) were also removed. Unrealistically long distances (over 30 miles) between the fire station and the incidents were also removed. The times and distances retained as realistic were selected based on the domain knowledge of the volunteer firefighter from Maine on our team.

K-means clustering, using Python, was performed on the fire incident data to obtain recommended fire station locations based on the coordinates of fire incident data. K-means clustering partitions n observations (fire incidents) into k clusters (fire stations), with each observation belonging to the cluster with the closest mean. This method provides the centroids of all the k clusters, i.e., predicted locations for fire stations.

Clicking on "Reset all" and checking the box "Predict Ideal Fire Station On" will show all recommended fire stations, as shown in (Figure 8). For ME, the recommended fire stations are shown in green, whereas in MA they are shown as blue. The driving distances for each of the recommended fire station locations to the incidents were calculated, and, on the average, the distance to the incidents was reduced by one-third compared to the actual fire station locations. These recommended locations would result in a significant reduction in response times. However, these locations are not located on existing streets, and expecting local governments to move their fire stations based on this data is unrealistic, which will be addressed in the discussion section.

3.5 Comparing and Predicting Firefighter Response Times across States and Years

By pressing the "Reset All" button on the right panel, the state level option menu is shown (Figure 9), including fire station response times, response times by year, actual versus log response times, driving distance comparisons, and the prediction of response times.

Clicking on the third menu item, "Actual vs Log Response Times" shows that the distribution of response times is a highly skewed distribution (Figure 9a). In order to use linear regression, all response times were converted to the log of response time, which was a more normal distribution. Clicking on the fourth menu item, the "Driving Distance Comparison" option shows that it is consistently about a mile further in ME (3 miles) compared to MA (2 miles) to get to the fire incidents, since the rural areas are more spread out (Figure 9b).

The average actual response times in both MA and ME are shorter than those predicted ($p < .0001$), with MA showing an actual response time about three min faster than the predicted time (Figure 9d). ME also showed faster actual than predicted times, but only by one minute. This is to be expected, given that the fire trucks can go faster than regular vehicles. For ME, increasing their speed is made more difficult by the poor road conditions and often older equipment.

Clicking on the fifth menu item, the "Prediction of Response Times" option shows that the annual predicted response time for both MA and ME and the actual response times for 2010, 2011, and 2012 show very little change from year to year, although there are significant differences in response times between the two states, with ME showing longer response times ($p < .0001$) by about two minutes every year. No changes in future years response times for either state are predicted.

An historical regression over a three year period showed significant ($p < .0001$) relationships between the log of the response time and the log of the driving distance to the fire incident, the state (with Massachusetts showing lower response times), and the month/year of the fire incident. The overall strength of the relationship, however, was very low (r squared=.071). Due to the small amount of change in response time by year, any reduction in response time will need to be from variables such as the relocation of the fire stations. Basically, the firefighters are getting to the fire incident fast enough, they just need to be located closer to the incidents.

3.6 Causality Analysis

Word clouds were created using D3 and JavaScript to visualize the causes of the fires. This is a high level view based on unigrams created from the fire cause data, with noise words removed. Each term is given a font size proportional to its frequency. This provides a quick, intuitive idea of the distribution of causes for incidents by county. Clicking on each word causes the incidents with that cause to be highlighted on the map.

By clicking "Reset All", and then clicking on an incident with a particular cause, in this example an "accident", highlighting of all incidents caused by that particular type of event in cyan will occur (Figure 10). Causes of fires can also be highlighted from the main maps for both states by clicking on one of the words on the left, such as accident, and then all of the incidents will appear. Zooming in, after selecting a cause from the word cloud, allows the user to see the actual nodes for each fire incident. If a particular county is then selected, then the distribution of fire causes for that county can be seen in the word cloud on the left.

3.7 Infrastructure and Personnel Analysis

The fire station infrastructure visualizations (Figure 11) were developed using a "Parallel Coordinates Graph" from

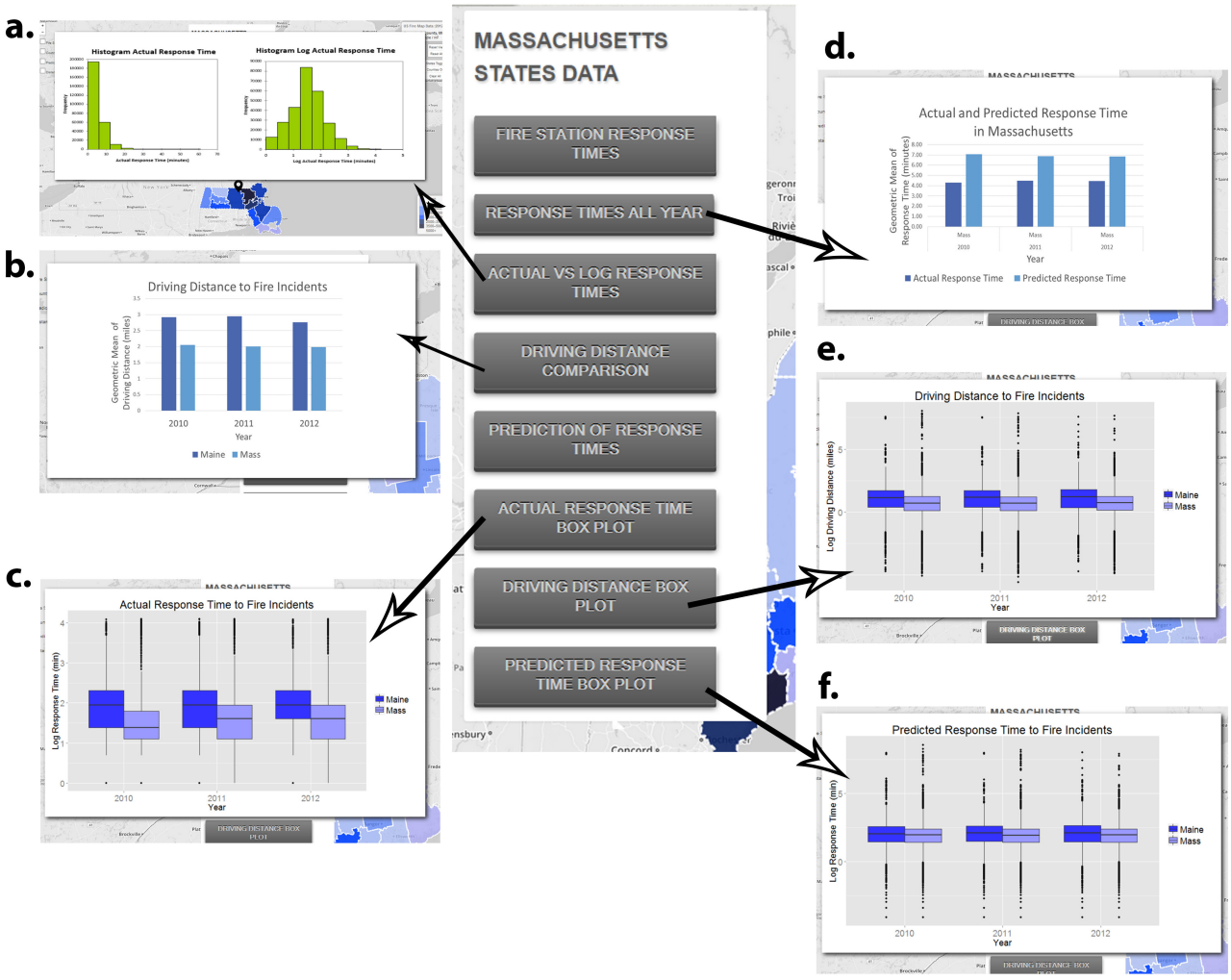


Figure 9: State level menu : (a) Linear and log distributions of response times. (b) Actual driving distance to fire incidents by year and state. (c) Actual Response Time to Fire Incidents Boxplot (d) Actual and predicted response times.(e) Driving distance to fire incidents (f) Predicted Response Time to Fire Incident

the D3 Library. Clicking the "Infrastructure" button at top of screen reveals data obtained from US Census Data provided though the Federal Emergency Management Agency (FEMA). For ME, *career firefighters* are the smallest group, followed by *volunteer* and *paid on call* firefighters. For MA, the biggest group is the career firefighters. Also we made the parallel coordinates graph interactive by enabling black boxes as brushes on top of each vertical axis. The user can drag them vertically and also can enlarge their sizes. Their role is to help the user to subset range of values to be displayed in bold colors, while other feature values would be ghosted on the back. This helps the user to selectively comprehend the flow and direction of the data.

4. DISCUSSION

Regarding the prediction of the recommended location of fire stations using K-means clustering, while the re-location of the fire stations based on this plan would reduce the distance to the fire stations to the fire incidents, such a plan is unrealistic. One improvement would be to treat the problem as a facility location analysis, with a set of potential realistic locations, but still using a clustering methodology. Another approach would be to provide interactive opening and closure of selected fire stations to observe the impact on the response times of the remaining fire stations. Since regression showed that the response times are not expected to increase in the future and the firefighters are either close to meeting or exceeding the required response times, this approach would make even more sense.

We observed that rural firefighters, on average, travel about one mile further to get to the fire incidents than those in an urban area, and their response times are about two minutes slower than in urban areas. Possible explanations for these findings include: (1) rural firefighters may need to travel

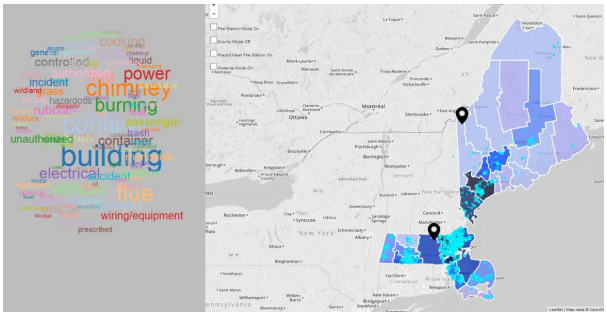


Figure 10: View showing all those fire incident dots in cyan related to a specific fire cause word ("accident") when clicked in the left panel.

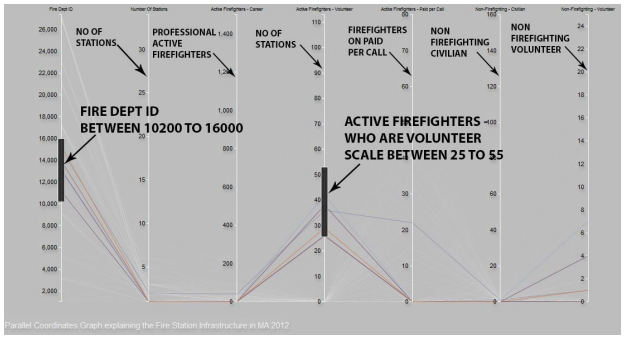


Figure 11: Infrastructure distribution for a county in Massachusetts in 2012.

on gravel and dirt roads, which slows them down; (2) firefighting equipment in rural areas is older, slower, and less specialized; (3) rural firefighters are more likely to be volunteers than career firefighters, so they may need to first travel to the fire station (e.g., from home) before going to the incident location; (4) rural fire incidents are more spread out geographically.

ISPARK and our approaches may easily work with data from any fire departments in the U.S. or other countries. However, the expertise to maintain the application will need to be available to the fire department. ISPARK may also work with data from other related domains, such as police, hospitals, and schools, so they could use similar technology.

It is noteworthy to mention that the word cloud we presented as one of the core features of visual analytics, shows a wide array of fire causes per county. Like wise to fit them in a restricted space, many of the words are overlapping. But to make it easier to be able to select each and any of those words by the user, we implement opacity changing mechanism, when the user's pointer hovers over them. This way the user in real time gets to know which word tag he is going to pick to see further analytics.

Our future work includes revising the methodology for determining recommended fire station locations, adding options for interactively closing and opening existing fire stations to observe the consequences on response times, extending ISPARK to the rest of the United States, and exploring more factors, including the spatiotemporal variables that would influence the fire station workload, such as time of day, day of the week, and month of the year. Additional studies should add traffic conditions to the predicted response time for increased accuracy. Also, wildland fires likely have unique characteristics which should be explored.

5. CONCLUSIONS

Our project has led us to the following four conclusions. First, an interactive, integrated analytical and geographic dashboard can be developed using entirely open source tools. Second, geographic mapping of the recommended fire stations, existing fire stations, and the fire incidents served by the fire stations can be done, but the recommended location methodology needs to be further refined. Third, the incident response times over the period 2010 through 2013 have been stable, and firefighters are either close to meeting or exceeding the standard response times now. Last, rural and urban areas do show differences in fire response times.

6. REFERENCES

[1] Malik A, Maciejewski R, Maule B, and Ebert D.S. 2011. A visual analytics process for maritime resource allocation and risk assessment. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on* (23-28). VAST, 221–230.

[2] Vladimir Agafonkin. 2011. Leaflet: An open-source JavaScript library for mobile-friendly interactive maps. (May 2011). `http://leafletjs.com/`

[3] National Fire Protection Association. 2015. National Fire Protection Association Website. `http://www.nfpa.org/`. (March 2015).

[4] Michael Bostock. 2011. A JavaScript visualization library for HTML and SVG. (February 2011). `http://d3js.org`

[5] US Census Bureau. 2015. Quick Facts. U.S. Census Bureau, U.S. Department of Commerce. `http://www.census.gov/quickfacts/table/PST045214/00`. (February 2015).

[6] Fire Department City of Boston Official Website. 2015. City of Boston, Official Website. `http://www.cityofboston.gov/fire/`. (February 2015).

[7] ESRI. 2011. Baltimore City (MD) Fire Department Uses GIS Technology to Optimize Resources. (September 2011).

[8] ESRI. 2015. GIS for Fire Station Locations and Response Protocol(2011). `http://www.esri.com/library/whitepapers/pdfs/gis-for-fire.pdf`. (March 2015).

[9] William Finley. 2007. Fire Station Location Master Plan. (January 2007). `http://www.usfa.fema.gov/pdf/efop/efo40119.pdf`

[10] David K. Hard. 2006. Analysis of Factors and Recognized Standards Utilized to Determine Fire Station Locations. *Klamath County Fire District No 1, Klamath Falls, Oregon* (June 2006).

[11] Pedro A Hernández-Leal, Alejandro González-Calvo, Manuel Arbelo, Africa Barreto, and Alfonso Alonso-Benito. 2008. Synergy of GIS and Remote Sensing Data in Forest Fire Danger Modeling. (December 2008).

[12] Open Data Institute. 2015. London Fire Stations. http://london-fire.labs.theodi.org. (March 2015).

[13] Ioannis Karafyllidis and Adonios Thanailakis. 1997. A model for predicting forest fire spreading using cellular automata, Vol. 99. Ecological Modelling, 87–97. Issue 1.

[14] Michael J. Karter Jr. 2014. Fire Loss in the United States during 2013. http://www.nfpa.org/newsandpublications/nfpa-journal/2014/september-october-2014/features/2013-fire-los. (September-October 2014).

[15] Nan Liu, Bo Huang, and Magesh Chandramouli. 2006. Optimal Siting of Fire Stations Using GIS and ANT Algorithm, Vol. 20. Journal of Computing in Civil Engineering, 361–369. Issue 5.

[16] Sergey Mescherin, Igor Kirillov, and Stanislav Klimenko. 2014. Optimizing and Visualizing Fire Dispatcher Activity. (October 2014).

[17] Federal Emergency Management Agency US Fire Administration National Fire Incident Reporting System, Department of Homeland Security. 2015. NFIRS 5.0 Reference Guide (2013). https://www.nfirs.fema.gov/documentation/reference/. (March 2015).

[18] Recep Nisanci, Volkan Yildirim, and Yasar Selcuk Erbas. 2007. *Fire Analysis and Production of Fire Risk Maps: The Trabzon Experience, Risk Management for the Future - Theory and Cases. Chapter 10.*

[19] Maciejewski R, Rudolph S, Hafen R, Abusalah A, Yakout M, Ouzzani M, Cleveland W S, Grannis S J, and Ebert D S. 2010. A Visual Analytics Approach to Understanding Spatiotemporal Hotspots. In *Visualization and Computer Graphics, IEEE Transactions on* (23-28). VAST, 205–220.

[20] Imas Sitanggang, Yaakob Sukaesih, Mustapha Razali, A N Norwati, and Ainuddin. 2012. Application of Classification Algorithms in Data Mining for Hotspots Occurrence Prediction in Riau Province Indonesia, Vol. 43. Journal of Theoretical and Applied Information Technology, 361–369.

[21] Fire Marshall State of Maine Official Website. 2015. State of Maine Official Website. http://maine.gov/dps/fmo/index.html. (February 2015).

[22] Division of System Planning Corporation TriData. 2006. Oklahoma City Fire Department Fire Station Location Study.

[23] Fernanda B Viégas, Martin Wattenberg, and Jonathan Feinberg. 2009. Participatory Visualization with Wordle. In *IEEE Transactions on Visualization and Computer Graphics* (23-28), Vol. 15. 1137–1144.

[24] A Şen, T Önden, C Gökgöza, and C Sen. 2011. A GIS Approach to Fire Station Location Selection. Inter al Soceity for Photogrammetry and Remote Sensing.

Interactive Clustering with a High-Performance ML Toolkit

Biye Jiang
Computer Science Division
UC Berkeley
Berkeley, CA 94720
bjiang@cs.berkeley.edu

John Canny
Computer Science Division
UC Berkeley
Berkeley, CA 94720
jfc@cs.berkeley.edu

ABSTRACT

Clustering is a class of machine learning algorithms which has important applications in many different fields. Users often use clustering to find hidden structures from data for those domain specific problems. However, evaluating clustering results is always a hard problem. In many and perhaps most of these applications, users need to trade off competing goals and encode prior knowledge into the model to define what is the best result. The learning algorithm however has evolved around the optimization of a single, usually narrowly-defined criterion, which may not obtain satisfactory results. In most cases, an expert makes trade-offs between different criteria which requires high-level (human) intelligence. This motivates us to provide *interactive customization and optimization* so that the expert can incorporate secondary criteria into the model-generation process in an interactive way.

In this demo paper we will demonstrate the techniques we developed to do customized and interactive model optimization for clustering algorithms. The keys to the approach are (i) high-performance training so that non-trivial models can be trained in real-time (using roofline design and GPU hardware), (ii) a machine learning architecture which is modular, and supports primary and secondary loss functions, and (iii) highly-interactive visualization tools that support dynamic creation of visualizations and controls to match the bespoke criteria being optimized.

Keywords

Interactive, Machine learning, Clustering, GPU

1. INTRODUCTION

Machine Learning is now at the center of data analysis in many fields across the sciences, business, health care and other realms. Among those ML algorithms, clustering is a class of unsupervised learning algorithms which is often used to find hidden structures of the data. In contrast to supervised learning which always uses accuracy to measure model quality, in general it is hard to evaluate clustering results. In practice users often need to trade off competing goals. Latent variable models such as K-Means clustering, or LDA[3] or NMF[18] find latent factors which maximize the likelihood of the observed data, but which may have secondary desiderata such as uniform cluster size, independence of factors, or coherence of topics.

The learning algorithms have evolved around the optimization of a single, usually narrowly-defined criterion, but users often find it hard to represent their criteria as a single objective function. The lack of evaluation methods also make it hard for analyzing algorithm behavior as well as debugging. In many cases, using clustering algorithms requires high-level (human) intelligence to make trade-offs between these criteria and examine the results manually.

Because ML models today are not flexible enough to incorporate all these criteria, secondary constraints are often applied *after* model training (by overriding the model's choices) in a way that is inevitably sub-optimal. Furthermore, the effects of downstream interventions require live testing to quantify. By contrast, *interactive customization and optimization* allows the analysts to incorporate secondary constraints into the model-generation process in an interactive way. There are several benefits to this:

- Models can be fully optimized given a suitable mixture of the criteria.
- Families of models can be trained to deal with variability in the application context.
- Analysts can explore the effects of particular trade-offs instantly, without waiting for a live test.
- Through this exploration, an analyst can gain intuition for the effects of various criteria, and make better trade-offs in the long run.

In this paper we develop the techniques to do customized and interactive model optimization, and demonstrate the approach on several examples. The keys to the approach are (i) high-performance training so that non-trivial models can be trained in real-time (using roofline design and GPU hardware), (ii) a machine learning architecture which is modular, and support primary and secondary loss functions, and (iii) highly-interactive visualization tools that support dynamic creation of visualizations and controls to match the bespoke criteria being optimized.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

2. RELATED WORK

2.1 Interactive clustering

Interactive clustering is an active area. Since clustering is widely-used to simplify the interpretation of large datasets, and since the natural metrics for a new domain may be difficult to articulate, interactive exploration [8, 24, 29] is a natural and powerful approach. In [8, 24] authors used visualization to rapidly explore the results of a clustering algorithm, and these approaches have become important tools in computational biology [8]. AverageExplorer [29] allow users to explore and summarize large collection of images by interactively posing constraints.

Recently, there has been much interest in using visualization to support the refinement of topic models [26, 12, 7]. Since the latent topics extracted by the algorithm are not always semantically meaningful [22, 6], different constrained topic models [20, 2, 21] have been developed. Systems like [26, 12] also allow users to iteratively refine the model based on their preference. However, those models always require solving a complicated optimization problem with some very specific constraint. And few systems have demonstrated real-time interaction with large scale dataset.

2.2 Interactive model refinement

In the context of supervised learning, one early influential paper on interactive machine learning was Fail and Olsen's paper [9], which describes partially-supervised learning with a user supplying some (sparse) labelled data to help an ML algorithm label the rest. A number of other works have followed this route, by focusing on manipulation of the training data rather than internals of a particular algorithm. Other work focused on human-assisted feature selection (rather than algorithm training) [23]. Amershi et al. [1] provides a detailed summary of the work in this area. Much of these works attempt to improve only the accuracy of a machine learning problem by adding a human in the loop, which is quite different from our work. Perhaps the closest to ours is [13] which integrates a human-assisted optimization strategy with the design of multi-class classifiers. But in this paper we focus on clustering and the optimization algorithms are different.

2.3 Large scale machine learning system

There has been a great deal of work recently on tools for Big Data. But much of these works emphasize scalability on clusters [14, 25, 10] without applying single-node acceleration (CPU and GPU-specific acceleration libraries). Those systems are typically optimized for scalability rather than latency, which is more important for interactive modeling.

3. SYSTEM DESIGN

3.1 BIDMach: high-performance, customized machine learning

The first key to interactive, customized machine learning is an architecture which supports it. BIDMach [5] is a new machine learning toolkit which has demonstrated extremely high performance with modest hardware (single computers with GPUs), and which has the modular design shown in Fig 1. BIDMach use minibatch updates, typically many per second, so that models are being updated continuously. This is a good match to interactive modeling, since the effects of analysts actions will be seen quickly. Rather than a single model class, models comprise first a primary model (which typically outputs the model loss on a minibatch and a derivative or other update for it). Next an optimizer is responsible for updating the model given gradients. Several are available including simple SGD, ADAGRAD, and Preconditioned CG. Finally, mixins represent secondary constraints or likelihoods. Gradient-based primary models and mixins are combined with a weighted sum. In our interactive context, these weights are set interactively.

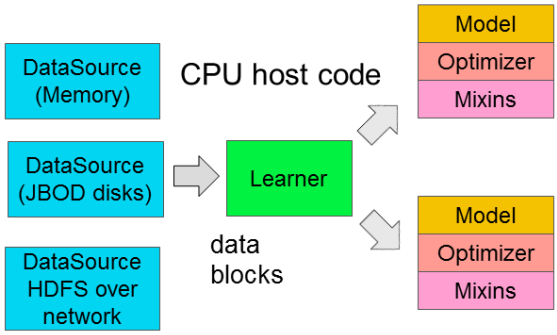


Figure 1: BIDMach's Architecture

Beyond the architecture of Fig 1, BIDMach has two layers. A general-purpose matrix layer called BIDMat, and BIDMach which includes the machine learning classes from figure 1. This organization shortens development time by providing high-level primitives for writing learning algorithms, and also allows us to leverage recent gains in the performance of GPU hardware. BIDMat is completely agnostic about matrix type: both CPU and GPU matrices (and sparse or dense and single or double precision) can be used and code is written generically.

BIDMach contrasts with most high-performance machine learning systems [14, 25, 10] in its emphasis on optimizing single-machine performance first, and then scale-up if needed. BIDMach uses *roofline design* to optimize computational kernels toward hardware limits. On a large collection of benchmarks it has proved to be typically two orders of magnitude faster than other single-machine toolkits (when BIDMach is used with a GPU), and one to two orders of magnitude faster than cluster toolkits running on 10-100 nodes. Part of the difference is due to BIDMach's complete suite of GPU primitives. Almost all computation is done on the GPU, CPU/GPU transfers are minimized, and custom kernels give close to theoretically optimal GPU performance. GPUs typically achieve an order-of-magnitude speedup in dense matrix operations vs. mid-range CPUs. Less well know is their advantage in main memory speed, which (at 300 GB/s) is nearly an order-of-magnitude faster than recent quad-channel CPUs (at around 40 GB/s). This memory speed gap also gives GPUs a similar advantage for *sparse* matrix operations which are central to most real-world ML applications. These differences explain one order of magnitude of the performance gap that we observe with BIDMach. The balance is due to the fact that most other systems are not close to their (CPU) rooflines. Because of this BIDMach

has a significant performance edge for most ML algorithms even when run on one CPU.

High performance is very important for interactivity. BIDMach has reduced the running time of many non-trivial ML tasks from hours to minutes. And even for models that take minutes to train fully, the effects of parameter changes are typically visible in seconds. We will see this in the examples later.

3.2 Client-server architecture

We use a client-server architecture with 3 components as shown in Fig 2: a computing (BIDMach) engine, a web server, and a web based front end. BIDMach is implemented in the Scala language which supports concurrency with high level "actor" primitives. Another thread runs in the same Scala runtime, communicating with the web server. This thread receives parameter updates from the web server, and updates the corresponding model training parameters. As the model is trained, primary and mixin cost functions are evaluated on minibatches, providing regular updates which are passed to the web server.

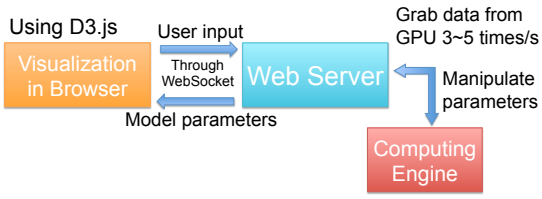


Figure 2: Visualization Architecture

In the client side, we implement a web based interface which uses D3.js[4] for data visualization. D3 is widely used, has very powerful graphics elements, and good support for animation. As a browser-based system, it runs transparently with a local or remote server. We will discuss the interface design in detail in the next section.

The communication between client and server is bi-directional, with both client and server initiating transfers. We therefore use WebSockets instead of e.g. a one-way RESTful web service. For simplicity and extensibility, we use JSON as the over-the-wire exchange format.

3.3 Secondary criteria as Mixins

Model customization is useful for both supervised and unsupervised problems. Unsupervised learning involves a certain amount of arbitrariness in the criteria for the "best" latent state. Therefore regularization is widely used as a secondary constraint on the primary objective [20, 2, 21, 26].

In a bit more detail, clustering algorithms like KMeans usually use the measure of model/centroid similarity, and may or may not use intra-model coherence measures or inter-cluster distance. Indeed, unsupervised learning models are often *evaluated* using a variety of criteria that are much more complex than the criteria used to derive the learning algorithms [22, 6]. The same holds true for topic models such as LDA, NMF and Word2Vec, and for collaborative filtering.

This is a paradox. Clearly one should get better scores for these criteria if they were directly optimized as part of training. Beyond these standard criteria, there are many others that are commonly used in the applications of machine learning. Historically it has probably been too difficult to optimize these criteria (the criteria may be expensive to evaluate, or non-locally computable).

On the other hand computing power is abundantly available now, especially in graphics processors. The bottleneck is often *moving* data rather than computing on it. Thus it is often practical to evaluate multiple, relatively complex criteria as part of optimization.

Combining these approaches, we can deal with a variety of secondary or "mixin" criteria as part of the learning process. In our present implementation, we use a linear combination of cost functions for primary and secondary criteria:

$$arg\min_x f(x,d) + \sum_i \lambda_i * g_i(x)$$

Where x is the model parameters, d is data, f is the primary cost function and g_i are the user-defined *Mixin* functions. The weights λ_i are "controls" that are dynamically adjusted by the analyst as part of training. For the primary criterion f and each secondary criteria g_i there should be at least one dynamic graphic that captures changes in that criterion in an intuitive way. The analyst watches these as each of the controls are adjusted to monitor the tradeoff between them.

4. INTERFACE DESIGN

In this section, we will describe the visual interface of our interactive machine learning system.

4.1 Visual dashboard

We use a dashboard approach where user can customize their own visualizations. As shown in Fig 3, the left side of the interface contains the menus and control sliders. From the menus, a user can select the metrics and controls for the modeling task. A corresponding control or metric visualization is then added to the dashboard, which can then be dragged, dropped and resized. There is at least one corresponding performance indictor for each control parameter, and more than one can be added to the dashboard. The details of each visualization component will be described next.

4.2 Visualizing the model

Directly visualizing the model provides a nice summarization for the dataset and users can gain a general understanding about the behavior of the algorithms. It can also help identify obvious errors and verify assumptions or intuitions. While there are different types of data and algorithm, the visualizations are necessarily model-specific, and should provide a natural interpretation of the model directly. For image data, the cluster centers can usually be visualized as in Fig 4a. For dictionary learning algorithms like NMF[18], the learned image dictionary can also be directly visualized as in Fig 4b, which is a much sparser representation. For more general matrix data, a simple direct visualization of element weights can work well. This was the approach taken

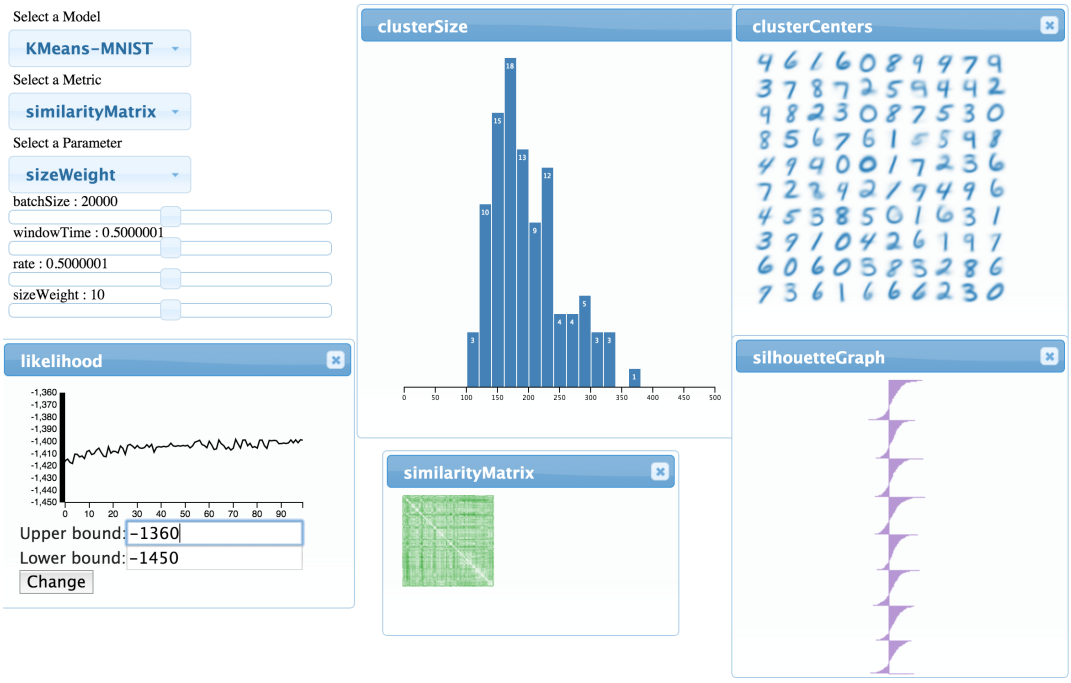
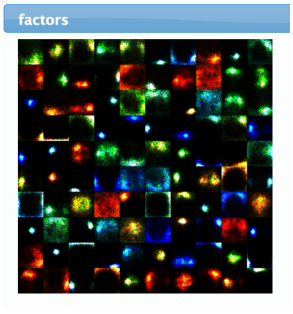


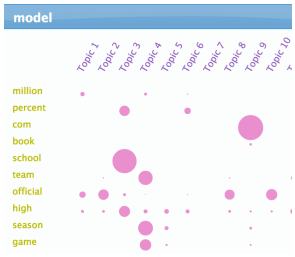
Figure 3: Dashboard for KMeans using MNIST dataset

in the "termite" system [7] and we use it also for our topic model visualization.

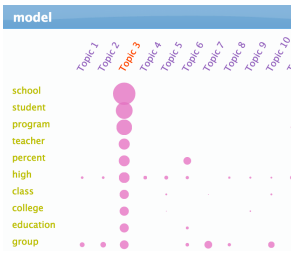
(a) MNIST clustering centers



(b) Dictionary learned by NMF on CIFAR dataset



(c) Ranking by global frequency



(d) Ranking within a topic

Figure 4: Model visualization

The topic model visualization follows the design from [7], as shown in Fig 4c and 4d. The radius of each pink circle in row i and column j encodes the weight for word i in topic j. We also display the word itself on the left side of each row to help people interpret the numbers. One common challenge for visualizing topic model comes from the huge size of the model. Typically hundreds or thousands of topics and even tens of thousands of different words. Limited screen size and limited human perception power require us to filter out information according to some saliency metric[7]. The metric will provide an ordering for words and topics and only the most important words and topics are displayed. Also, such an approach will significantly reduce the the amount of data that need to be transferred from the server to the client.

As we are displaying in real-time an evolving model, it is important to have a smooth and consistent word order after each update. We therefore only use the original topic weight to get the order of the words. In order to support a detailed zoom in for each topic, we also support ranking within a topic. This feature will be triggered when user mouse over the topic title, as shown in Fig 4d.

4.3 Continuous visualization of model quality

As described in 3.3, we are optimizing an additive function which consists of a main loss term as well as several *Mixin* terms. The value of the cost function can reflect how the model behaves under each criteria. As the engine keeps computing the update, we visualize those metrics as streaming data, as shown in Fig 5. Visualizing the main loss function is very important when we change the control parameters. It will reflect how algorithm responses to the user control and whether the tradeoff for *Mixin* functions may affect the general model performance.

As discussed earlier, the main loss function is computed on each minibatch, and is a single scalar. We use a simple, dynamic curve plot to display its state. This style of plot is easy to read and understand. With the parameters fixed, one normally sees a rapid initial increase in likelihood, followed by a slow increase on a plateau as in fig 5.

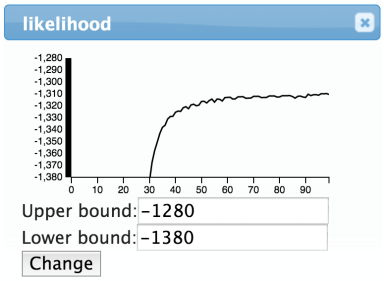


Figure 5: Continuous visualization of the likelihood function

4.4 Visualizing other performance indictors for Clustering

As mentioned above, the evaluation of unsupervised clustering algorithm is hard in general. Therefore showing multiple aspects of the clustering results at the same time would help users better interpret the model.

4.4.1 *Cluster size distribution*

To examine the cluster size balance, we are interested in the *distribution* of cluster sizes. The natural visualization for this kind of data is a histogram or kernel density plot. Fig 6a shows the size distribution for the clusters of digits on the MNIST dataset.

4.4.2 *Silhouette graph*

Another useful metric is the widely used silhouette graph (Fig 6b) for evaluating clustering results. The silhouette score is calculated for each data point x_j as:

$$s_j = \frac{b_j - a_j}{max(a_j, b_j)}$$

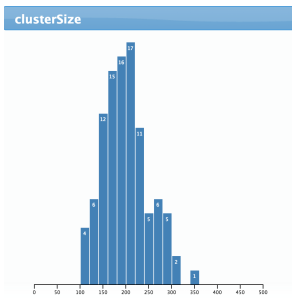
Where a_j is the average distance between x_j and all other data points in the same cluster, b_j is the lowest average distance of x_j to any other cluster. It could be seen that $-1 \leq s_j \leq 1$ and larger value indicates better clustering results.

4.4.3 *Recovered images from NMF*

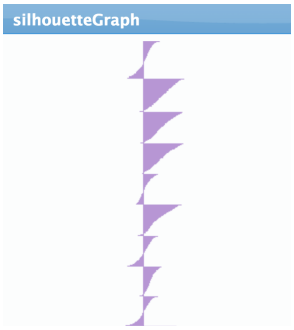
For the Non-negative Matrix Factorization algorithm[18], it is easy to recover the approximated matrix by multiplying the two factorized matrixes. Although the quality of the recovered matrix can be measured using L2-distance between the original matrix and the approximated one, directly showing the recovered result for image data (Fig 6c) can provide more details about what kind of information is being lost or preserved.

4.5 Slider controls

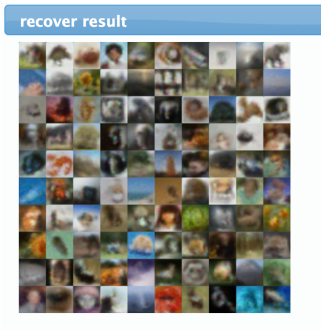
Along with the visual interface, we also provide several kinds of control including weights for *Mixin*, learning rate, etc. We also provide temperature control by changing the sample variance of the Gibbs sampler using SAME sampling[27]. So far these are all continuous scalars, and are all implemented as slider widgets. When the user select one of the controls from the menu, a labeled slider widget is created on the dashboard. The user drags and resizes this widget in an appropriate area of their dashboard. The selection of controls is currently independent of the selection of related



(a) Distribution of Cluster Size



(b) Silhouette graph



(c) Recovered images from NMF on CIFAR dataset

Figure 6: Performance indictors for clustering

metrics, and they are placed separately as well. In future we will explore intuitive ways of linking them (e.g. highlighting related metrics when a control is selected and vice-versa, or moving them as a group).

5. USE CASES

In this section, we will demonstrate several representative use cases for our system.

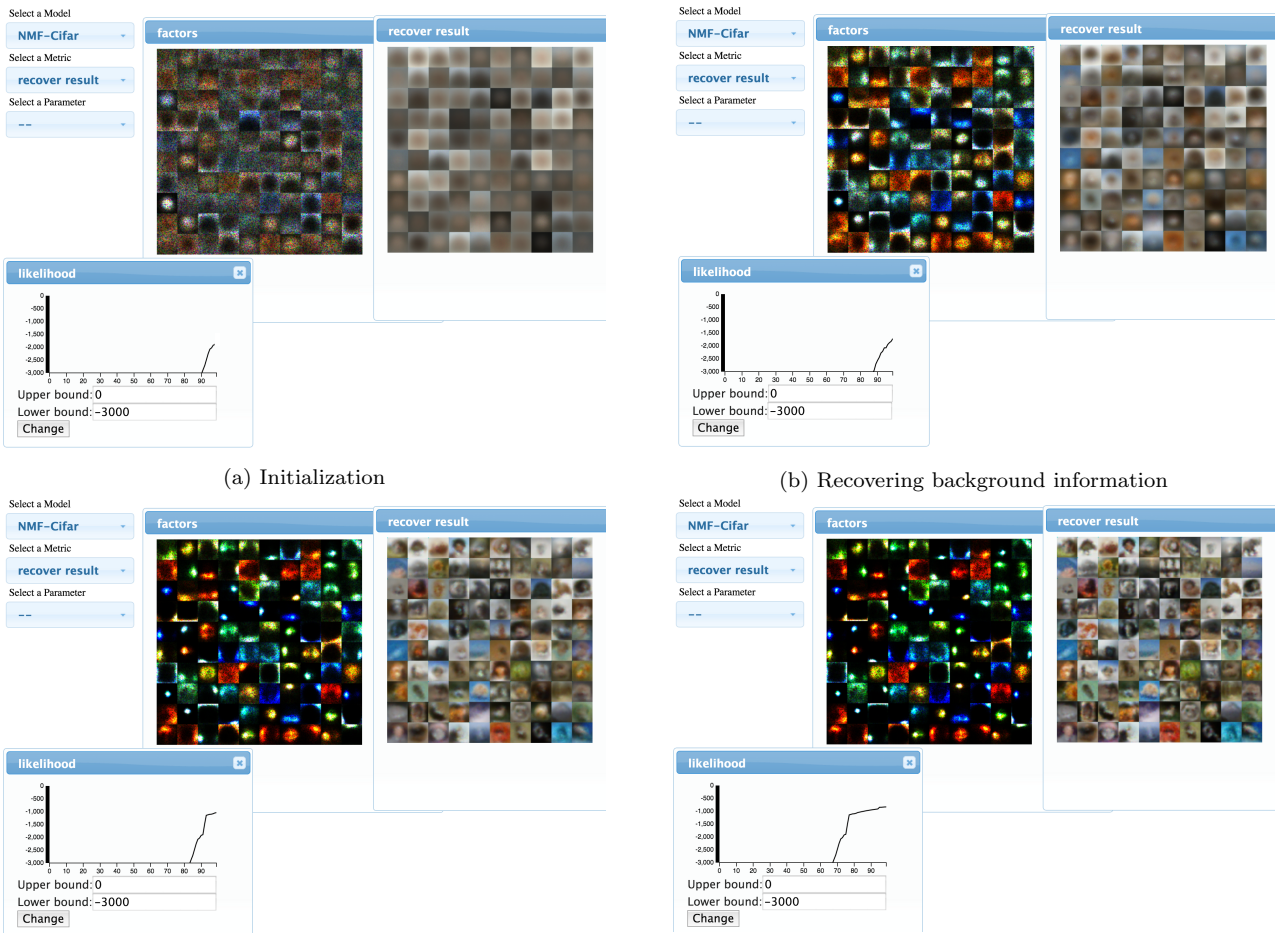
5.1 Non-negative Matrix Factorization

Our first demo is to monitor model update of the NMF algorithm in real-time. Although in this demo we don't have interactive control to the algorithm, we will still demonstrate how we can gain insight by viewing the learning process.

The NMF algorithm is trying to find low rank representation of the data matrix V by factorizing it into a product of two matrixes W, H with lower dimension. This is done by solving the optimization problem using multiplicative updates described in[18]:

$$arg \min_{W,H} ||V - WH||_2$$

This technique is very powerful in practice and can be used to find clusters of local patches for image data, or topics for text data. In our demo, we apply the NMF algorithm on the CIFAR-100 dataset[15]. The CIFAR dataset contains 50000 32*32 tiny RGB images which is stored in a 3072*50000 matrix. As we set the number of factors to be 256, the data matrix will be factorized into a 3072*256 matrix which is the dictionary for images and a 256*50000 matrix which contains the weights for each original image. Therefore each



(a) Initialization

(b) Recovering background information

(c) Recovering texture and edge

(d) Details become clearer

Figure 7: Online update for NMF

original image can be approximated as a linear combination of the 256 template images.

We train the model using a NVIDIA GTX-690 GPU with 2GB graphics memory, which costs less than a second for one iteration on the entire dataset, achieving around 150 GFlops. Such speed allows us to monitor real-time update as the model converges. We visualize the first 100 template images as well as the first 100 recovered images to help interpret what information is being extracted from the dataset. We also visualize the L2-loss $||V - WH||_2$ as a quantitative measurement.

The four representative stages of the training procedure is shown in Fig 7. Initially, the weight matrix are set to all one. Therefore a reasonable local optimal is to find some averaging images as the dictionary, and the recovered results all look similar, as shown in Fig 7a. Later in the procedure, the algorithm begins to recover background information as shown in Fig 7b. The sparseness of NMF model can also be seen since most pixels in the dictionary are black. This is consistent with [17] that the NMF algorithm always learns parts of the images. More texture and edge information will then be learned in the third stage, shown in Fig 7c. Some of the recovered images gradually become recognizable. Also, the template images clearly fall into two categories. One captures background information and the other represents tiny local patterns. Finally, in the last stage (Fig 7d), more details are being recovered and the L2-loss is about to converge. User now can trade off the model quality and training time based on their requirement.

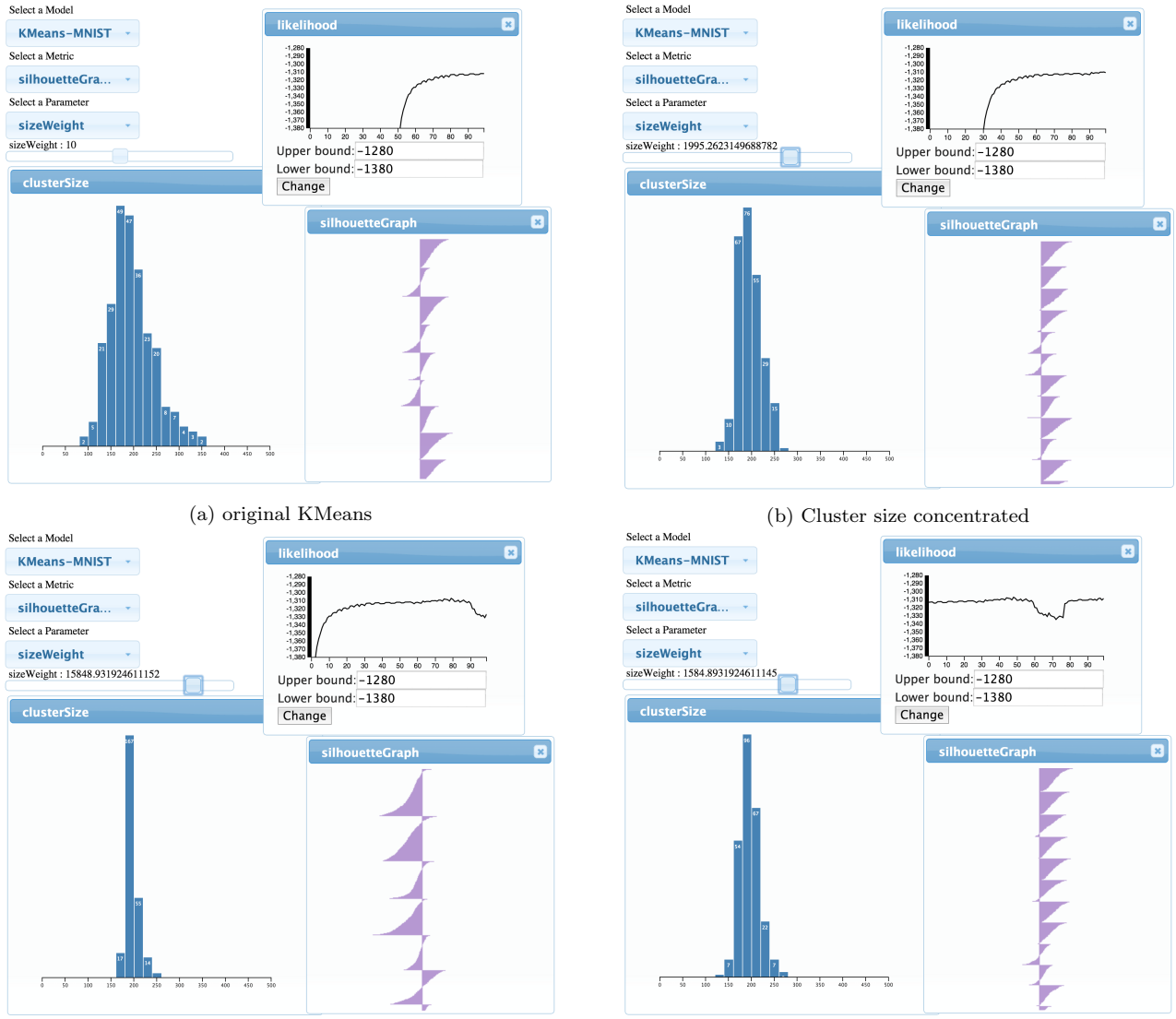
Comparing to the L2-loss metric, directly visualizing the model provides much more information than a single scalar. Users can gain more insight about the algorithm behavior by viewing the whole training process, which could help them make better decisions in the future.

5.2 KMeans

Our second demo is KMeans. This time we will have some direct control to the model which can demonstrate the full power of interactive clustering.

5.2.1 Evaluation and implementation for KMeans

For KMeans, the primary loss function is inertia: the sum of squared distances from points to their centroid. And the algorithm is straightforward: iteratively assigns data points to clusters and updates the cluster centroids using the mean of data.



(a) original KMeans

(b) Cluster size concentrated

(c) Begin to loss performance

(d) Recover the model

Figure 8: Interactive tuning for KMeans

However, as mentioned previously, the evaluation and tuning of the KMeans algorithm turn out to be hard. We therefore use multiple criteria to examine different aspects of the clustering results. Aside from the main loss, we use silhouette graph to measure how tight it is for each cluster. A low silhouette score typically indicates small clusters at the periphery of larger ones.

Besides, we also use cluster size balance as a criteria, which could be naturally visualized with a histogram. This criteria can also be optimized within the KMeans algorithm by adding size as a secondary loss. The overall minimization problem then becomes:

$$id(x) = arg\min_i ||cluster_i - x||_2^2 + \lambda * size_i \qquad (1)$$

$id(x)$ is used to assign a cluster id for each data point x. Where $size_i$ represents how many data points have already been assigned to the i-th cluster, $cluster_i$ is the center of the i-th cluster.

The loss term $\lambda * size_i$ penalizes clusters with a larger size. The algorithm would prefer to assign new data points to a smaller cluster, which will tend to balance cluster sizes over time. Size homogeneity matches most users' intuition about clustering, and it may also be important for accurate estimation of cluster statistics. The λ also becomes a parameter that we can interactively tune.

5.2.2 *Incremental update*

In order to take the advantage of mini-batch processing which can return early feedback during the training, we follow a similar approach to [28] for incremental KMeans updating. And we also need an averaging update for $size_i$ to maintain its scale and make the visualization consistent. For

each batch $\{x_j\}$, we compute the update as:

$$average_i = \frac{\sum_j x_j * \mathbb{1}(id(x_j) = i)}{\sum_j \mathbb{1}(id(x_j) = i)}$$

$$cluster_i = cluster_i + \eta * average_i$$

$$size_i = size_i + \alpha * \sum_j \mathbb{1}(id(x_j) = i)$$

Normally η and α are set to $0.1 \sim 0.2$.

5.2.3 Experiment on MNIST dataset

We ran an experiment on the MNIST dataset [16], which contains 8 million 28×28 images of hand written digits. We train the model using NVIDIA GTX-690 GPU, which could process roughly 500MB raw data (16.7k images) per second. As we set the batch size to be 50000, every second the system could perform 3 batch updates, which is enough for real-time visualization.

In our dashboard, we choose to visualize the main loss, cluster size distribution, as well as the silhouette graph. The main loss is the averaging distance between the data points and their assigned cluster centers (not taking into account size).

The parameter that we choose to tune is the size weight λ in eq(1), which we refer as $sizeWeight$ in the interface. Initially, we set the number of clusters K as 250 and we assigned a small value to $sizeWeight$, so that the algorithm will behave as the original KMeans algorithm. As shown in Fig 8a, the likelihood is gradually improving and it quickly converges to local minimal. The cluster size distribution is quite diverse. We then increase the $sizeWeight$ slightly, which gives us a more concentrated cluster size distribution, while the main loss and silhouette score are not affected, as shown in Fig 8b. This implies that the algorithm now moves to another local optimal by assigning some data points to a suboptimal but smaller cluster. Notice that this has almost no effect on the primary likelihood.

However, as we continue to increase $sizeWeight$, the loss will start to increase, and the silhouette graph also shows more defects (more negative area) (Fig 8c). From here, we decrease the $sizeWeight$. Since the KMeans algorithm is incremental, this change brings us back close to the likelihood before the last increase, as shown in Fig 8d. This example illustrates the tradeoffs that can be made, and the speed of recognizing poor parameter choices.

5.3 L1-regularized Topic Model

Our last demo is applying our system to Latent dirichlet allocation(LDA) topic modeling [3], one of the most widely used topic models.

5.3.1 Implementation

LDA is a generative process to model the documents. For each document d, it proceeds as follows (K is the number of latent factors):

- Draw a topic distribution for the document d as $\theta_d \sim Dirichlet(\alpha)$, a K-dimensional Dirichlet.
- For each word position i (across all docs), draw a topic index $z_{d,i} \in \{1, ..., K\}$ from $z_{d,i} \sim \theta_d$
- Draw the word $w_{d,i}$ from the multinomial distribution $w_{d,i} \sim \varphi_{z_{d,i}}$, which also has a prior: $\varphi_{z_{d,i}} \sim Dirichlet(\beta)$

α, and β are hyper-parameters specifying the Dirichlet prior. The other two parameters of the model are θ, which can be represented as a document-topic matrix, and φ, the word-topic matrix. The algorithm therefore is to use Gibbs sampler to draw samples for hidden states z_i:

$$P_X(z_{d,i} = k | z_{-d,i}, \alpha, \beta, x_{d,i} = w) \sim \theta_{d,k} * \varphi_{k,w} \qquad (2)$$

After drawing the samples, θ and φ can be updated via Maximum Likelihood Estimation. In order to apply annealing to the optimization procedure, we further use SAME sampling [27] to draw m independent samples instead of just one from Z each time. This results in a cooled Gibbs sampler and the parameter m can be used to control the temperature. A low value of m gives a higher-variance random-walk while increasing m can cause parameters converge to a nearby optimum. By default m is set to 100, and it can be tuned during the training. We refer m as $nsamps$ in the interface.

Similar to KMeans, we also use incremental update for LDA as described in [11]. A L1-regularization is also added into the model to enforce sparsity. As describe in 3.3, the implementation is very straightforward. For each batch, after computing the model update φ', we also compute the subgradient update for the L1-regularization:

$$g(\varphi_{k,w}) = -\lambda * sign(\varphi_{k,w})$$

We then add those two terms φ' and $g(\varphi)$ into the model using the weighted averaging approach we discussed above. We refer the weight λ as $L1 - reg$ in the interface.

5.3.2 Experiment on NYTimes dataset

We run an experiment on the NYTimes dataset [19], which contains about 300K documents, 102K different words and totally 100M tokens. Again, we train our model using GTX 690 GPU.

We first set topic number K as 1024 and the model converged in about two minutes. As shown in Fig 9, the resulting topic matrix is very sparse even without adding the L1-regularization. This is due to that we set a large K and many independent topics are generated.

We then set K to be 32. Without any regularization, the algorithm's behavior is shown in Fig 10. The likelihood quickly converges to a local optimal, but the topic results are still very noisy, and the topics are overlapping. We then adjust the $L1 - reg$ slider away from zero. However, tuning $L1 - reg$ will not always give good results on complex likelihood functions due to local optima. Since SAME sampling is used in our LDA implementation, we can decrease $nsamps$ to increase the variance of the random-walk, which makes it easier to jump between possible solutions.

After we increase the temperature, as shown in Fig 11, the likelihood drops significantly but we get a very sparse model.

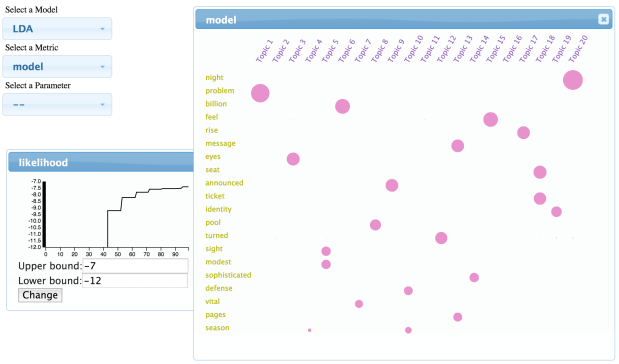


Figure 9: Converged sparse model, K=1024

Afterward, we set the $L1-reg$ back to a small value and use a large *nsamps* (cold state) which prevents large changes in model state. This is equivalent to only allowing the model to make very small movement around that local optimal. From Fig 12, we can see that the likelihood returns to a normal value while the sparsity is maintained.

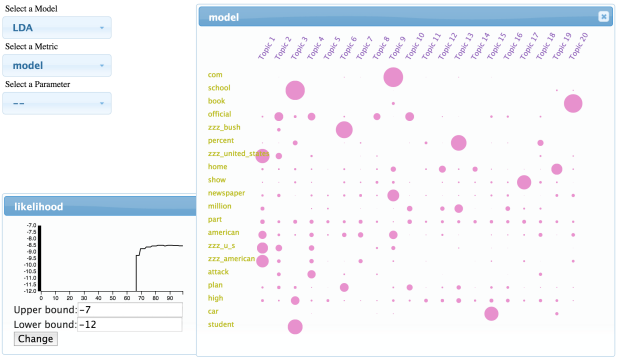


Figure 10: Overlapping topics, K=32

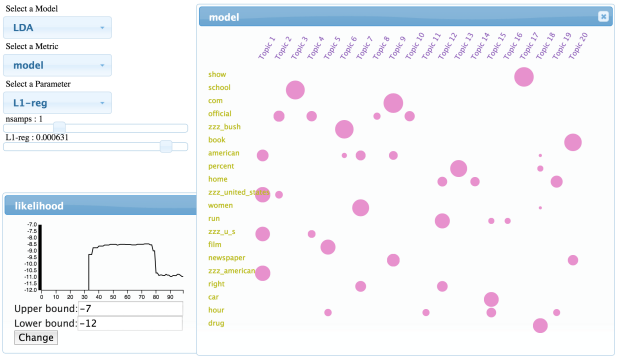


Figure 11: High temperature with high L1-reg

6. CONCLUSION & FUTURE WORK

We have demonstrated how to perform interactive optimization on customized models using our system. The *Mixin* function is a convenient and useful way to capture user's intuition and create customized model. The dashboard also enable users to easily monitor several performance indicators at the same time, which help users to evaluate the model

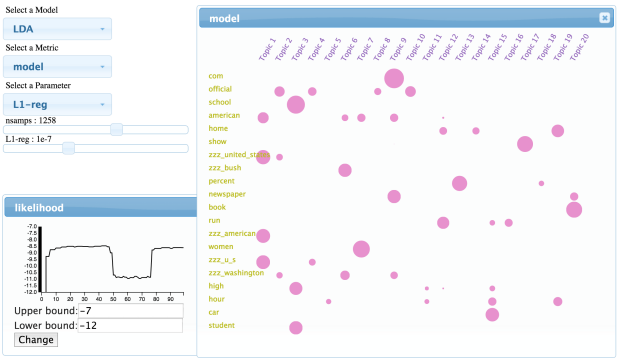


Figure 12: Likelihood back to normal, sparsity preserved

from different perspectives. By summarizing this information, the trade-off decisions become straightforward. Also, the GPU accelerated toolkit makes it possible to get real-time feedback. This ensures that users can understand cause and effect of the algorithm behavior in an iterative refinement procedure.

More *Mixin* functions like measuring independence of factors, or coherence of topics could be implemented in the future. Also, our framework is not limited to the unsupervised learning algorithms. Some concrete examples of competing goals in supervised learning include computational marketing where the primary goal is maximize revenue, but where secondary goals include user satisfaction, advertiser satisfaction, and budget constraints. Recommender systems seek to recommended the highest-rated items, but may also need to cover the available item inventory, favor more or less expensive items, or favor items which encourage future purchases. Those are all potential extensions that could be explored in the future.

7. REFERENCES

[1] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4):105–120, 2014.

[2] D. Andrzejewski, X. Zhu, M. Craven, and B. Recht. A framework for incorporating general domain knowledge into latent dirichlet allocation using first-order logic. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1171, 2011.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[4] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011.

[5] J. Canny and H. Zhao. Big data analytics with small footprint: Squaring the cloud. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–103. ACM, 2013.

[6] J. Chang, S. Gerrish, C. Wang, J. L. Boyd-graber, and D. M. Blei. Reading tea leaves: How humans interpret topic models. In *Advances in neural information*

processing systems, pages 288–296, 2009.

[7] J. Chuang, C. D. Manning, and J. Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 74–77. ACM, 2012.

[8] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

[9] J. A. Fails and D. R. Olsen Jr. Interactive machine learning. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 39–45. ACM, 2003.

[10] J. E. Gonzalez, Y. Low, H. Gu, D. Bickson, and C. Guestrin. Powergraph: Distributed graph-parallel computation on natural graphs. In *OSDI*, volume 12, page 2, 2012.

[11] M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[12] Y. Hu, J. Boyd-Graber, and B. Satinoff. Interactive topic modeling. In *Association for Computational Linguistics*, 2011.

[13] A. Kapoor, B. Lee, D. S. Tan, and E. Horvitz. Performance and preferences: Interactive refinement of machine learning procedures. In *AAAI*. Citeseer, 2012.

[14] T. Kraska, A. Talwalkar, J. C. Duchi, R. Griffith, M. J. Franklin, and M. I. Jordan. Mlbase: A distributed machine-learning system. In *CIDR*, 2013.

[15] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 1(4):7, 2009.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[18] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[19] M. Lichman. UCI machine learning repository, 2013.

[20] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *Proceedings of the 17th international conference on World Wide Web*, pages 101–110. ACM, 2008.

[21] D. Newman, E. V. Bonilla, and W. Buntine. Improving topic coherence with regularized topic models. In *Advances in neural information processing systems*, pages 496–504, 2011.

[22] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

[23] H. Raghavan, O. Madani, and R. Jones. Interactive feature selection. In *IJCAI*, volume 5, pages 841–846, 2005.

[24] J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results [gene identification]. *Computer*, 35(7):80–86, 2002.

[25] E. R. Sparks, A. Talwalkar, V. Smith, J. Kottalam, X. Pan, J. Gonzalez, M. J. Franklin, M. I. Jordan, and T. Kraska. Mli: An api for distributed machine learning. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 1187–1192. IEEE, 2013.

[26] Y. Yang, S. Pan, Y. Song, J. Lu, and M. Topkara. User-directed non-disruptive topic model update for effective exploration of dynamic content. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 158–168. ACM, 2015.

[27] H. Zhao, B. Jiang, and J. F. Canny. SAME but different: Fast and high-quality gibbs parameter estimation. *CoRR*, abs/1409.5402, 2014.

[28] S. Zhong. Efficient online spherical k-means clustering. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 5, pages 3180–3185. IEEE, 2005.

[29] J.-Y. Zhu, Y. J. Lee, and A. A. Efros. Averageexplorer: Interactive exploration and alignment of visual data collections. *ACM Transactions on Graphics (TOG)*, 33(4):160, 2014.

Opinion Marks: A Human-Based Computation Approach to Instill Structure into Unstructured Text on the Web

Bum Chul Kwon
Universität Konstanz
bumchul.kwon@uni-konstanz.de

Jaegul Choo
Korea University
jchoo@korea.ac.kr

Sung-Hee Kim
Purdue University
kim731@purdue.edu

Daniel Keim
Universität Konstanz
keim@uni-konstanz.de

Haesun Park
Georgia Tech
hpark@cc.gatech.edu

Ji Soo Yi
Purdue University
yij@purdue.edu

ABSTRACT

Despite recent improvements in computational approaches such as machine learning, natural language processing, and computational linguistics, making a computer understand human-generated unstructured text still remains a difficult problem to solve. To alleviate the challenges, we propose an approach called "Opinion Marks," which enables writers to mark positive and negative aspects of a topic on their own text. In addition, Opinion Marks incorporates an automatic marking suggestion algorithm to offload a user's marking effort. The phrases marked with Opinion Marks can be further used to clarify the sentiments of other text in a similar context. We implemented Opinion Marks at a question answering website `http://caniask.net`. To test the efficacy of Opinion Marks, we conducted a crowdsourced experiment with 144 participants in a between-subject design under three different conditions: 1) human marking only; 2) machine marking only (automatic marking suggestion); and 3) human-machine collaboration (Opinion Marks). This study revealed that Opinion Marks significantly improves the quality of marked phrases and usability of the system.

Keywords

human-based computation; user interface; crowdsourcing

Categories and Subject Descriptors

H.5.m. [**Information Interfaces and Presentation (e.g. HCI)**]: Miscellaneous

General Terms

Human Factors; Design; Experimentation.

1. INTRODUCTION

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

On the web, user-generated, unstructured text documents (hereinafter called text) are being rapidly generated in various forms such as product reviews (e.g., `http://www.amazon.com`), question answering (e.g., `http://www.ask.com`), and discussion forums (e.g., `http://www.ubuntuforums.org`). As the volume of text grows, it is becoming more challenging for people to efficiently grasp useful information such as others' opinions and recommendations.

Computational approaches developed by natural language processing and machine learning have made notable progresses in delivering a great quality of summaries out of copious text. However, these techniques done in a fully automated manner bear inherent limitations in capturing the true semantic meanings and intentions that writers intend to convey.

Alternatively, beyond fully automated computational approaches, human-computing approaches [28], which leverage human capabilities in computational steps, have been gaining popularity. Many of the semi-supervised learning methods from machine learning areas assume human input as a main source of additional supervision, which often lead to significant improvement in the desired tasks.

Nonetheless, the main issue is how to support and encourage users so that they can effortlessly but accurately perform their human-computing tasks (e.g., assigning labels), while doing their original jobs (e.g., writing on the web). In the context of microblogging and social networking services, a representative example is a user-generated hashtag, a word tag with a prefix symbol "#". Through small efforts taken by analysts, hashtags have been shown useful when other users group and filter microblogs, which would otherwise be difficult [7, 13].

Motivated by such progress, we propose Opinion Marks, an advanced human-based computing technique that allows users to mark their opinions during writing processes in an efficient, user-friendly manner. Basically, Opinion Marks assumes three different possible components available in textual data representing humans' opinions: (1) a topic to be discussed, (2) a positive aspect, and (3) a negative aspect. Figure 1 shows that a writer can describe a positive aspect ("so sweet") as well as a negative aspect ("too rich") of a particular topic ("Ben & Jerry's") with three surrounding symbols (i.e., **# #**, + +, - - for a topic, a positive aspect, and a negative aspect, respectively).

The main goal of Opinion Marks is to provide convenient user interfaces through which users can easily mark the three components on their text they write in real time. Furthermore, in order to help users easily adopt our new technique and encourage their

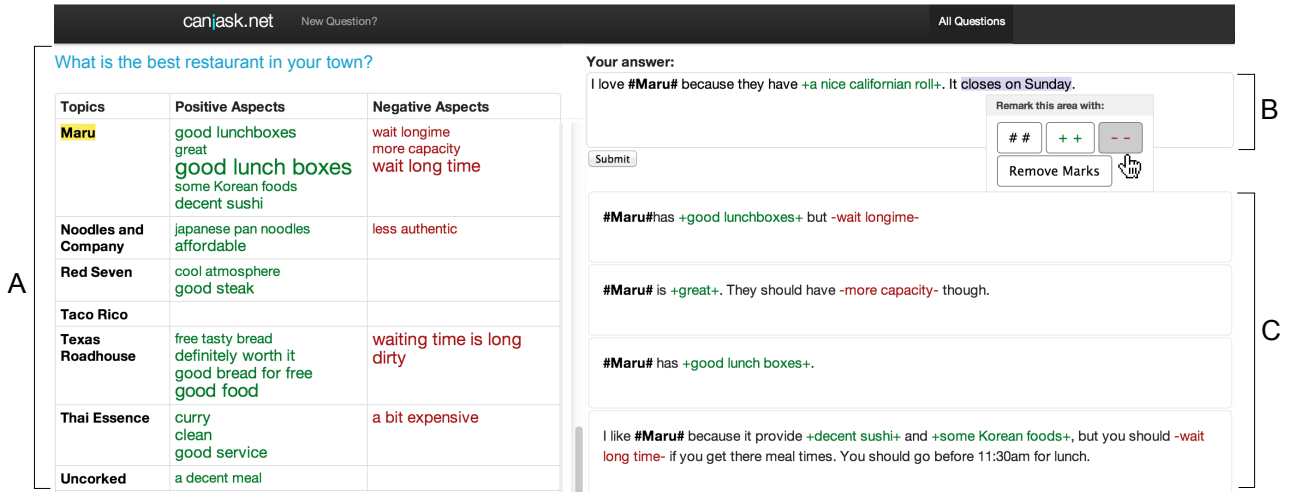


Figure 2: An overview of `http://caniask.net`. (A) Summary Table shows topics, positive aspects, and negative aspects in separate columns from left to right. Users can agree with each positive/negative aspect by clicking each entry, and the font size of each entry reflects the number of agreeing users. In addition, a user can sort columns and filter by topics and aspects, e.g., "Maru" in this case; (B) A user writes his/her own answer and can mark topics, positive aspects, and negative aspects via user interactions provided by Opinion Marks. Upon clicking "Submit," each of the marked phrases is added to Summary Table; (C) Previously created answers are listed. Currently, only those filtered by the topic, "Maru," from (A) were shown.

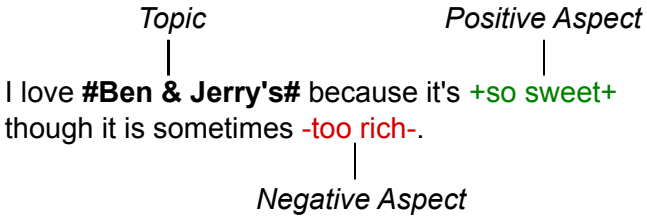


Figure 1: A sentence marked with Opinion Marks. A *topic*, a target entity to be discussed, is surrounded by # # and its *positive aspect* and *negative aspect* by + + and - -.

marking activities, Opinion Marks has an automatic marking suggestion capability. In detail, the novel features of Opinion Marks are as follows:

- As ways to mark topics and their positive/negative aspects during writing phases, we provide a graphical user interface, which is easily accessible by novice users, as well as an inline marking interface, which is geared towards efficient marking processes for experienced users.
- We integrate text mining and natural language processing techniques, such as part-of-speech tagging and lexicon-based sentiment analysis models, to provide automatic suggestions for default marking and encouraging user involvement. Our proposed model adapts itself in response to user corrections.

To deploy Opinion Marks in a familiar environment on the web, we have developed a website, **`http://caniask.net`**, which integrates Opinion Marks in a question answering type of web services (e.g., Quora). Similar to other existing question answering services, our system allows users to freely post questions and write answers to them. When writing an answer, the integrated capabilities of Opinion Marks help users mark topics and positive/negative aspects.

Our paper will present how we substantiated our idea of Opinion Marks in this web-based system and demonstrate our crowdsourced user study, which highlights the improvements in an adoption rate and an accuracy of our proposed technique due to automatic marking suggestion.

2. OPINION MARKS: AN INLINE MARKUP TECHNIQUE

To provide an effective means to instill structures to unstructured text, we identified two design criteria to maximize the user adoption: First, using this technique should not interrupt the natural process of writing text. Second, the technique should be intuitive enough for users to grasp and learn quickly. To meet the criteria, we develop an inline markup technique, called Opinion Marks. As a user writes an answer with Opinion Marks, including + + (positive aspect marks), - - (negative aspect marks), and # # (option marks), the marked text is captured as a positive aspect, a negative aspect, and an option, respectively. At the same time, the marks and marked text will change the colors to green, red, and bold black respectively on the fly, which is basically the same as the syntax highlighting feature in various text editors. When the answer is submitted, the enclosed text is captured by Opinion Marks, and the captured text can be used for the Summary Table, which will be described in a later section.

Our goal was to minimize the impact on one's normal writing behavior but to capture text segments without any ambiguity. That is why we dropped the twitter's '#' tagging because it does not show where the phrase ends (thus being only appropriate for collecting a single word). Furthermore, using a single plus or minus signs may conflict with other uses (e.g., "It's 30+ year old" or "I am a decision-maker"). Through several design discussion and pilot studies, we came to our conclusion that enclosing marks, represented by simple marks available on keyboards, work efficiently for users and for our system. Thus, we chose to use # #, + +, and - - with intuitive color schemes to increase users' correct adoption.

3. OPINION MARKS ON THE WEB

We introduce our publicly available, question answering website, **http://caniask.net**, which was used as a testbed for integrating Opinion Marks. We chose to implement a question answering service because we believe that we can induce online discussion evaluating different topics without implementing an full-fledged e-commerce website, which require too much implementation effort. However, we believe that Opinion Marks can be extended to other online services.

The main page of **http://caniask.net** (see Figure 2) consists of Textbox with Opinion Marks, Submitted Answers, Summary Table, and Instructions. In **Textbox with Opinion Marks (Figure 2(B))**, users can write actual answers and put markings via user interactions (with a mouse operation or an inline labeling) provided by Opinion Marks. User-marked phrases are highlighted in different colors as shown in Figure 1. Submitted responses are added to the area of **Submitted Answers (Figure 2(C))**, and the parsed results of the submitted response are added to the **Summary Table (Figure 2(A))**. Summary Table provides a comprehensive overview of topics and aspects captured by Opinion Marks from user-generated text. Each row represents a topic along with its collected positive (red-colored) and negative (green-colored) aspects marked by multiple users mentioning the same topic. The font size of positive/negative aspects encodes the number of agreements made by other users (similar to the Like button in Facebook). The Agree button appears when a user hovers around a positive/negative aspect. In addition, clicking on an aspect allows users to filter only the answering posts containing the corresponding keyword as shown in Figure 2(C). To help users better understand how to use Opinion Marks, we provided an instruction when a user fails to mark a topic and an aspect from a sentence. The instruction included two different ways to mark: typing and context menu. We also showed how those marked phrases are inserted into the summary table. All instructions are provided using animated GIF (Figure 3 shows selected clips from the instruction).

4. OPINION MARKS

To provide an effective means to instill structures to unstructured online text, we identified two design criteria to maximize the user adoption: First, using this technique should not interrupt the natural process of writing text. Second, using the technique should be intuitive enough for human users to grasp and learn quickly. Opinion Marks achieves these criteria by providing (1) two types of marking interfaces for both experienced as well as inexperienced users and (2) automatic marking suggestion that minimizes human marking efforts as well as encourages user participation in marking tasks. In **http://caniask.net**, we implemented Opinion Marks within Textbox. By using Opinion Marks, users can mark topics (what the questioner asked) and their aspects (good and bad things about the topic) on their own answers.

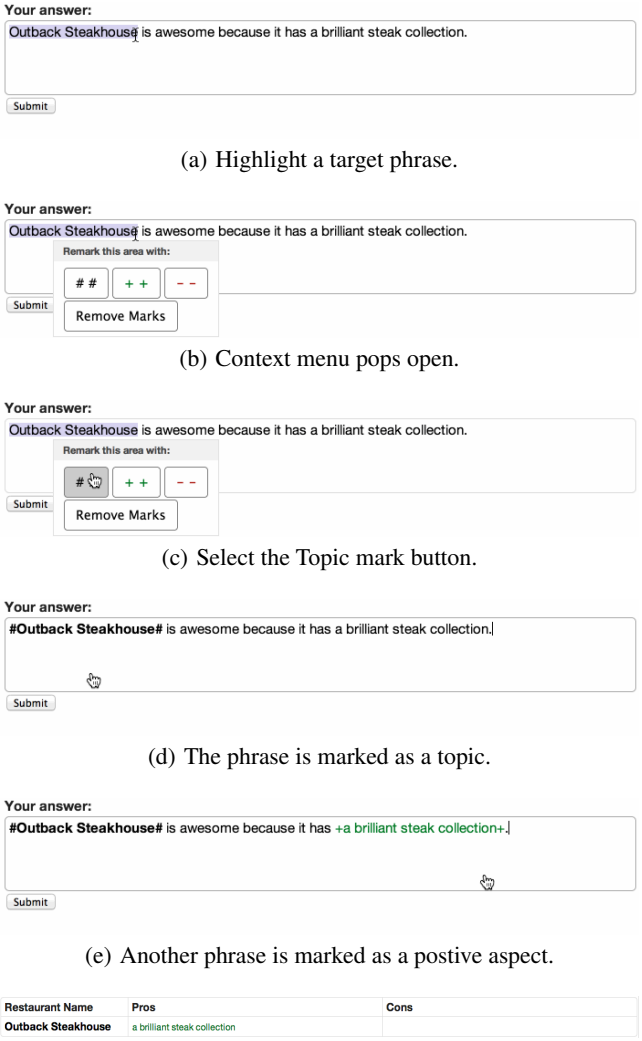
In the following, we describe the details of Opinion Marks in terms of the main concept, user interfaces, and an automatic suggestion module.

4.1 Main Concept

As described in the example in Figure 1, the main concept of Opinion Marks is to mark three types, a topic, its positive aspects, and its negative aspects, from unstructured text that users generate on the web. Such marking is seamlessly integrated into text itself by surrounding particular phrases with special characters **# #** (topic marks), + + (positive aspect marks), and - - (negative aspect marks). In this sense, Opinion Marks can be considered analogous to a widely-used hashtag, but it conveys more sophisticated information than a hashtag. Considering the great success of a hashtag in the information retrieval context, the marked text via Opinion Marks has significant potential in various text analysis processes such as sentiment analysis and summarization.

We chose these three special characters, **#**, +, and -, for Opinion Marks because they are more intuitive than other less frequently used characters (e.g., ^ (caret)). However, the special characters may conflict with other uses of them (e.g., "It's 30+ year old" and "I am a decision-maker"). In order to avoid such conflicts, we detected these characters for Opinion Marks only when they are shown at the beginning of the entire text or preceded by a single white space.

4.2 User Interface



(a) Highlight a target phrase.

(b) Context menu pops open.

(c) Select the Topic mark button.

(d) The phrase is marked as a topic.

(e) Another phrase is marked as a postive aspect.

(f) Submission appears on the summary table.

Figure 3: An example of marking target phrases using the context menu on Textbox

To support marking processes in a user-friendly and efficient manner, Opinion Marks provide two different user interfaces. The first one is to just let users put the corresponding special characters directly in their text during the writing phase. When a user is familiar with Opinion Marks, this type of a user interface works

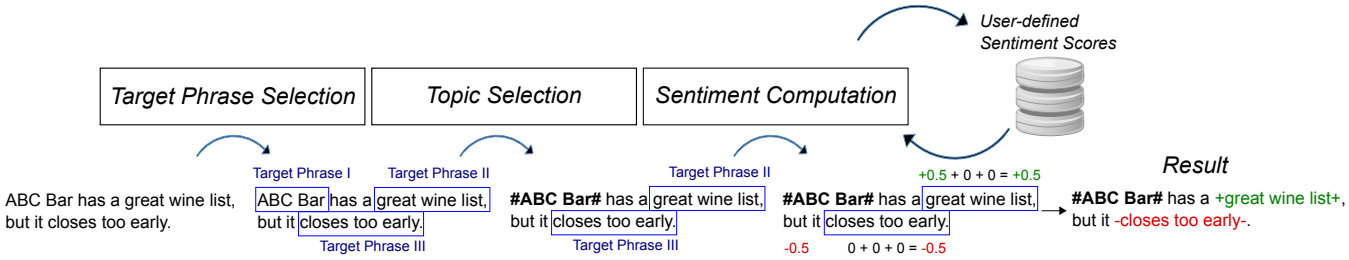


Figure 4: The flow diagram of phrase detection and sentiment computation algorithm

as the most efficient, straightforward means for marking. In addition, Opinion Marks changes the color of the marked text to bold black (for topics), green (for positive aspects), or red (for negative aspects) on the fly, which is similar to syntax highlighting features in various text editors. In order to further help users, we provided a drop-down menu when a user types the starting mark; the drop-down menu shows topics or aspects entered previously by users who answered the same question.

The second user interface is via a context menu. As Figure 3 shows, once a user highlights a particular phrase via a mouse drag-and-release operation, a custom-designed context menu pops up where s/he can select one of the four options: *topic*, *positive*, *negative*, and *remove*. The highlighted phrase will then be marked with the selected option. This user interface provides an inexperienced user with an easy, intuitive interface to select one among the three supported marking types. Furthermore, as will be described in the next section, this type of a user interaction via mouse drag-and-release can also be used to efficiently rectify automatically suggested markings on given text.

4.3 Automatic Marking Suggestion

In addition to the supported user interfaces, we have developed an automatic marking suggestion approach for the parts of text that have to be potentially marked but are not done so yet by users. Given that any single state-of-the-art techniques cannot fully catch all the human semantics and intent, the main purpose of this module is not to generate perfectly accurate markings in a fully automated manner, but deliver to users our best-effort candidates to be marked so that we can mitigate human efforts and encourage user participation in marking processes. In this manner, even if an inexperienced user does not mark his/her text, this module will provide machine-suggested markings with a reasonable quality, which would make users naturally learn how the system works, revise the suggested markings, and generate their own markings that were missed by the automatic module.

As a first step to create the module, we first deployed and released a minimalistic version of **`http://caniask.net`** in the wild. Then, we investigated how users marked phrases into topics, positive aspects, and negative aspects. Based on our study, our automatic marking suggestion module is built upon three steps: (1) topic/aspect candidate generation, (2) topic selection, and (3) aspect selection based on adaptive sentiment computation. Figure 4 shows the overview of steps of how our automatic marking suggestion works. In the following, we describe each step in detail.

Topic/aspect candidate generation. Given a user-generated textual post, e.g., a single tweet, review article, or comment, this step generates a set of candidate phrases for marking. Basically, we want each of our candidate phrases to be the most meaningful but shortest possible phrase that keeps the author's intent intact. From our preliminary user study where we asked participants to mark text using Opinion Marks (with no automatic suggestion), we found that a majority of marked phrases are noun phrases and verb phrases. Thus, we created a rule-based algorithm that can mimic the marking behavior.

In detail, we first parse each sentence and obtain a part-of-speech (POS) tag[1] for each word. Then, we find the main verb for the sentence. If this verb is contained in our predefined set of insignificant verbs with any tense, we exclude them from our candidate phrases since they either do not add much meaning (e.g., be, do, and have) or their meanings can be represented via our positive/negative marks (e.g., like, love, hate, and dislike).

Next, we find nouns or noun phrases used as objects. Then, for each of them, we find and include preceding nouns, pronouns, and adjectives because they are likely to add meanings to it. For the same reason, we include adverbs and preposition phrases following each of main nouns. Throughout the process, we achieve a set of candidate phrases $P = \{p_1, p_2, \cdots, p_n\}$, where p_i is in an order of appearance in d.

In addition, we also compare this candidate phrase set with topics and phrases that are marked by other users previously. If we find a more comprehensive and inclusive phrase in previous phrases, then we used the phrase instead. This was done to reflect human-marked phrases from other users.

Topic selection. The next step is to determine a topic phrase p_t from P. An important assumption here is that a user discusses only a single topic in each post d. For instance, in case of an online review, a restaurant review, even though a user could discuss various aspects such as service, food quality, atmosphere, we assume that s/he talks about a particular restaurant, which would be marked as a topic.

Based on this assumption, we select p_t as the first appearing p_i (the smallest i value) such that p_i satisfies either of the two conditions: (1) p_i is tagged as a subject in its corresponding sentence or (2) any noun in p_i starts with a capital letter. When two condition conflicts, we prefer (2) because the capitalized nouns show the author's more explicit intent. For example, if an author writes "I feel McDonald's is great because they have cheap and nice burgers", then McDonald's will be captured as a topic because this is the first capitalized noun that appears in the sentence.

Aspect selection with adaptive sentiment computation. Now, we select positive or negative aspects from $P \setminus \{p_t\}$. Our basic approach is a lexicon-based sentiment analysis approach that we modified from a previously developed algorithm [23]. That is, we compute an overall sentiment score $S(p_i)$ for p_i by aggregating the word-level sentiment scores for words contained in p_i. Specif-

[1]We used Stanford Natural Language Processing library available at **`http://www-nlp.stanford.edu/software/index.shtml`**.

ically, suppose p_i is composed of a sequence of m_i words, i.e., $p_i = \left(p_{i,1}, p_{i,2}, \cdots, p_{i,m_i}\right)$ where $p_{i,j}$ represents the j-th word in p_i. Assuming that an word-level sentiment score $s\left(w_j\right)$ for a word w_j is defined, $S\left(p_i\right)$ is computed as

$$S\left(p_i\right) = \sum_{i=1}^{m_i} s\left(p_{i,j}\right).$$

After computing $S\left(p_i\right)$ for each p_i in $P \backslash \{p_t\}$, we select positive aspects as those p_i's with $S\left(p_i\right) \geq \delta_+$ and negative ones for those with $S\left(p_i\right) \leq \delta_-$ where we set $\delta_+ = 0.3$ and $\delta_- = -0.3$.

Now, let us describe how we determine $s\left(w_j\right)$, which we call a crowd-driven sentiment score. Initially, we start with a set of words each of which, w_j, is assigned a predefined score $\hat{s}\left(w_j\right)$ ranging from -1 (negative) to 1 (positive)[2]. Starting with these predefined scores, we adaptively adjust our crowd-driven sentiment score $s\left(w_j\right)$ for w_j during user marking processes, i.e.,

$$s\left(w_j\right) = \begin{cases} \hat{s}\left(w_j\right) & \Delta\left(w_j\right) \leq 0 \\ \left(1 - \frac{\Delta\left(w_j\right)}{N}\right)\hat{s}\left(w_j\right) & 0 < \Delta\left(w_j\right) \leq N \\ -\operatorname{sgn}\left(\hat{s}\left(w_j\right)\right)\frac{\exp(k(\Delta(w_j)-N))-1}{\exp(k(\Delta(w_j)-N))+1} & \Delta\left(w_j\right) > N \end{cases} \quad (1)$$

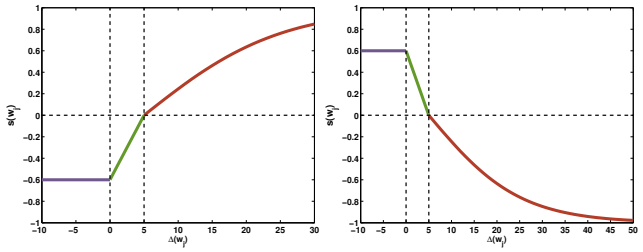
where k and N are parameters (e.g., $k = 0.1$ and $N_{thres} = 5$ in our case). In addition, $\Delta\left(w_j\right)$ is defined as

$$\Delta\left(w_j\right) = -\operatorname{sgn}\left(\hat{s}\left(w_j\right)\right)\Delta_{p-n}\left(w_j\right)$$

where $\Delta_{p-n}\left(w_j\right)$ represents the occurrence count of w_j among existing positive aspects minus that of w_j among existing negative aspects. Intuitively, the value of $\Delta\left(w_j\right)$ represents how often opposite sentiments have been observed relatively to agreeing sentiments with respect to the predefined sentiment polarity of w_j, i.e., $\operatorname{sgn}\left(\hat{s}\left(w_j\right)\right)$. Eq. (1) reflects such observations and adjusts the sentiment $s\left(w_j\right)$ accordingly. Figure 5 shows the example functions of $s\left(w_j\right)$ depending on $\Delta\left(w_j\right)$. In Figure 5(a), starting from an initially negative sentiment score, $\hat{s}\left(w_j\right) = -0.6$, as we observe more examples with the opposite sentiment polarity, i.e., increasing $\Delta\left(w_j\right)$, $s\left(w_j\right)$ changes gradually from a negative value to a positive one. The parameter N, e.g., 5 in our case, determines a particular value of $\Delta\left(w_j\right)$ where the original sentiment polarity starts to be reversed. Figure 5(b) shows another example with an initially positive sentiment score. Depending on the size of a text corpus, one can change the parameter values of N and k.

In practice, this measure plays a role of reflecting (1) the context in which the textual post was written as well as (2) the general impression of users associated with a particular facet. That is, in the case of the former, a particular word may have a different sentiment depending on the context. For instance, the word "expensive" can be used with a positive sentiment in a casual conversations, say, in "Wow! you wear an expensive watch!" while it is with a negative sentiment when it comes to most of the product reviews. On the other hand, in the case of the latter, we aim at taking the general positive or negative opinion about a particular facet that takes users into account even for those words initially with no sentiments attached. For example, suppose a previously marked review article about a restaurant is available as "I had to -wait too long on lunch time-." Suppose also that a newly created but yet unmarked review is posted as "You need to get there before 11:30am for lunch." In this example, the word "lunch" was contained in a negative aspect in the first review. Now, in the second one, suppose we need to determine the sentiment on the candidate phrase "need to get there before 11:30am for lunch." In this phrase, there are possibly no words associated with negative sentiment. However, the user-defined sentiment score for the word "lunch" will have a negative value due to the previous marking, which would result in suggesting the candidate phrase as a negative aspect.

Finally, we handle those conjunctions changing the polarity of a sentiment (e.g., "but," "however," and "on the other hand") as follows: If a candidate phrase appears after such a conjunction, we add the opposite sentiment score from the phrase before the conjunction. This sentence-level correction was placed to reflect user's intent more accurately.



(a) $\hat{s}\left(w_j\right) = -0.6$ (b) $\hat{s}\left(w_j\right) = 0.6$

Figure 5: The example graphs of $s\left(w_j\right)$ vs. $\Delta\left(w_j\right)$. We used $k = 0.1$ and $N = 5$.

5. USER EVALUATION

To evaluate our system, we conducted a crowdsourcing-based user study, where we asked a total of 204 participants to answer the question, "What is your favorite fast-food restaurant?" To analyze the impact of the different features of Opinion Marks, participants were assigned randomly with one of three conditions: 1) 77 participants with Machine marking only (***M-only***); 2) 67 participants with Human marking only (***H-only***); and 3) 60 participants with Human + Machine marking (***H+M***). In all the three conditions, the website first provided a brief instruction on how to use the three mark types (**# #**, + +, and - -). The three conditions differ in how our system behaves after a participant initially submits an answer. In *M-only*, once a participant submits an answer, it is marked by the automatic marking suggestion module. This automatically marked answer is considered final, so no additional user interaction is allowed after the initial submission. In *H-only*, if a participant submits an answer without any markings on, the website shows an animated instruction explaining how to use markings. Then, the website returns the original unmarked answer to the participant and gives another chance to put markings and resubmit the marked answer. In *H+M*, the procedure is the same as *H-only* except when a participant submits an answer without any markings, the website returns the answer containing automatically marked phrases.

5.1 Results

User Adoption. Table 1 shows the percentage of participants' submissions with marks in their initial and final submissions, regardless of whether their markings were properly done or not. As expected, few participants adopted Opinion Marks in their initial submissions. About 1 or 2 out of 10 participants used marking in their initial submissions (i.e., 14.29% for *M-only*, 10.45% for *H-only*, and 22.64% for *H+M*), which are not significantly different from each other ($\chi^2(2, N = 204) = 2.299, p = 0.313$). After initial submission, in *H-only*, despite the additional instruction on how

[2]We obtained the word-score list from **`https://github.com/cmaclell/Basic-Tweet-Sentiment-Analyzer`**.

to mark, 26 (38.81%) participants still did not adopt markings. In *H+M*, 41 (68.33%) participants refined their markings after seeing the suggested markings, which shows statistically significant difference ($\chi^2(1, N = 108) = 33.191, p < 0.001$). This is interesting because 41 out of 47 participants, who received answers with default marks from automatic suggestion, did not keep default markings even though it is easier to do so. They rather refined their markings to show their clear intent. This shows that automatic marking suggestions clearly nudge participants to adopt markings.

Table 1: Number of participants submitted marked/unmarked answers.

Initial Submission Final Submission	Marked -	Unmarked Unmarked	Unmarked Marked
M-only	11 (14.29%)	-[a]	66 (85.71%)
H-only	7 (10.45%)	26 (38.81%)	34 (50.75%)
H+M	12 (20.00%)	-[a]	48 (80.00%)[b]

[a] In *M-only* and *H+M*, since each submission is marked by automatic marking suggestion, there is no unmarked cases in their final submissions.
[b] Out of 48 participants, 7 participants kept the machine suggested markings, but the rest of 41 manually refined the machine-suggested markings.

Marking Correctness. To evaluate the correctness of markings, two of the authors codified each submission into seven error types.[3] First, reversed sentiments (RS) indicate that aspect phrases are marked with an opposite sentiment, e.g., "+many people on weekend rush+", which should have been marked with "--". Second, incorrect phrase types (IP) indicate that topic phrases are marked as aspect phrases or vice versa, e.g., "**#delicious biscuit#**". Third, fragmented phrase (FP) indicates that a single markable phrase is cut into multiple phrases with marks, e.g., "+lots of +offers+ and +discounts+ on +festival days+". Fourth, merged phrase (MP) indicates that a marked phrase includes unnecessary words so it should have been shorter or should have been broken into multiple phrases, e.g., "-It has a lot of waiting issue and need to do advance booking so it appears as a con to me-" and "+Fresh ingredients and a lot of choices+". Fifth, unmatched marks (UM) indicate that phrases are marked with different starting and ending marks, e.g., "#Five Guys+". Sixth, missed markable phrase (MM) indicates that the submission has markable phrases that are not marked at all. Seventh, falsely marked phrase (FM) indicates that the submission includes phrases that should not have been marked but were marked.

To test the effects of the three conditions on each error type, we conducted logistic regression analysis with Type III tests to fit the result and computed the odds ratios with 95% confidence intervals. The results show that *H+M* generate less errors in two error types (MM and FP) that *H-only* and *M-only* are likely to generate. Participants in *H+M* generated less errors than those in *H-only* for missing marks on markable phrases (MM, $F(2,201) = 68.12$, $p < 0.001$, Figure 6 (f)); participants in *H+M* generated less errors than those in *M-only* for breaking a single phrase with several marks (FP, $F(2,201) = 10.12$, $p = 0.006$, Figure 6 (g)). Furthermore, participants in *H+M* did not show significantly higher or less errors for the rest of the error types. In contrast, participants in *H-only* made more mistakes than *M-only* in properly enclosing phrases with starting and ending marks (UM, $F(2,201) = 6.87$, $p = 0.032$, Figure 6 (e)) and marking all of markable phrases (MM, $F(2,201) = 68.12$, $p < 0.001$, Figure 6 (f)). On the other hand, participants in *M-only* made more errors than *H-only* in marking non-markable phrases (FM, $F(2,201) = 6.06$, $p = 0.048$, Figure 6 (c)) or breaking a single markable phrase into several phrases (FP, $F(2,201) = 10.12$, $p = 0.006$, Figure 6 (g)) than those in *H-only*. The distinctive error distributions show limitations of *H-* and *M-only*.

The experiment results show that automatic marking suggestion increase the correct adoption of Opinion Marks. We had a concern over implementing automatic suggestions: people may not revise the default marks, thereby leading to generating unsupervised, incorrect phrases. This was wrong–people revised phrases after suggestion, so we could collect more phrases. This iterative loop between machine suggestion and human correction makes positive impacts on the output quality. The result shows that users can "do more and better" with "a slight nudge".

We see interesting implications about human-machine collaboration on the Opinion Marks tasks in Figure 6. Automatic suggestion cannot mark phrases accurately so that they deliver users' intent. It makes mistakes in falsely marking phrases that are not markable (see Figure 6 (g)) and in separating phrases that should be merged to one (see Figure 6 (c)). On the other hand, humans make errors in using the markups accurately (see Figure 6 (e)) and in putting marks on phrases that should be marked (see Figure 6 (f)). Our suggestion, tightly integrating *H+M*, indeed improved some weaknesses on both sides and result in the higher marking rates in the submitted answers.

6. DISCUSSION

As seen in the previous section, we demonstrated that Opinion Marks allows us to collect accurate and compact phrases along with their sentiments by leveraging both machine learning and human computation. In this section, we discuss further applications and limitations of Opinion Marks.

First, the main idea of Opinion Marks, which converts unstructured text into a partially structured form, can potentially boost the performance of various machine learning techniques in text analysis. Previously, the first step in this domain is usually to preprocess unstructured text data using a bag-of-words encoding scheme, which basically uses most of the available keywords in a corpus regardless of their respective importance and noise levels. On the other hand, the phrases collected via Opinion Marks provide a higher quality of data representations with a selective set of meaningful keywords. Furthermore, these keywords are represented at a right level of granularity based on human understanding, which is between a word level (fine-grained) and a document level (coarse-grained). In this sense, our work can help to improve machine learning tasks, such as topic modeling, document summarization, and sentiment analysis.

Second, although we currently applied Opinion Marks to our own proof-of-concept system, **`http://caniask.net`**, it can be easily embedded into broader online writing environments, such as online product reviews, social media, and discussion forums. The user interfaces that we designed for Opinion Marks does not require significant modifications towards its integration to an existing system. Upon integration, Opinion Marks could also be utilized in efficiently marking large-scale data already available on the system as well as newly generated data, e.g., Amazon product reviews. In this manner, Opinion Marks will accelerate a tedious process of reading individual text much faster. Alternatively, one could create a pluggable online writing platform equipped with Opinion Marks. For example, a web-based service available at **`http://disqus.com`** provides users with a discussion thread platform that can be integrated into their own websites. In this manner, the impact of Opinion Marks can quickly reach various real-world domains.

Third, we believe that our technique can be applied to collect accurate phrases from existing large-scale online text (e.g., prod-

[3]To avoid any potential biases, we first removed the information in which conditions each answer was written.

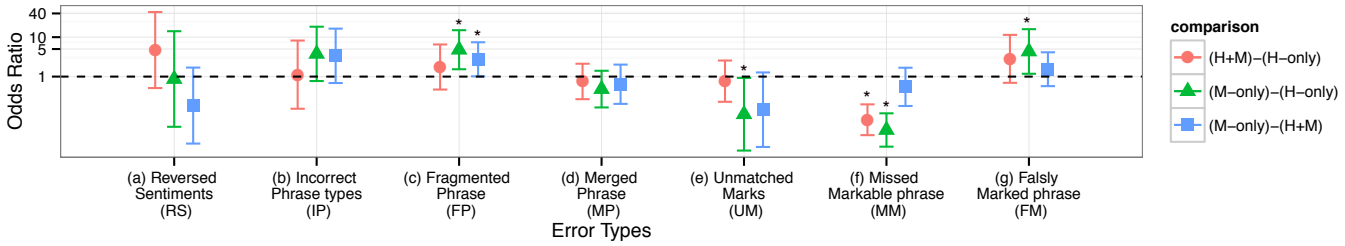


Figure 6: The 95% confidence intervals of odds ratios of seven error types in pairwise comparison between three experimental conditions. The asterisk mark (*) indicates statistically significant odds ratios.

uct reviews) written by other users. One of the major issues with our technique is the potential lack of clear motivation to use our technique. However, even when the adoption by users may not be high enough, major companies maintaining the online text might be willing to spend money on hiring crowdsourced workers or more experienced people to create a human-readable summary out of copious text, and thereby improving tehir services. Furthermore, our "agreement" measures in the summary table could gradually motivate a large number of users by giving them social-psychological incentives. Finally, in addition to these monetary and social incentives, we can also utilize the gamification idea for the phrase extraction process.

7. RELATED WORK

In this section, we review related work from three perspectives: computational approaches, visualization approaches, and human-computing approaches.

7.1 Computational Approaches

Many computational approaches for text analysis generally aim at revealing meaningful insights and summaries out of unstructured text data. In the following, we review two closely related areas to our work: (1) sentiment analysis/opinion mining and (2) key phrase extraction.

Sentiment analysis and opinion mining [17, 22] intend to detect contextual polarity of given text information, e.g., a positive or a negative sentiment. Different methods work at a different level, such as a keyword, a phrase, a sentence, or an entire document [20, 25, 32]. Traditionally, a simple lexicon-based approach, which aggregates word-level sentiment scores, has been widely used [11]. Until recently, numerous methods have also tried to capture subtle connotations in human language, context dependency, and incorrectness [3, 6, 27, 33]. Beyond traditional online review data, sentiment analysis has been actively applied to novel social media data [2, 19]. Nonetheless, sentiment analysis still remains as an active area of research due to its difficulty. In addition, in many approaches, it is not usually a main concern to extract the key phrases that directly support detected sentiments out of an entire text.

On the other hand, various approaches for key phrase mining have also been proposed [9, 35]. In a sense that these key phrases are useful in summarizing documents and revealing high-level topics, it has often been studied together with topic modeling [15, 18, 31]. However, most of these methods rely on the frequent occurrence of a particular phrase in the exact same form, and thus they cannot properly detect various phrases with the same meaning [26].

Generally, fully understanding writers' meaning and intent is still one of the biggest challenges when using fully automated approaches.

7.2 Visualization Approaches

Visualization approaches, often used along with other computational approaches, provide users with a gateway to interactively explore text data. Many developments have been made particularly in context of online reviews. OpinionBlocks provide the overview of snippets from multiple consumer reviews [1]. Oelke et al. [21] presented features and sentiments in a matrix visualization. Review Spotlight shows word clouds of useful adjective-noun word pairs [34]. Similarly, ReCloud also provides word cloud of online reviews [30]. RevMiner and Odin provide mobile interfaces for users to explore reviews based on opinion mining [10, 12]. Utopian uses a scatter plot visualization for topic summary, and it has been applied to online review data to summarize their topics [4]. Termite uses a matrix visualization to show coocurrences of key topic words appearing in documents [5]. These approaches rely upon computational approaches to extract features (e.g., sentiment) to visualize. Therefore, it is necessary to improve the phrase and sentiment extraction tasks for visualizations to be unbiased and useful.

7.3 Human-Computing Approaches

Human-computing approaches aim to delegate part of jobs for machines to human workers so that they can achieve the best outcome [8, 28]. Such human computing approaches have been successfully applied in image tagging [29], image segmentation [24], and text categorization tasks [14], which lead to better performances of diverse machine learning algorithms.

In the context of sentiment analysis and opinion mining, *Opinion Observer* provides an interface for user correction on analyzed sentiment results from product review data [16]. More recently, a web-based sentiment analysis system where a user can run sentiment analysis on his/her own text and make corrections has been proposed [25]. However, as far as our knowledge goes, none of the previous systems have smoothly integrated inline labeling approaches to the writing phrase while preserving a user's semantic context. Such integration is crucial in maximizing user participation and accuracy in the human labeling processes [29].

8. CONCLUSIONS

In this paper, we presented Opinion Marks and its proof-of-concept system, `http://caniask.net`, which enables users to mark their opinions while they write text. To this end, we designed two interaction methods to add such marks with minimal human efforts. Opinion Marks also features automatic marking suggestion based on our carefully designed algorithm so that we can maximize user participation as well as accuracy. Our crowdsourced user study demonstrates that Opinion Marks successfully leveraged the collaborative effects between human users and computer machines for enhancing the output quality and user participation.

Our work has great potential in diverse online writing applica-

tions. Specifically, as our future work, we plan to analyze large-scale document data such as Amazon.com product reviews based on Opinion Marks. In this scenario, we will investigate how to further improve the efficiency of a marking process given numerous text data by using more advanced automatic suggestion algorithms as well as more convenient user interfaces. In addition, instead of just a tabular-style display, we will focus on how to effectively summarize the marked entries and support humans for better understanding of them.

9. ACKNOWLEDGMENTS

This work was supported in part by NSF grant CCF-0808863 and DARPA XDATA grant FA8750- 12-2-0309. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of funding agencies. We thank Nag Varun Chunduru, Sebastian Lee, Zhihua Dong, and Sukwon Lee for their constructive feedback throughout this project.

10. REFERENCES

[1] B. Alper, H. Yang, E. Haber, and E. Kandogan. OpinionBlocks: visualizing consumer reviews. In *Interactive Visual Text Analytics for Decision Making in conjunction with VisWeek 2011*, Providence, RI, 2011.

[2] L. Chen, W. Wang, M. Nagarajan, S. Wang, and A. P. Sheth. Extracting diverse sentiment expressions with target-dependent polarity from twitter. In *Proceedings the 6th International AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 50–57, 2012.

[3] Y. Choi and C. Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 793–801, 2008.

[4] J. Choo, C. Lee, C. K. Reddy, and H. Park. UTOPIAN: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 19(12):1992–2001, 2013.

[5] J. Chuang, C. D. Manning, and J. Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 74âĂŞ–77, New York, NY, USA, 2012. ACM.

[6] X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings the International Conference on Web Search and Data Mining (WSDM)*, pages 231–240, 2008.

[7] M. Efron. Hashtag retrieval in a microblogging environment. In *Proceedings the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 787–788, 2010.

[8] S. Farnham, H. R. Chesley, D. E. McGhee, R. Kawal, and J. Landau. Structured online interactions: improving the decision-making of small discussion groups. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 299–308, 2000.

[9] D. Fetterly, M. Manasse, and M. Najork. Detecting phrase-level duplication on the world wide web. In *Proceedings the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 170–177, 2005.

[10] J. M. Hailpern and B. A. Huberman. Odin: Contextual document opinions on the go. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1525–âĂŞ1534, New York, NY, USA, 2014. ACM.

[11] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–177, 2004.

[12] J. Huang, O. Etzioni, L. Zettlemoyer, K. Clark, and C. Lee. RevMiner: an extractive interface for navigating reviews on a smartphone. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 3–âĂŞ12, New York, NY, USA, 2012. ACM.

[13] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings the 19th International Conference on World Wide Web (WWW)*, pages 591–600, 2010.

[14] A. C. KÃűnig and E. Brill. Reducing the human overhead in text categorization. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 598–603, 2006.

[15] B. Liu, C. W. Chin, and H. T. Ng. Mining topic-specific concepts and definitions on the web. In *Proceedings the 12th International Conference on World Wide Web (WWW)*, pages 251–260, 2003.

[16] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351, 2005.

[17] B. Liu and L. Zhang. A survey of opinion mining and sentiment analysis. In *Mining Text Data*, pages 415–463. Springer, 2012.

[18] Z. Liu, W. Huang, Y. Zheng, and M. Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 366–376, 2010.

[19] Y. Mejova and P. Srinivasan. Crossing media streams with sentiment: Domain adaptation in blogs, reviews and twitter. In *Proceedings the 6th International AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 234–241, 2012.

[20] T. Nakagawa, K. Inui, and S. Kurohashi. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794, 2010.

[21] D. Oelke, M. Hao, C. Rohrdantz, D. A. Keim, U. Dayal, L. E. Haug, and H. Janetzko. Visual opinion analysis of customer feedback data. In *IEEE Symposium on Visual Analytics Science and Technology, 2009. VAST 2009*, pages 187–194. IEEE, 2009.

[22] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[23] C. Rohrdantz, M. C. Hao, U. Dayal, L.-E. Haug, and D. A. Keim. Feature-based visual sentiment analysis of text document streams. *ACM Transaction of Intelligent Systems and Technology.*, 3(2):26:1–26:25, Feb. 2012.

[24] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image

annotation. *International Journal of Computer Vision*, 77(1-3):157–173, May 2008.

[25] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, 2013.

[26] V. Stoyanov, N. Gilbert, C. Cardie, and E. Riloff. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, pages 656–664, 2009.

[27] R. Trivedi and J. Eisenstein. Discourse connectors for latent subjectivity in sentiment analysis. In *Proceedings the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 808–813, 2013.

[28] L. von Ahn. Human computation. In *Proceedings of the 46th Annual Design Automation Conference*, pages 418–419, 2009.

[29] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, 2004.

[30] J. Wang, J. Zhao, S. Guo, C. North, and N. Ramakrishnan. ReCloud: semantics-based word cloud visualization of user reviews. In *Proceedings of the 2014 Graphics Interface Conference*, pages 151âĂŞ–158, Toronto, Ont., Canada, Canada, 2014. Canadian Information Processing Society.

[31] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings the 7th IEEE International Conference on Data Mining (ICDM)*, pages 697–702, 2007.

[32] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354, 2005.

[33] Y. Wu, Q. Zhang, X. Huang, and L. Wu. Phrase dependency parsing for opinion mining. In *Prof. the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1533–1541, 2009.

[34] K. Yatani, M. Novati, A. Trusty, and K. N. Truong. Review spotlight: a user interface for summarizing user-generated reviews using adjective-noun word pairs. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 1541–1550, New York, NY, USA, 2011. ACM.

[35] H. Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–120, 2002.

In Search of User Features for Identifying Different Inspection Behaviors on Recommended Items

Kibeom Lee
Graduate School of Convergence Science and Technology
Seoul National University
Seoul, Korea
kiblee@snu.ac.kr

Sangmin Lee
Graduate School of Convergence Science and Technology
Seoul National University
Seoul, Korea
pizzicato@snu.ac.kr

Kyogu Lee
Graduate School of Convergence Science and Technology
Seoul National University
Seoul, Korea
kglee@snu.ac.kr

ABSTRACT

In recent years, research in recommender systems have began focusing on other elements of recommender systems besides accuracy, such as novelty, diversity, and serendipity. Naturally, research in these areas concentrate on providing novel and relevant recommendations. However, when presented with such recommendations, it is important that users actually inspect the unknown, novel items. Encouraging users to inspect such items can be achieved through the system itself, such as building trust or using previews and explaining recommendations.

However, in this study, we analyze the users, rather than the system, to find features that are good indicators of the users' behaviors towards novel items. In order to achieve this, we carry out a user study and observe the user's interactions with the recommendations. These users are divided into different groups depending on their reactions to recommended items: Explorers and Indifferents. We then search for features in user profiles that can help distinguish the differences between the two groups.

Based on the results of independent samples t-tests, we propose that artist, genre, and tag diversity, in addition to widely-distributed listening behaviors across artists, are features that show significant differences in mean values between Explorers and Indifferents. We believe that these features can be utilized to add another layer of personalization to existing recommenders to adjust the novelty of recommendations according to the target user's group affiliation.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human Factors; H.3.3 [**Information Search and Retrieval**]: Information Filtering

General Terms

Experimentation

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

Keywords

user features, novelty, recommender systems, recommendation interaction, recommendation inspection

1. INTRODUCTION

Recommender systems have been an extremely active field of research, with its importance growing in recent years due to the rapid advancements in technology and massive amounts of data available. Research on recommender systems first emerged in the 1990s, with the introduction of collaborative filtering [16, 18]. Throughout the years, other methods of recommender systems, such as content-based recommenders [9,14,15] and hybrid recommenders [2,3,6,17] were also proposed.

Until recently, the majority of research and development efforts on recommender systems were focused on accuracy: trying to predict the users' ratings on items. Algorithm rankings in competitions such as the Netflix Prize [4] and the KDD Cup were also based on accuracy metrics. In recent years, research in recommender systems that go beyond accuracy have emerged, stemming from findings that user satisfaction and recommender accuracy are not always correlated [12, 25]. This led to research in various aspects of recommender systems, such as increasing the diversity, novelty, and serendipity of the recommended items [7].

The various research on diversity and novelty have an inherent agreement that users will actually examine the provided novel recommendations. Thus, it is accepted that motivating users to inspect recommendations can be attained through various facets of recommender systems, such as trust, transparency, etc. However, in this paper, rather than focusing on the above facets of recommender systems from a technical viewpoint, we study the users themselves based on the interactions with recommended items. By doing so, our goal is to search for user features that can be used to differentiate two types of contrasting users: the type that proactively samples unknown recommended items, and the type that shows no interest in such recommendations.

2. RELATED WORK

Project Phoenix, a study carried out by media company Emap, surveyed 2,200 15-39 year olds about their music listening habits. The people were then divided into four tiers of interest in music: indifferents, casuals, enthusiasts, and savants [8]. Based on these groups, Celma suggested that

each of these groups would require different types of recommendations [5]. Besides Project Phoenix, there were no reported studies on differentiating groups of users according to their behavior or attitudes toward novel recommendation items to our knowledge.

Various recommenders were proposed that aimed to increase diversity and novelty of the recommended items [1,10, 11,21–23]. These studies identified the problems with previous recommender systems as only focusing on the accuracy recommendations. Each paper presented its own method to tackle this problem. However, while it is widely accepted that there is more than accuracy when providing recommendations, it is difficult to find research on what kinds of users inspect such recommendations. In this paper, we attempt to identify users who show interest in novel recommendations and those who are indifferent to such recommendations.

Besides studies on algorithms, there have also been research that found different aspects of recommender systems that were correlated to user satisfaction. Novelty, diversity, serendipity, trust, transparency, and social factors were some of the aspects that influenced user satisfaction of the recommender system [5, 7, 13, 19]. In particular, Swearingen and Sinha explored the design elements of recommender systems that enabled the system to introduce users to novel items and convince them to view them [19]. Based on the results of their user study, they suggested that different recommender systems would be needed to satisfy the needs of different users. Their proposed solution to this was to let the users decide what recommendations they wanted or to explicitly ask the kind of recommendations they desired at the beginning of each session. In this paper, we search for user features that can indicate their attitudes toward novel items, thus removing the need of requesting explicit feedback.

3. USER STUDY

We designed a user study in order to divide users into groups depending on their behavior toward novel items in their recommendations. With our criteria, users would be categorized largely into two extreme groups: (1) those who showed interest in the novel items in their recommendation lists, and (2) those who showed no interest to the recommendations at all.

3.1 Design of User Study

The most important part of the design of the user study was to capture the most natural behaviors of the participants towards recommendations. To achieve this, we refrained from any explicit instructions on the user study and avoided encouraging users in inspecting their recommendations. Instead, on the welcoming page, we declared that this was a user study on recommender systems and that the participants would be provided a list of artists based on their Last.fm profiles, which they were free to explore as they wish. They were also notified that a simple question would be asked in the end.

The user study was largely divided into three stages, as illustrated in Figure 1. Once the participant input their Last.fm ID, we generated 10 recommendations that included artists who had a high probability of being unknown (novel) to the user. Details of generating the recommendation list is provided in the next section. Next, the participants viewed their recommended artists and could click the links to access more information and listen to the artists on their respective Last.fm artist pages. During this exploration stage, we tracked the number of click-throughs of each artist. The participants could end their browsing session by clicking a button, which brought them to the last stage of the user study. Here, they were represented with identical recommendation list and were requested to select all the artists that they were already familiar with. By doing so, we were informed of which recommended artists were actually novel to the user.

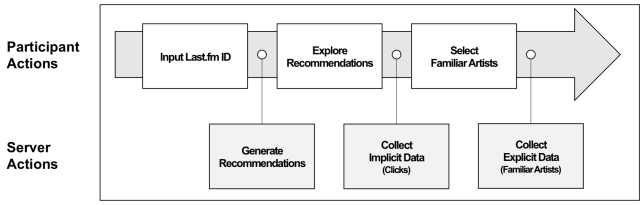


Figure 1: Process of the user study. Users input their Last.fm ID and receive 10 recommendations that include several novel items. While they explore the recommended items (if at all), we record the time spent on the user study webpage and any clicked artists. The only explicit information we require from them is to flag any familiar artists, which is collected after the exploration stage is complete.

3.2 Recommendations with Novel Items

3.2.1 Algorithm

The Myrrix[1] recommender system was used to generate the recommendations, which uses a variant of the ALS-WR algorithm [24]. The parameters used for the ALS-WR algorithm were the default $\lambda = 0.01$ and $\alpha = 40$.

The Myrrix recommender was trained with data gathered from Last.fm, which is discussed in detail below. From the recommendations provided by Myrrix, we took the top seven items as the extremely accurate items and took the 100th, 200th, and 300th items as the potentially novel items. By doing so, we aimed to bring novel items to the participants while keeping them moderately relevant instead of offering random, unknown artists. In total, a list of 10 recommendations were generated for each participant with seven items that were highly probable of being known to the participants and three items that had higher chances of being novel. The order in which the artists appeared were randomized.

We deliberately had the recommendations contain seven extremely accurate items in order to build trust on the recommender system, as indicated by several studies [19]. Thus, we aimed to provide trust in the recommender with the accurate items and observe the behavior towards the remaining novel items.

3.2.2 Data

The data used to train the Myrrix recommender was gathered with the Last.fm API. Listening data of 32,413 Last.fm users were gathered by querying their 50 most listened artists. The profiles of the 32,413 users covered 184,890 unique artists.

To train the Myrrix recommender with the collected Last.fm dataset, the playcount information was converted to ratings with a common rating scale of $[0, n]$ using the func-

[1]The Myrrix project has been discontinued as of December 31, 2013 and is currently part of the Oryx project

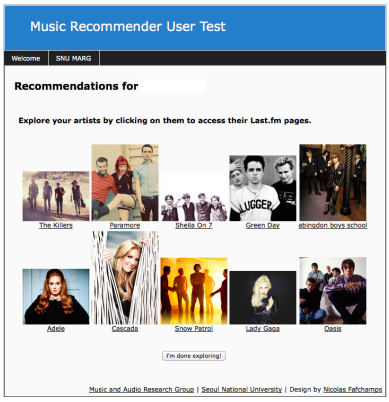


Figure 2: Screenshot of user study (username is erased) showing 10 recommendations for a user. The user is free to explore the artists by clicking on them to go to their respective Last.fm pages where they can read about the artists and listen to their songs.

tion $r(u,i) = n * F(\text{playcount}_{u,i})$, where $\text{playcount}_{u,i}$ is the playcount of user u on item i, and $F(\text{playcount}_{u,i})$ is the cumulative distribution function of $\text{playcount}_{u,i}$ defined by $|\{j \in \boldsymbol{u} | \text{playcount}_{u,j} \leq \text{playcount}_{u,i}\}|/|\boldsymbol{u}|$, using the items in u's profile - $\boldsymbol{u}$, as in [20].

To summarize, the training data for the recommender was based on 32,413 Last.fm user profiles. These profiles were converted to ratings, making up 2.9 million ratings.

4. GROUPING BASED ON SAMPLING BEHAVIOR

Participants for our user study were recruited through various outlets, such as Last.fm message boards, MIR mailing lists, and online communities. The only requirement was that they have a Last.fm ID. A total of 148 participants visited the user study, of which 110 participants actually took part in the user study. Among the 110 participants, we removed users who were not presented with any novel items in their recommendations using the feedback from the last stage of the user study, which left us with 92 participants. Lastly, we filtered out the bottom 10% of participants with the least cumulative playcounts, signifying developing profiles that were not yet 'mature'. In the end, our data was made up of 83 participants. A screenshot of the user study is shown in Figure 2

Based on the Last.fm personal profiles, we present demographics on the participants. Not all users had their profiles public, so the following statistics do not accurately represent all the participants. The average age was 24.94 with 11 women and 55 men (17 unknown) and while the nationalities were diverse, the most dominant nationalities were the U.S. (24) and the U.K. (13).

The goal of our experiment was to find features in user profiles that could help differentiate between users with opposing interests towards recommendations. Thus, we divided the participants into two groups - Explorers and Indifferents - based on their interactions with the recommended items. Explorers were users who showed interest in the novel (unknown) items and viewed additional information by clicking the artist links. In contrast, Indifferents were users who showed absolutely no interest in any of the recommended items.

Using the explicit feedback from the participants, we found that an average of 2.99 novel recommendations (out of 10 recommendations) were given to the 83 participants. The Explorers group was provided an average 3.12 novel recommendations and the Indifferents was given 2.97 recommendations.

Regarding Indifferents, we were aware that the lack of inspection, or interaction of items could be due to many factors. For apathetic behavior, in particular, it could be argued that users with such characteristics could wrongly belong in this group. However, we decided that these characteristics are natural attributes of the user. Thus, we believe that if such factors exist in the study, then they would also exist in the real-world, representing an accurate portrayal of real-world behavior.

5. USER FEATURES

In order to search for features that could work as good identifiers of these groups, we explored various facets of the data and defined several features that we predicted would differentiate Explorers and Indifferents. The search for features was done as a full exploration of features that could be acquired from the available data (music listening history, tags, personal profile, friends list, etc). Throughout the paper, we define the user's profile, denoted as $\boldsymbol{u}$, as the top 50 most listened artists from the user's library.

5.1 Features on Profile Diversity

Intuitively, we predict that users with diverse listening habits will likely be Explorers. Thus, we define features that measure diversity from four different perspectives, namely artists, genres, tags, and online friends.

5.1.1 Artist Diversity

We predict that artist diversity will be a good metric in distinguishing the two groups, as stated in Hypothesis 1.

HYPOTHESIS 1. *Explorers will have profiles with relatively higher artist diversity, while Indifferents will have profiles with relatively lower artist diversity.*

We measured the diversity of artists in a user profile by collecting the top 20 similar artists (obtained through the Last.fm API) for each artist in the profile and finding the ratio of unique artists to total artists [2], which we name Artist Diversity.

More formally, let $A = \{sim_{20}(x) | \forall x \in u\}$ be the multiset of top-20 similar artists of each artist in $\boldsymbol{u}$, and A_d be the set of distinct artists in A. Then, we define Artist Diversity (AD) as,

$$\text{AD} = \frac{|A_d|}{|A|} \quad (1)$$

5.1.2 Genre Diversity

Likewise, we predict that the two groups will have significant differences in genre diversity, as in Hypothesis 2.

HYPOTHESIS 2. *Explorers will have profiles with relatively higher genre diversity, while Indifferents will have profiles with relatively lower genre diversity.*

[2]Formula adapted from `http://anthony.liekens.net/pub/scripts/last.fm/supereclectic.php`.

Because Last.fm lacked genre metadata, we obtained the artists' genres using the Echonest API. The profiles of the participants spanned 922 genres, as Echonest assigns each artist with dozens of weighted genres. Using this data we formulated two varying methods of measuring genre diversity, which we labeled Genre Diversity and Genre-Space Uniformity.

To calculate Genre Diversity, we collect the associated genres with weights above a certain threshold for each artist in the user's profile $\boldsymbol{u}$. Genre Diversity is then given by the ration of unique genres to all genres.

Formally, let $G = \{\text{TopGenres}\,(x, w_{\text{genre}})|\forall x \in \boldsymbol{u}\}$ be the multiset of genres from each artist x in $\boldsymbol{u}$ with weights $\geq w_{\text{genre}}$, and G_d be the set of distinct genres in G. Then, we define Genre Diversity (GD) as

$$\text{GD} = \frac{|G_d|}{|G|} \tag{2}$$

As another way to measure genre diversity, we represent each artist in a user's profile as a vector in the genre space with the genre weight as entries. Thus, a user's profile creates an $M \times N$ matrix, where M is the number of artists in the user's profile and N is the number of genres.

Formally, let H be the set of all genres; $\text{GenreWeight}(\boldsymbol{u}_i, j)$ be the weight of genre j for artist i in $\boldsymbol{u}$; and S be an $M \times N$ matrix where $M = |\boldsymbol{u}|$, $N = |H|$, $S(i,j) = \text{GenreWeight}(\boldsymbol{u}_i, j)$, $Q = \{x|x \in 1 \cdot S, x > 0\}$ and $\mathbf{1}$ is a $1 \times M$ vector of all ones. Then, we define Genre-Space Uniformity (GSU) as,

$$\text{GSU} = \frac{\sum \mathbf{1} \cdot S}{|Q|} \tag{3}$$

5.1.3 Tag Diversity

Similarly, we anticipate that tag diversity will also be an effective method of differentiating Explorers and Indifferents, as we state in Hypothesis 3.

HYPOTHESIS 3. *Explorers will have profiles with relatively higher tag diversity, while Indifferents will have profiles with relatively lower tag diversity.*

Tag data for each artist was gathered using the Last.fm API, which returns tags weighted on a scale between (0...100]. Using these weighted tags, we calculate Tag Diversity the same way we did Genre Diversity.

Stated formally, let $T = \{\text{TopTags}\,(x, w_{\text{tag}})|\forall x \in \boldsymbol{u}\}$ be the multiset of tags from each artist x in $\boldsymbol{u}$ with weights $\geq w_{\text{tag}}$, and T_d be the set of distinct tags in T. Then, we get Tag Diversity (TD) with,

$$\text{TD} = \frac{|T_d|}{|T|} \tag{4}$$

5.1.4 Social Diversity

Regarding social diversity, we anticipate that Explorers will have friends with diverse listening habits while Indifferents will have friends with similar tastes, leading to less diversity overall, as stated in Hypothesis 4.

HYPOTHESIS 4. *Explorers will have social networks mainly comprised of friends with relatively differing musical tastes, while social networks of Indifferents will mainly be comprised of friends with relatively similar musical tastes.*

Social Diversity measures the average dissimilarity of musical taste between a user and his/her social network. This feature makes use of the Tasteometer metric in the Last.fm API, which measures the similarity between two users based on their profiles.

Let $\text{tasteometer}(u, v)$ be the similarity between user u and v, and let u_{friends} be the set of friends of u in Last.fm. Then, we calculate Social Diversity (SD) with,

$$\text{SD} = \frac{\sum \text{tasteometer}\,(u, v)}{min\,(|u_{\text{friends}}|\,, 50)}, \quad \forall v \in u_{\text{friends}} \tag{5}$$

5.2 Features on Listening Behavior

5.2.1 Profile Popularity

HYPOTHESIS 5. *Explorers will have profiles with relatively less popular artists, while Indifferents will have profiles comprised of relatively popular artists.*

We develop two methods of measuring the overall popularity of a user's profile. The first method uses the number of unique listeners of an artist on Last.fm as a quantitative measure of popularity, which we denote as $\text{pop}(x)$. Thus, we calculate Mean Profile Popularity (MPP),

$$\text{MPP} = \frac{\sum \text{pop}\,(x)}{|\boldsymbol{u}|}, \quad \forall x \in \boldsymbol{u} \tag{6}$$

The second method borrows the formula for calculating spectral centroids and applies it to playcounts, which we name Rank Centroid (RC). The formula is,

$$\text{RC} = \frac{\sum \text{pop}_{\text{rnk}}\,(x)\,|\text{plays}\,(u, x)|^2}{\sum |\text{plays}\,(x)|^2}, \quad \forall x \in \boldsymbol{u} \tag{7}$$

where $\text{pop}_{\text{rnk}}(x)$ is the global popularity rank of artist x, and $\text{plays}(u, x)$ is the playcount of artist x by user u.

5.2.2 Playcount Distribution

HYPOTHESIS 6. *Explorers will distribute their music listening to a relatively larger number of artists, while Indifferents will have skewed music listening towards a relatively smaller number of artists.*

We measure the distribution of playcounts in a user's playlist via two methods. The first method is done by sorting the artists in $\boldsymbol{u}$ in descending order by playcount. Using this distribution of playcounts across artists in a user's profile, we calculate Playcount Skewness (PS) by adapting the adjusted Fisher-Pearson standardized moment coefficient,

$$\text{PS} = \frac{n}{(n-1)\,(n-2)} \sum \left(\frac{\text{plays}(u, x) - \overline{\boldsymbol{u}}}{s} \right)^3, \quad \forall x \in \boldsymbol{u} \tag{8}$$

where $n = |\boldsymbol{u}|$, s is the sample standard deviation, and $\overline{\boldsymbol{u}}$ is the mean playcount of $\boldsymbol{u}$.

The second method, similar to RC, is based on calculating spectral spreads. We call this feature Rank Spread (RS) and define it as,

Table 1: Results of the independent samples t-tests performed on Explorers and Indifferents using the proposed features as variables. All tests were done at the 5% significance level. Tests that reject H0 are in bold.

	Mean (Std. Deviation)				
Feature	Explorers	Indifferents	*t*	*df*	*p*
Artist Diversity	0.74 (0.11)	0.69 (0.10)	2.03	61.84	**0.05**
Genre Diversity	0.18 (0.08)	0.20 (0.10)	-0.72	63.99	0.48
Genre-Space Uniformity	1.10 (0.20)	1.20 (0.22)	-2.09	63.70	**0.04**
Tag Diversity	0.46 (0.10)	0.40 (0.09)	2.71	62.47	**0.01**
Social Diversity	0.62 (0.22)	0.58 (0.19)	0.67	56.76	0.51
Mean Profile Popularity	8.73^{*} (3.94^{*})	7.33^{*} (4.78^{*})	1.31	64.33	0.19
Rank Centroid	$8.41^{\dagger}$ ($4.15^{\dagger}$)	$9.82^{\dagger}$ ($9.82^{\dagger}$)	-1.07	58.25	0.29
Playcount Skewness	2.69 (1.22)	3.01 (1.26)	-1.05	64.73	0.30
Rank Spread	$6.72^{\dagger}$ ($2.13^{\dagger}$)	$5.61^{\dagger}$ ($1.65^{\dagger}$)	2.37	58.24	**0.02**

$* : \times 10^5$, $\dagger : \times 10^2$

$$\text{RS} = \sqrt{\frac{\sum \left(\text{pop}_{\text{rnk}}(x) - \text{SC}_{\text{ArtistRank}}\right)^2 |\text{plays}\,(u,x)|^2}{\sum |\text{plays}\,(u,x)|^2}}, \quad \forall x \in \boldsymbol{u} \tag{9}$$

6. RESULTS & DISCUSSION

The distribution of the 83 participants were: 32 in Explorers and 35 in Indifferents. The remaining 16 participants did not fall into either group (i.e. they only inspected recommendations that they were familiar with). The act of inspecting only those artists that the participants were familiar with were not characteristics of Explorers nor Indifferents. Due to their vagueness, these participants were removed and the analysis was done on the two extreme groups.

In order to test the hypotheses, we performed an independent samples t-test on each feature comparing Explorers and Indifferents. The results of the tests are summarized in Table 1.

There were significant differences in mean values for Artist Diversity between Explorers and Indifferents, indicating that Explorers listen to a more diverse range of artists compared to Indifferents and supporting Hypothesis 1.

Genre Diversity was measured with $w_{\text{genre}} = 1.0$ and $w_{\text{genre}} \geq 0.9$ for TopGenres(x, w_{genre}). Results for $w_{\text{genre}} = 1.0$ are shown in Table 1. For $w_{\text{genre}} \geq 0.9$, the t-test also failed to reveal a statistically reliable difference between Explorers ($M = 0.19, SD = 0.07$) and Indifferents ($M = 0.19, SD = 0.08$); $t(64.91) = 0.1, p = 0.92$. On the other hand, mean values of Genre-Space Uniformity showed significant differences between Explorers and Indifferents. We believe that Genre Diversity fails to measure genre diversity accurately because of its misrepresentation of the real-world due to the lack of using genre weights. As can be seen in the formula for Genre Diversity, it does not take into account genre weights in the calculations but simply uses them as a threshold. Thus, all genres are treated equally regardless of weight, resulting in a limited method of expressing various user profiles via genres when artists are affiliated to different genres unequally. Therefore, by using Genre-Space Uniformity, we can support our assumptions in Hypothesis 2.

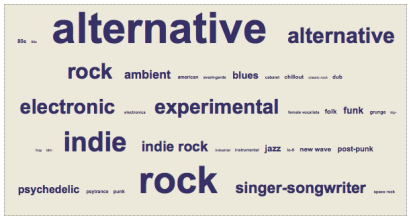
Results of Tag Diversity showed significant differences in mean values for the two groups. This feature was measured with $w = 100, 90, 80, 70, 60, 50$ for TopTags(x, w). Results of the t-tests for different w thresholds are shown in Table 2. In Last.fm, there is only one tag with maximum weight 100 assigned to each artist. Thus, for $w = 100$, each user is represented with tags that are equal in number with the number of artists in his/her profile, making it a conservative measure of diversity. As the threshold for w is lowered, tags that are less and less accurate begin to cloud the metric. The test fails for $w = 50$, where users are associated with an abundant amount of tags but are inaccurate. Such tags create too much noise in the data, making it difficult to extract meaningful interpretations. The t-test results support the idea that Explorers have higher tag diversity and Indifferents have lower tag diversity, as stated in Hypothesis 3. A real example of tag clouds of sample users from the two groups is shown in Figure 3.

Regarding Social Diversity, the two groups did not have any significant differences in mean values. In other words, friend relationships on the social network seem to be formed independent of similarities in musical tastes, contrary to what we predicted. This is interesting as Last.fm is also a social networking service centered on music and musical preferences. Here, we failed to find any supporting data for Hypothesis 4.

On profile popularity, results showed that both Mean Pro-

Table 2: Results of the independent samples t-tests using Tag Diversity as the variable with varying tag weight thresholds (w). All tests were done at the 5% significance level. Tests that reject H0 are in bold.

	Mean (Std. Deviation)				
w	Explorers	Indifferents	*t*	*df*	*p*
100	0.46 (0.10)	0.40 (0.09)	2.71	62.47	**0.01**
90	0.44 (0.09)	0.38 (0.08)	3.01	61.50	**0.00**
80	0.41 (0.09)	0.36 (0.07)	2.76	58.65	**0.01**
70	0.39 (0.09)	0.34 (0.06)	2.62	56.05	**0.01**
60	0.37 (0.09)	0.32 (0.06)	2.55	53.67	**0.01**
50	0.35 (0.09)	0.33 (0.07)	1.47	60.16	0.15



(a) Sample user from the Explorer group.

(b) Sample user from the Indifferents group.

Figure 3: Tag cloud of user's profiles. There is a perceivable difference in the variety of tags between a user from the Explorer group and a user from the Indifferents gruop. Tag cloud images generated from `http://anthony.liekens.net/pub/scripts/last.fm`

file Popularity and Rank Centroid failed to show significant differences in mean between Experts and Indifferents. We had anticipated that Explorers would be listening to long-tail artists and Indifferents would be concentrated towards popular artists. However, according to these results, looking at the popularity of artists is not an effective measure of classifying Explorers and Indifferents. Again, we were not able find supporting data for Hypothesis 5.

Lastly, t-test results on features measuring the listening distribution of users showed significant differences in mean for Rank Spread but failed for Playcount Skewness. We had predicted that Explorers would have relatively less skewed listening habits compared to Indifferents, resembling a balanced consumption of music. However, results do not support this assumption, which could be explained by the nature of how we consume music. Because songs are listened to multiple times, the formation of a power-law distribution in listening patterns may be inevitable when viewing user profiles that represent years of music consumption. Thus, it may be more meaningful to look at the skewness of listening patterns in time scales of a week or month, rather than overall.

Rank Spread, on the contrary, showed significant differences in mean between Explorers and Indifferents. Because the t-test failed for Rank Centroid, we assume that both groups have equal means in Rank Centroid but have significantly different means in Rank Spread. In other words, while both groups listened to similarly popular artists, the distribution of listening by Explorers were spread widely across other artists and the distribution of listening by Indifferents were less spread and more focused on a smaller range of artists, which is in agreement with Hypothesis 6.

7. CONCLUSION

There are numerous studies on increasing novelty and diversity in recommender systems. It is widely accepted that such research on recommenders are necessary as accuracy is simply one of many unknown factors that influence user satisfaction. Likewise, the act of inspecting recommendations, regardless of novelty, may depend on a range of factors, from various elements of the system such as trust, transparency, and user interface. In this paper, we suggest that besides the perspective of the system, there could be human factors that influence interactions with recommendations, which we believe would be embedded in user profiles. Thus, we ventured to find features in user profiles that could differentiate two extreme groups of users: Explorers, who were users that sampled unknown, novel items and Indifferents, who were users that refrained from inspecting any items.

Based on our experiments, the findings indicate that users who inspect unknown, novel items have certain characteristics in their user profiles that are indicators of their behavior.

When dividing the groups into Explorers and Indifferents, the features that distinguish those two groups seem to be Artist Diversity, Genre-Space Uniformity, Tag Diversity, and Rank Spread, which are in support of the Hypotheses 1, 2, 3, and 6.

By using the features proposed in this study, we believe that tailored recommender systems can emerge, in which the system generates different recommendations for users in Explorers and Indifferents groups. For instance, the system could generate more diverse and novel recommendations to users in the Explorers group at the cost of accuracy, while providing more conservative and accurate recommendations to users in the Indifferents group.

8. FUTURE WORK

The user study in this research was designed to be as unobtrusive as possible to the participants, because we wanted to capture their behavior that was the most representative of the real world. To do this, the participants were not explicitly instructed to click on recommendations, but were simply informed that they could through the hyperlinks. However, the implicit data that was collected through the user study may not be representing a user's true intentions. To overcome this problem, the user study could possibly have a post-study survey to record the participants' intentions.

In addition, as with all user studies, a larger sample size would have yielded a more reliable representation of the user population. Regarding features, we attempted to find a wide range of features that targeted different aspects of the user profiles. There were features that we anticipated would work but actually failed. With a larger sample size, there is a possibility that these features could be significant.

Although the presented study did have a rather limited sample size, we believe it does indicate potential features that can be used to predict user behavior towards novel recommended items. Nonetheless, it would be valuable to investigate whether it is indeed the case that user satisfaction can be improved by taking the user's propensity to explore into account. We believe that, with more robust studies regarding interactions on recommendations, this can lead to recommender systems that dynamically adjust its parameters to add another level of personalization for the user. This extra layer of personalization would decide, perhaps, the degree of novelty and diversity in the final recommendations. Such a system would result in a customized recommender for each user, which contrasts to existing recommender systems that use one-size-fits-all personalization algorithms to generate recommendations.

9. ACKNOWLEDGMENTS

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (490-20130014).

10. REFERENCES

[1] G. Adomavicius and Y. Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *Knowledge and Data Engineering, IEEE Transactions on*, 24(5):896–911, May 2012.

[2] M. Balabanović and Y. Shoham. Fab: Content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, Mar. 1997.

[3] C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *In Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 714–720. AAAI Press, 1998.

[4] J. Bennett, S. Lanning, and N. Netflix. The netflix prize. In *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[5] O. Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[6] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR workshop on recommender systems*, volume 60. Citeseer, 1999.

[7] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004.

[8] D. Jennings. *Net, blogs and rock'n'roll: how digital discovery works and what it means for consumers, creators and culture*. Nicholas Brealey Publishing, 2007.

[9] K. Lang. Newsweeder: Learning to filter netnews. In *in Proceedings of the 12th International Machine Learning Conference (ML95*, 1995.

[10] K. Lee and K. Lee. My head is your tail: Applying link analysis on long-tailed music listening behavior for music recommendation. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 213–220, New York, NY, USA, 2011. ACM.

[11] K. Lee and K. Lee. Using dynamically promoted experts for music recommendation. *Multimedia, IEEE Transactions on*, 16(5):1–10, 2014.

[12] S. M. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S. K. Lam, A. M. Rashid, J. A. Konstan, and J. Riedl. On the recommending of citations for research papers. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, CSCW '02, pages 116–125, New York, NY, USA, 2002. ACM.

[13] S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, pages 1097–1101, New York, NY, USA, 2006. ACM.

[14] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, DL '00, pages 195–204, New York, NY, USA, 2000. ACM.

[15] M. Pazzani and D. Billsus. Learning and revising user profiles: The identification ofinteresting web sites. *Mach. Learn.*, 27(3):313–331, June 1997.

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM.

[17] J. Salter and N. Antonopoulos. Cinemascreen recommender agent: Combining collaborative and content-based filtering. *IEEE Intelligent Systems*, 21(1):35–41, Jan. 2006.

[18] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating “word of mouth”. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[19] K. Swearingen and R. Sinha. Beyond algorithms: An hci perspective on recommender systems. In *ACM SIGIR. Workshop on Recommender Systems*, volume Vol. 13, Numbers 5-6, pages 393–408, 2001.

[20] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, pages 109–116, New York, NY, USA, 2011. ACM.

[21] S. Vargas and P. Castells. Exploiting the diversity of user preferences for recommendation. In *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*, OAIR '13, pages 129–136, Paris, France, France, 2013. Le Centre De Hautes Etudes Internationales D'Informatique Documentaire.

[22] M. Zhang and N. Hurley. Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 123–130, New York, NY, USA, 2008. ACM.

[23] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515, 2010.

[24] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management*, AAIM '08, pages 337–348, Berlin, Heidelberg, 2008. Springer-Verlag.

[25] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, pages 22–32, New York, NY, USA, 2005. ACM.

Empirical Comparison of Active Learning Strategies for Handling Temporal Drift

Mohit Kumar
Flipkart
k.mohit@flipkart.com

Mohak Shah
Robert Bosch LLC
Mohak.Shah@us.bosch.com

Rayid Ghani
University of Chicago
rayid@uchicago.edu

Zubin Abraham
Robert Bosch LLC
Zubin.Abraham@us.bosch.com

ABSTRACT

Active learning strategies often assume that the target concept will remain stationary over time. However, in many real world systems, it is not uncommon for the target concept and distribution properties of the generated data to change over time. This paper presents an empirical study that evaluates the effectiveness of using active learning strategies to train statistical models in the presence of various temporal-drift scenarios. The study also evaluates the benefit of incorporating popular approaches to address temporal drift on the various active learning strategies. The performance of the best performing active learning strategies, were found to be at least comparable, if not significantly better than random sampling strategy across the various types of temporal drifts in 99% of the scenarios tested. In approximately 50% of those instances, active learning strategies were significantly better than random sampling. However, the further away the temporal drift, less is the advantage of using active learning strategies over random sampling. It is shown that uncertainty-based sampling often had the best performance among the various active learning strategies.

1. INTRODUCTION

Active learning algorithms attempt to learn an accurate statistical model by selecting the most informative data to be used for training. The approach is primarily motivated by the fact that in certain domains, labeling of data needed for training a model is expensive. Similar to most other passive learning strategies (where all training examples are labeled), active learning strategies assume that the target concept remains stationary over time [17]. However, many real world data mining applications are deployed in settings that are meant to run for extended periods of time, during which the target concept and data distributions may change. Given that statistical models (such as classifiers that assume the data to be stationary), are known to show reduced accuracy in such *temporal drift* scenarios, there is a need to explore the impact of temporal drift on the active learning strategies used to build these statistical models. Fraud detection, intrusion detection, medical diagnosis, information filtering, and video surveillance are examples of applications that would benefit from this study, given that their labeled examples are expensive to generate and since their domain is prone to temporal drift.

Temporal drift has been categorized into three main types: 'shifting class distribution' (SCD), 'shifting subclass distribution' (SSD) and 'fickle concept drift' (FCD) [7]. SCD is defined to occur when the relative proportion of cases in the different classes may change over time, but the samples within a given class are i.i.d stationary. SSD is defined when a class category may be comprised of a union of (potentially undiscovered) subclasses or themes, and the class distribution of these subclasses may shift over time. FCD refers to the scenario when individual samples may take on different ground truth labels at different times. In this paper, we study two types of drifts, SSD and FCD. We do not study SCD in context of active learning as it is difficult to interpret the results of active learning for SCD, as the difference in performance may be attributed to the underlying change in the class distribution and analyzing the contributions of different active learning strategies may be difficult.

Concept Drift [14] is one form of temporal drift that has been well studied. Concept drift typically refers to the change in the target concept that needs to be learnt over time. There has been work in active learning on streaming data with concept drift [19] and without concept drift [3]. The results from [19] show that random sampling performs better than the proposed active strategies and the authors recommend randomization of active sampling strategies. However, the key difference between streaming data and our focus, is that in the streaming data setup, instances are streaming in to the system and a decision needs to be made right away whether to ask for a label or not. The incoming unlabeled data cannot be stored and queried later. This scenario happens in certain real-world problems (e.g., web search) but is rare in enterprise problems. In most enterprise interactive data mining systems, data needs to be stored anyway for other purposes (e.g., auditing), and the constraint of making a labeling decision instantaneously is not present. Also, in these problems, the domain experts labeling the data are the expensive components of the process and data storage costs often pale in comparison. For these practical reasons, we consider a setting where the unlabeled

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

pool gets augmented with new data coming in, which is different from the two extreme settings of fixed unlabeled pool and completely stream-based setup with no memory. Chu et al. also mention that a periodically changing unlabeled pool is a more realistic scenario than the two extremes of static unlabeled pool and online streaming data [3]. There are multiple settings for the evolving unlabeled pool. *Cumulative streaming pool* setting is when new unlabeled examples keep coming in and is added to the streaming pool, thus increasing the unlabeled pool available to the learner. *Recent streaming pool* setting is where only the most recent unlabeled examples are available to the learner. In the current work, we only experiment with the *recent streaming pool* setting and leave the *cumulative streaming pool* setting for future work. This corresponds to the *Daily Classification Task* setup recommended by Forman for studying concept drift [7].

In addition to dealing with periodically changing unlabeled pool, it's also not clear whether traditional instance selection strategies (namely uncertainty and density based) still perform well and help adapt the system in the presence of temporal drift. In learning from data streams with concept drift, the popular approach has been to learn classifiers over different time periods and combine them in weighted ensembles [14, 16, 20]. However, the effectiveness of traditional instance selection strategies in the periodically changing unlabeled pool setup is not well understood and hence, explored in this study.

Zliobaite exhaustively reviews learning approaches under temporal drift [21], including learner adaptivity approaches such as adaptive base learners [10], learners with adaptive parametrization [13], adaptive training set formations and fusion rules of the ensembles [18] that are relevant to the current work. Hoens et al. focus on learning with streaming data where there is both concept drift and class imbalance [9]. The authors highlight that this is an under-researched area and applies to many real-world problems. We take special note of this in our work and specifically address problem settings where there is significant class imbalance and show empirical comparison of approaches with different levels of imbalance.

This paper evaluates existing active learning techniques under various temporal drift scenarios to assess if it's worth the additional effort to implement intelligent sample selection strategies over using simple random-sampling, when obtaining labels for training is expensive and the domain is susceptible to temporal/concept drift. The setup of the modeling choices for handling temporal drift consists of three components: 1) the type of model used - ensemble or single, 2) instance or model weighting scheme, within the different types of models/ensembles and 3) the type and amount of concept drift. The setup also helps answer the additional questions - which if any, sample selection techniques is appropriate for a given type of temporal drift? Does the choice of the best performing strategy depend on the evaluation metric chosen? This paper also explores the impact of adapting techniques developed in the temporal drift literature to active learning strategies.

2. FRAMEWORK FOR EMPIRICAL COMPARISON

We present a framework that allows researchers and practitioners to compare the performance of various active learning techniques under temporal drift in a broad range of real-world problems. Specifically, we focus on problem settings where a classifier periodically provides the experts with a ranked list of cases to review and verify. The general setting is analogous to the *Daily Classification Task* introduced by [7] for studying concept drift. Time is discretized into periods (e.g., days) and of all the new data that comes in during that period, a subset of it is labeled based on the active sampling strategy. For example, the number of audited cases of health insurance claims [8], is close to 2% of all new claims that come in a day.

The analysis in this paper is structured based on the following five parameters that can be determined a priori by a domain expert. The type of drift, the amount of drift, the target class distribution, the evaluation metric of interest and the cost of labeled data. Distinct combinations of the five parameters, results in 144 different problem settings on two real-world problems, 'information filtering' and 'intrusion detection'. For these 144 problem settings, a study of the performance of several algorithms combining active learning strategies with temporal drift motivated strategies is carried out. The active learning strategies evaluated include 'certainty sampling', 'uncertainty sampling', 'density-based sampling', and 'sparsity-based sampling'. The various learner adaptation strategies for temporal drift evaluated include 'single model with instance-level weighting', and 'weighted ensemble models'. The three variants of weighting schemes evaluated are uniform, linear, and exponential.

2.1 Domain characterization

2.1.1 Type and amount of drift.

We experiment with two types of drift scenarios, Fickle Concept Drift (FCD) and Shifting Subclass Drift(SSD) [7]. FCD is defined where an individual case may have different class labels at different times. For example, in information filtering system the user's preference for relevant news articles may change over time. This kind of drift can be characterized by rate of change in user's preference over time. Thus the amount of drift is parameterized by the probability of switching from one class of interest to another (randomly selected) for the next time period. Even though it can be argued that the user interest may not switch randomly and there may be a semantic pattern to it, we chose to use random switching to be more general and not to introduce an additional bias factor of semantic pattern. We experiment with drift probabilities of 20%, 50% and 100% (labeled as CD0.2, CD0.5 and CD1.0 respectively in figures) as well as the 'no drift' scenario labeled CD0.0.

SSD happens when the positive or negative class comprises of a union of subclasses, and the distribution of these subclasses shift over time. For instance, in network intrusion detection, certain types of intrusions may show up over time as was described in the KDD Cup'1999 dataset [12]. Consequently, while the feature distribution given a particular subclass may be stationary, the feature distribution of the super-class varies over time, because its mixture of subclasses varies. We parameterize the amount of drift by the frequency of the appearance of new subclasses and the disappearance of old ones. We experiment with two drift amounts: drift occurring every 2nd iteration (labeled as Drift=Low), and drift occurring every iteration (labeled as

Drift=High).

2.1.2 Target class distribution.

Most large-scale enterprise data mining problems exhibit class skewness with the level of skewness varying across domains. We experiment with skewness of 10% and 20% for the 'information filtering' task and 1% and 2% for the 'network intrusion detection' task. Although the natural distribution of intrusion cases is very high for the 'KDD Cup network intrusion' dataset, the typical percentage of intrusion cases is expected to be 1 to 2%, which is widely used in studies employing this dataset [5].

2.1.3 Evaluation metric of interest.

Another important characteristic of real-world problems is the performance metric of interest. The choice of evaluation metric is dependent on the domain and the operating range of interest in the domain. We chose the following metrics to cover a broad range of domains: Area Under ROC (AUC) curve, Precision@1st percentile and Precision@10th percentile. The AUC metric is correlated with the ranking accuracy of examples through the entire range [4] and relevant for applications where the performance over all the examples matters. The precision@Kth percentile, is a more focused metric that helps distinguish the performance on the 'top-k' scored cases, making it more relevant for skewed classification problems.

2.1.4 Cost of labeled data.

The number of cases/samples to label is an important design choice, which is also affected by factors such as the budget for labeling. We experiment with representative batch sizes for labeling in a time period with 10 queries and 100 queries, which corresponds to roughly the number of positive examples expected in new unlabeled batches for the two datasets.

2.2 Learning Strategies

We use Support Vector Machine (SVM) as the base classifier and employ various learning strategies as described below.

2.2.1 Active Learning (Sample Selection) choices.

We experiment with four active sampling strategies and compare it to the baseline strategy of random sampling. These include the classical settings of uncertainty and density based sampling and variants of those settings that have been found useful in skewed classification settings [6]. The variant of uncertainty sampling is certainty sampling, where we sample the examples that the classifier is confident on. For linear SVM classifiers, this is basically the distance from the SVM hyperplane as represented by the SVM score. We sample equally from the most confident positive examples, as well as negative examples, to come up with a balanced training dataset. The variant of density sampling include density outlier sampling and sparsity sampling, where we select the examples that are most sparse (least dense). We also evaluate the passive learning setup, where all the data are assumed to be labeled.

2.2.2 Learner adaptation strategies based on historical data.

When building classification models from labeled data spanning more than one period, there are multiple ways to use the historical labeled data for the learner to adapt [21]. We focus on two popular strategies: The first approach builds separate temporal models from each time window by using only the labeled data from that window and then combine those models using ensemble techniques; and second approach builds a *single model* combining all the data (from all time periods) with instance weighting. The *ensemble models* approach has been popularly used to handle concept drift in recent years [14], where a classifier is learnt for each time period and then combined in a (weighted) ensemble. However, a drawback of ensemble based methods is that they need enough labeled data for each time period to build a reasonably good model. In cases where there is not enough labeled data available for each time period, the ensemble based approach may not be ideal. The *single model* approach makes the model more robust (in the presence of limited training data), but less adaptive to temporal drift. One way to compensate for that is weighting the instances differently, based on the time period they belong to.

We experiment with three weighting schemes, for both historical models (in an ensemble's case) and examples (in a single model case): exponential, linear and uniform. Exponential weighting scheme gives exponentially decreasing weight to history, linear weighting scheme gives linearly decreasing weight to the history whereas uniform gives equal weight to history.

3. DATA GENERATION

In order to generalize our results and conclusions beyond the data sets we initially used, we use those data sets to then generate variations that span the spectrum in terms of the parameters mentioned earlier. All the datasets along with the relevant parameter details will be made publicly available shortly. For the experiments, we report the results averaged over 10 random splits.

3.1 Fickle Concept Drift

We create FCD datasets based on the '20-Newsgroup' and 'Reuters-RCV1' datasets [15, 2]. For '20-newsgroup', there are 18,774 documents corresponding to 20 news categories after pre-processing and data clean-up. The Reuters-RCV1 dataset is preprocessed as described by [1], where the label hierarchy is reorganized by mapping the data set to the second level of the topic hierarchy. The documents that have labels of the third or fourth level are mapped to their parent category of the second level. The documents that only have labels of the first level are not mapped onto any category. Further, the multi-labeled instances are removed. Out of the resulting 53 second level topics, we select the top 20 most frequent topics and sample 1000 documents for each topic.

For creating datasets with fickle concept drift, for each time period we sample 50 cases each for the 20 categories for both datasets, resulting in 1000 documents per time period. This gives us 18 time iterations for 20-newsgroup data and 19 time iterations for the RCV1 dataset. We experiment with positive class percentage as 10% (2 out of 20 categories) and 20% (4 out of 20 categories). We test with 0%, 20%, 50% and 100% drift probability over each iteration. Figure 1 shows a sample iteration where the numeric ids correspond to newsgroup categories. For instance, category id '1' corresponds to 'alt.atheism'.

3.2 Shifting Subclass Distribution

	Time Period																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Positive Category A	16	16	16	16	8	8	8	20	20	20	20	12	12	12	12	12	12	12
Positive Category B	17	17	17	17	17	17	17	17	17	17	17	3	3	6	14	3	3	3

Figure 1: A sample iteration of 20-newsgroup data with 10% positive class percentage(2 categories) and 20% drift probability, indicating the positive category id for each time period

Intrusion Name	Time Period																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
warezclient	0	0	0	0	0	0	0	13	12	11	11	9	7	7	13	13	13	13

Figure 2: A sample of the number of cases of subclass 'warezclient' included in 18 time period batches for the KDD cup network intrusion dataset

We derive two SSD datasets using '20-Newsgroup' and 'KDD Cup network intrusion detection' [12] datasets. Network intrusion dataset is a natural choice for this type of drift, as different types of intrusions occur at different times. We ignore the training/test split as suggested by the original dataset and instead resample the data according to our strategy to create the temporal dataset.

For each time period, only a subset of the positive classes are present in a batch. We design the sampling scheme such that the first time iteration has only a few subclasses and then new subclasses get added over time, while some existing ones are removed. The exact sampling schemes are not included in the paper in the interest of space; however the sampling scheme will be made publicly available.

Figure 2 shows the number of 'warezclient' intrusion cases included in the batches across 18 time periods for the network intrusion data for *high drift*. There are 40 intrusion subclasses in the dataset, however, we use the 25 most frequent ones. The negative class is predetermined for the network intrusion dataset (subclass: 'normal') and rest of the 24 subclasses are positive (intrusions). We create batches of 8000 datapoints for each time period with positive class varied between 1% and 2%. Sampling for 20-newsgroup is similar, where we have 1000 cases in each time period with positive class varied between 10% and 20%. We arbitrarily select the 'talk' newsgroup category as the positive class with talk.politics.guns, talk.politics.mideast, talk.politics.misc and talk.religion.misc subclasses.

4. RESULTS

The results for the performance metric(s) (AUC, Precision@10 and Precision@1) are computed at each time period, and the average is calculated over all time periods analogous to that of [7]. This performance is averaged over 10 randomized trials with different data samples to come up with a summary evaluation of each learning strategy choice. Namely, the choice of active sampling strategy, the type of model (single vs ensemble), the weighting scheme (for historical instances or models) for various drifts (FCD/SSD with varying amount of drift) and the domain scenarios (class skewness, cost of labeled data). Thus we get a ranking of 30 learning choices for 144 drift/data/performance metric scenarios. While we could choose to evaluate and report results on other measures or measure with finer granularity over Precision@k, the intent in this paper is to cover a reasonable representative range of parameters to observe different trends.

FCD				CD0.0	CD0.2	CD0.5	CD1.0
20news	Precision@1	pos=10%	Q10	1*	1*	0	0
			Q100	1*	1*	0	0
		pos=20%	Q10	1*	1*	0	0
			Q100	1*	1*	1*	0
	Precision@10	pos=10%	Q10	1*	0	0	1
			Q100	1*	1*	1	0
		pos=20%	Q10	1*	0	0	0
			Q100	1*	1	1	0
	AUC	pos=10%	Q10	-1	0	0	1
			Q100	0	0	1	0
		pos=20%	Q10	0	0	0	0
			Q100	1	1	1	0
RCV1	Precision@1	pos=10%	Q10	1*	0	1*	0
			Q100	1	0	0	0
		pos=20%	Q10	1	0	0	0
			Q100	1	0	0	0
	Precision@10	pos=10%	Q10	0	0	1*	0
			Q100	1	0	0	1*
		pos=20%	Q10	1	0	0	0
			Q100	1	0	0	0
	AUC	pos=10%	Q10	0	0	0	0
			Q100	1	0	0	0
		pos=20%	Q10	0	0	0	0
			Q100	1	0	0	1

Figure 3: Performance comparison of the best performing active learning strategy and random sampling. 1 (green colored cells) indicate that best active learning strategy is statistically better than the best random strategy; 0 (orange colored cell) indicates that there is no statistical difference between best active learning strategy and best random strategy and -1 (red colored cell) indicates that the best random strategy is statistically better than the best active learning strategy. $*$ indicates that the performance is 10% better for information filtering task.'pos' is the skew percentage of the positive class and 'Q' the number of queries labeled in a time period.

4.1 Intelligent vs. Random Sampling

Earlier research has shown that random sampling can often outperform active learning strategies under temporal drift [19] when restisted to streaming data. However, unlike the previous study that was restricted to streaming data, the experiments in this study consider the more commonly encountered setting where the unlabeled pool gets augmented with new data coming in.

Figures 3 and 4 show the statistical significance determination results for FCD and SSD under such a scenario. Barring one instance, active learning strategy was comparable or significantly better than random sampling strategy across concept drifts types. Active learning strategy being significantly better almost 50% of the time. We compared the performance by first undertaking a two-way ANOVA omnibus test followed by Bonferroni post-hoc test with 0.05 significance level(p) [11] using the function multcompare in Matlab.

SSD				Drift=Low	Drift=High
20news	Precision@1	pos=10%	Q10	1*	1*
			Q100	1*	1*
		pos=20%	Q10	1*	1*
			Q100	1*	1*
	Precision@10	pos=10%	Q10	1*	1*
			Q100	1*	1*
		pos=20%	Q10	1*	1*
			Q100	1*	1*
	AUC	pos=10%	Q10	1	1
			Q100	1	1
		pos=20%	Q10	1	1
			Q100	1	1
KDD	Precision@1	pos=01%	Q10	1*	0
			Q100	1*	1*
		pos=02%	Q10	1*	1*
			Q100	1*	1*
	Precision@10	pos=01%	Q10	0	0
			Q100	0	0
		pos=02%	Q10	0	0
			Q100	0	0
	AUC	pos=01%	Q10	0	1*
			Q100	0	0
		pos=02%	Q10	0	0
			Q100	0	0

Figure 4: Performance comparison of the best performing active learning strategy and random sampling. 1 (green colored cells) indicate that best active learning strategy is statistically better than the best random strategy; 0 (orange colored cell) indicates that there is no statistical difference between best active learning strategy and best random strategy and -1 (red colored cell) indicates that the best random strategy is statistically better than the best active learning strategy. $*$ indicates that the performance is 10% better for information filtering task (20 newsgroup) and 1% for network intrusion detection task (KDD Cup).'pos' is the skew percentage of the positive class and 'Q' the number of queries labeled in a time period.

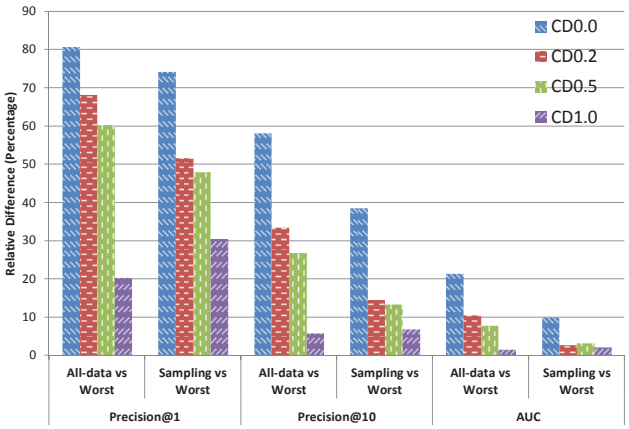


Figure 5: Relative difference between passive learning (using all the data) and best active sampling with the worst performance for the sampling techniques indicating the spread of performance. The positive class percentage is 20% and the number of queries labeled per iteration is 100. Labels CD0.0, CD0.2, CD0.5 and CD1.0 correspond to drift scenarios with probabilities:0, 0.2, 0.5 and 1 respectively

Effect of type and magnitude of drift, on active sampling: Active sampling is the preffered choice over random sampling, in the presence of both high and low magnitude of SSD, as seen in Figure 4. For FCD, which is considered to be a more difficult drift situation [7], the advantage of using active learning strategies wanes as temporal drift increases. The results do however vary across datasets where we observe greater performance difference for the '20-newsgroup' dataset compared to the 'KDD cup' and 'RCV1' dataset. This is intuitively explained by the fact that the subclasses are more closely related to each other in 20-newsgroup (same higher level category 'talk') than the 'KDD cup' dataset, making historical labeled data more useful for '20-newsgroup' than for 'KDD cup'. We conjecture this is because the categories in '20-newsgroup' datasets are more closely related to each other than for RCV1, making historical labeled data more useful for '20-newsgroup' than 'RCV1'. For SSD, we observe that active learning is very useful under different magnitude of drifts for 20-newsgroup whereas only for Precision@1 for KDD cup dataset.

Do the relative performances vary based on the evaluation metric? In general, we observe that performance gain is more pronounced for focused metrics such as Precision@1, whereas for coarser metrics such as AUC, the performance is less variable as observed in Figures 3 and 4. This is observed across various active learning strategies. If the domain of interest has a narrow operating range, such as many real-world problems with class skewness, the difference in performance of active sampling techniques with random sampling is more noticeable.

The relative performance of the worst and best performing sampling strategies gives an indication of the spread of performance and how sensitive the performance is to the choice of sampling strategy. We also compare the best and worst performing sampling choice with the passive learning setup (all the data are labeled and available for training), which both gives us an upper bound on performance and

FCD			CD0.0	CD0.2	CD0.5	CD1.0
20news	Precision@1	pos=10%	0	0	-1	0
		pos=20%	0	0	0	0
	Precision@10	pos=10%	-1	-1	-1	0
		pos=20%	-1	-1	-1	0
	AUC	pos=10%	-1	-1	-1	0
		pos=20%	-1	-1	-1	0
RCV1	Precision@1	pos=10%	-1	-1	-1	0
		pos=20%	-1	-1	-1	0
	Precision@10	pos=10%	-1	-1	-1	1
		pos=20%	-1	-1	-1	1
	AUC	pos=10%	-1	-1	-1	1
		pos=20%	-1	-1	-1	1

Figure 6: Statistical significance comparison between models with 10 actively sampled examples versus 100 randomly selected samples for FCD.

SSD			Drift=Low	Drift=High
20news	Precision@1	pos=10%	0	0
		pos=20%	0	0
	Precision@10	pos=10%	-1	-1
		pos=20%	-1	-1
	AUC	pos=10%	-1	-1
		pos=20%	-1	-1
KDD	Precision@1	pos=01%	-1	-1
		pos=02%	-1	-1
	Precision@10	pos=01%	-1	0
		pos=02%	-1	-1
	AUC	pos=01%	-1	0
		pos=02%	-1	-1

Figure 7: Statistical significance comparison between models with 10 actively sampled examples versus 100 randomly selected samples for SSD.

also gives an indication of the scope of improvement for the different sampling choices and metrics. Figure 5 shows the relative performance difference in percentage between the passive learning (labeled *all-data* in the figure) and the best sampling method (including random) as well as the worst sampling choice for FCD, for the '20-newsgroup' dataset. The major pattern observed is the difference between the best and the worst sampling strategy is large for Precision@1 and reduces progressively for relatively less focussed metrics, such as Precision@10 and AUC.

For all domain settings, the scope for improvement using any intelligent sampling strategy is smallest for Precision@1 and increases for Precision@10 and AUC. However an interesting observation is that when the drift amount is highest (CD1.0) i.e., when in each iteration the positive class is completely changed, the performance of best sampling strategy is better than using *all-data* (comparing the CD1.0 observation across 'All-data vs Worst' and 'Sampling vs Worst' columns). This is probably because for *all-data*, the history is not quite relevant in learning the new class definition. This shows that the history is not useful when drift is extremely high and it is better to use samples of newly obtained data and minimize the use of historical data in learning.

Are the patterns different for different class skewness? There is no significant pattern observable with the different class skewness for comparable data setups for FCD or SSD from Figures 3 and 4.

Are the patterns different for different number of queries per iteration?: There is no significant pattern observable other than that occasionally, the improvement of using active sampling over random sampling was more pronounced with more queries (100) when compared to less number of queries (10) for FCD.

4.2 Practical Considerations

For practical implementation, the actual performance gain achieved is very important in order to justify the value (and added cost of system complexity) of doing active learning. Note that this is not necessarily the same as obtaining a significant difference in a statistical sense. The threshold for the justification of effort may vary across applications. For example, increasing the Precision@1 by 5% can be very significant for applications such as credit card fraud detection, while possibly not be as valuable for information filtering. We choose a threshold of 10% for the 'information filtering' tasks (FCD: 20-newsgroup, RCV1 and SSD: 20-newsgroup) whereas 1% for 'network intrusion' task (SSD: KDD cup) and highlight the results with a '*' in Figures 3 and 4. A cell has a '*', if the difference between the best active sampling strategy and the best random strategy is more than the mentioned threshold.

Is the performance difference significant and worth the cost of implementation? The more focused the evaluation metric, the more significant is the performance difference observed (Figures 3 and 4).

Choosing between labeling more examples randomly or using less labeled examples that are chosen intelligently. We compare the scenario where we label 10 queries using active sampling to the scenario where an order of magnitude more (100 queries) are randomly labeled. Figures 6 and 7 show the results of the statistical significance comparison for both FCD and SSD. We find that labeling more data randomly, almost always gives better performance than intelligent sampling, if the number of samples is one order of magnitude different. One practical implication of this observation is that if the cost of setting up intelligent sampling is high, it may be worth spending the same resources on labeling more randomly sampled data instead. This may in general be true for non-drift situations as well, and may be correlated with sample complexity measures [17], typically used to estimate sample complexity to reach passive learning performance.

4.3 Detailed Results

Figures 8 and 9 show the heatmap of the difference in performance for each active learning strategy relative to the respective best performing choice. The heatmap has separate images for the three different metrics, as the scale

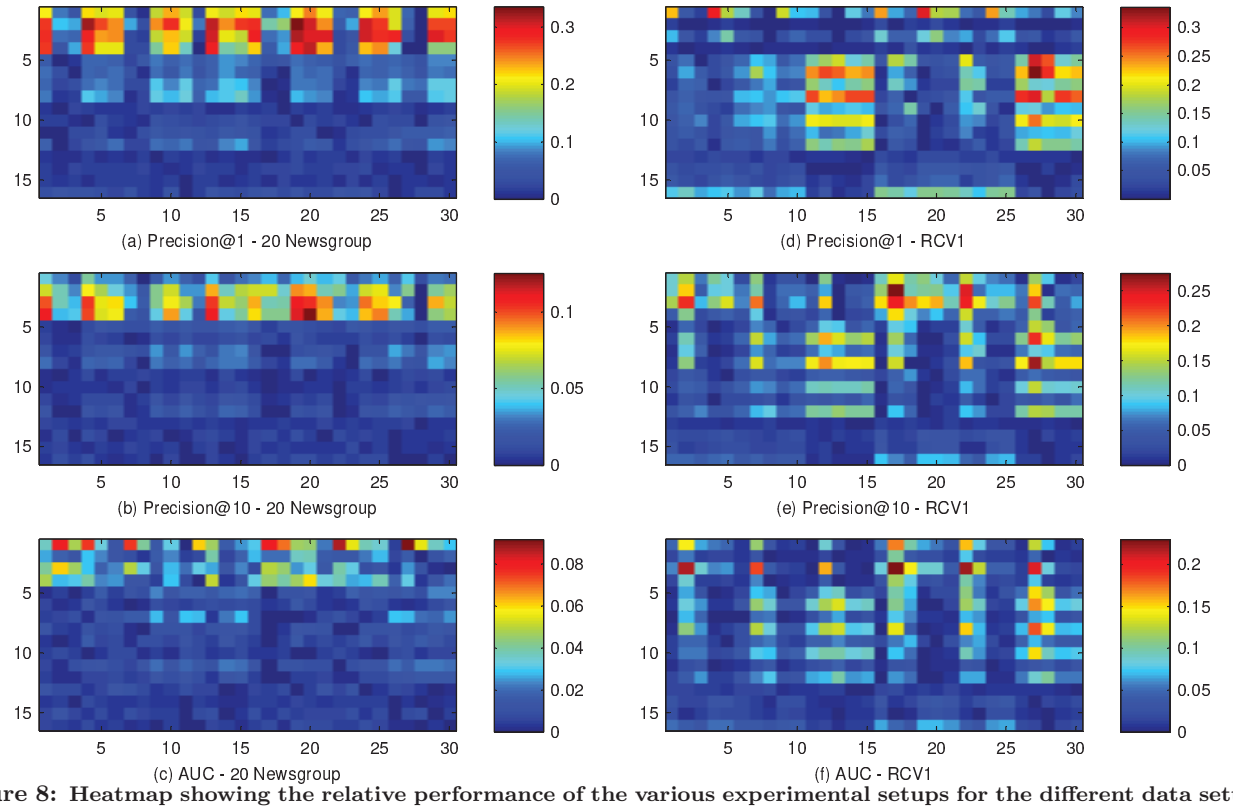


Figure 8: Heatmap showing the relative performance of the various experimental setups for the different data settings for FCD. Figure 10 shows the respective indexing scheme for the heatmap.

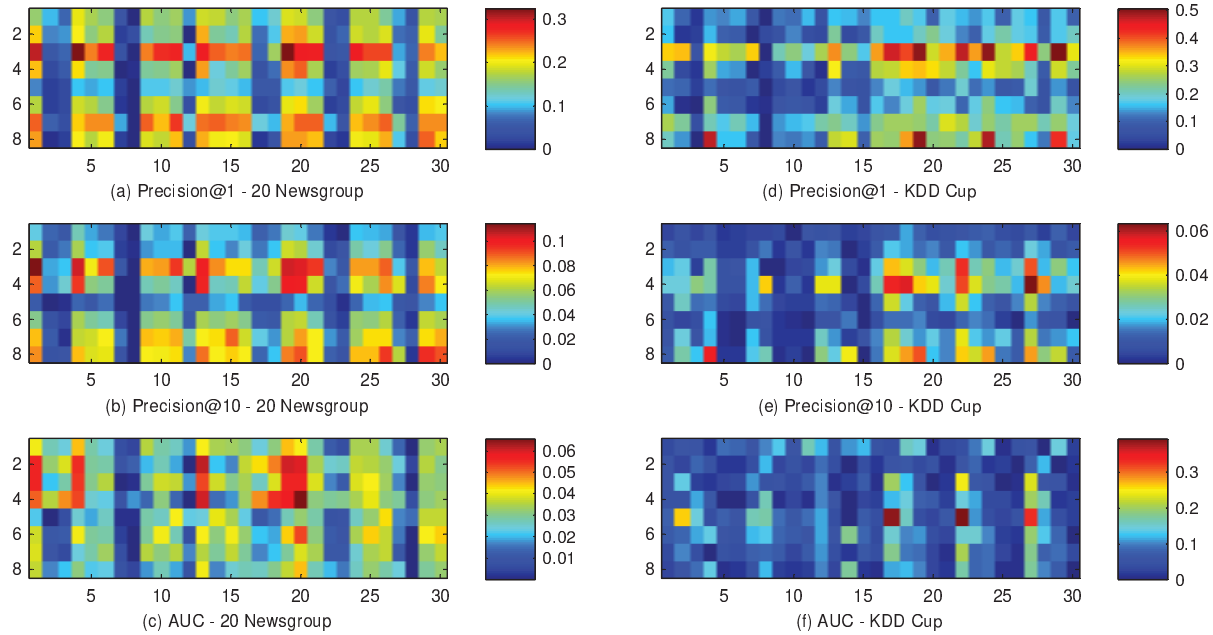


Figure 9: Heatmap showing the relative performance of the various experimental setups for the different data settings for SSD. Figure 11 shows the relative indexing scheme for the heatmap.

				model type=single																model type=ensemble											
				weighing scheme=exp					weighing scheme=linear					weighing scheme=uniform					weighing scheme=exp					weighing scheme=linear					weighing scheme=uniform				
				rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl
			Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
No Drift	pos=10%	Queries=10	1	0.23	0.36	0.39	0.22	0.24	0.24	0.40	0.43	0.25	0.21	0.25	0.41	0.23	0.25	0.22	0.24	0.36	0.34	0.23	0.19	0.23	0.37	0.37	0.23	0.20	0.24	0.37	0.44	0.24	0.22
		Queries=100	2	0.31	0.52	0.53	0.31	0.36	0.35	0.57	0.57	0.41	0.37	0.40	0.58	0.29	0.39	0.41	0.30	0.52	0.50	0.28	0.29	0.29	0.56	0.55	0.29	0.31	0.31	0.56	0.59	0.29	0.32
	pos=20%	Queries=10	3	0.32	0.50	0.50	0.28	0.33	0.36	0.51	0.57	0.36	0.32	0.39	0.52	0.29	0.36	0.33	0.32	0.46	0.44	0.27	0.30	0.36	0.54	0.50	0.29	0.33	0.41	0.54	0.60	0.33	0.33
		Queries=100	4	0.46	0.68	0.63	0.45	0.53	0.53	0.70	0.70	0.58	0.51	0.60	0.73	0.44	0.59	0.57	0.51	0.67	0.63	0.44	0.42	0.50	0.70	0.68	0.50	0.48	0.60	0.72	0.73	0.57	0.57
Drift 20%	pos=10%	Queries=10	5	0.13	0.19	0.17	0.12	0.12	0.12	0.16	0.13	0.11	0.12	0.11	0.13	0.12	0.09	0.10	0.12	0.15	0.14	0.11	0.12	0.13	0.15	0.12	0.11	0.12	0.12	0.13	0.15	0.10	0.10
		Queries=100	6	0.20	0.24	0.23	0.19	0.18	0.19	0.24	0.23	0.15	0.18	0.15	0.19	0.19	0.15	0.15	0.18	0.26	0.25	0.18	0.18	0.19	0.24	0.25	0.18	0.18	0.15	0.18	0.19	0.15	0.15
	pos=20%	Queries=10	7	0.21	0.25	0.29	0.21	0.22	0.21	0.25	0.28	0.21	0.23	0.20	0.24	0.22	0.20	0.22	0.22	0.27	0.29	0.21	0.22	0.21	0.25	0.28	0.22	0.22	0.21	0.22	0.24	0.21	0.21
		Queries=100	8	0.30	0.37	0.37	0.27	0.29	0.30	0.38	0.36	0.27	0.29	0.26	0.32	0.29	0.27	0.26	0.29	0.38	0.39	0.28	0.30	0.30	0.38	0.38	0.28	0.29	0.27	0.33	0.33	0.27	0.27
Drift 50%	pos=10%	Queries=10	9	0.12	0.13	0.10	0.11	0.11	0.11	0.11	0.10	0.11	0.10	0.11	0.10	0.12	0.12	0.11	0.11	0.11	0.13	0.11	0.11	0.11	0.12	0.12	0.12	0.11	0.11	0.11	0.12	0.11	0.11
		Queries=100	10	0.17	0.17	0.14	0.14	0.14	0.16	0.17	0.14	0.12	0.14	0.13	0.13	0.14	0.12	0.12	0.13	0.16	0.15	0.14	0.14	0.17	0.16	0.14	0.14	0.14	0.12	0.12	0.11	0.11	0.11
	pos=20%	Queries=10	11	0.23	0.23	0.24	0.25	0.21	0.23	0.23	0.24	0.20	0.21	0.22	0.22	0.22	0.21	0.20	0.22	0.22	0.22	0.24	0.22	0.23	0.22	0.23	0.24	0.21	0.22	0.22	0.24	0.21	0.20
		Queries=100	12	0.24	0.30	0.28	0.26	0.25	0.23	0.28	0.28	0.23	0.23	0.22	0.26	0.25	0.22	0.20	0.25	0.29	0.27	0.27	0.24	0.24	0.29	0.27	0.25	0.24	0.23	0.25	0.25	0.21	0.21
Drift 100%	pos=10%	Queries=10	13	0.09	0.10	0.10	0.09	0.10	0.09	0.10	0.09	0.10	0.09	0.09	0.09	0.10	0.09	0.10	0.10	0.11	0.10	0.08	0.09	0.09	0.09	0.10	0.08	0.09	0.09	0.11	0.10	0.09	0.09
		Queries=100	14	0.09	0.10	0.09	0.10	0.09	0.10	0.10	0.10	0.10	0.10	0.11	0.09	0.09	0.10	0.10	0.09	0.11	0.09	0.10	0.10	0.09	0.10	0.09	0.10	0.10	0.11	0.10	0.10	0.10	0.10
	pos=20%	Queries=10	15	0.20	0.21	0.20	0.20	0.20	0.20	0.20	0.20	0.21	0.21	0.20	0.19	0.20	0.20	0.21	0.20	0.20	0.20	0.19	0.20	0.20	0.20	0.19	0.19	0.21	0.20	0.20	0.22	0.19	0.21
		Queries=100	16	0.18	0.17	0.18	0.18	0.19	0.20	0.18	0.20	0.21	0.18	0.22	0.20	0.18	0.18	0.19	0.18	0.17	0.19	0.17	0.19	0.19	0.17	0.18	0.17	0.19	0.22	0.20	0.19	0.19	0.20

Figure 10: Table indicating the indexing scheme for the Heatmaps in Figure 8. The data shown in the table correspond to Figure 8(a) - Precision@1 for 20 Newsgroup dataset. The abbreviated naming convention for the active learning strategies are: rand - random; cer - certainty; uncer - uncertainty; den - density; outl - density outlier

				weighing scheme=exp					weighing scheme=linear					weighing scheme=uniform					weighing scheme=exp					weighing scheme=linear					weighing scheme=uniform				
				rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl	rand	cer	uncer	den	outl
			Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Drift=Low	pos=10%	Queries=10	1	0.18	0.24	0.27	0.14	0.17	0.19	0.29	0.34	0.16	0.19	0.17	0.25	0.16	0.16	0.19	0.17	0.24	0.21	0.15	0.15	0.18	0.29	0.31	0.16	0.17	0.20	0.25	0.30	0.17	0.17
		Queries=100	2	0.25	0.39	0.38	0.26	0.31	0.29	0.45	0.43	0.33	0.32	0.32	0.47	0.24	0.32	0.33	0.29	0.39	0.36	0.24	0.24	0.28	0.43	0.40	0.28	0.29	0.28	0.40	0.41	0.27	0.32
	pos=20%	Queries=10	3	0.31	0.55	0.54	0.29	0.37	0.33	0.57	0.61	0.37	0.34	0.33	0.51	0.32	0.35	0.37	0.36	0.54	0.44	0.28	0.32	0.33	0.56	0.55	0.32	0.35	0.35	0.51	0.54	0.37	0.39
		Queries=100	4	0.53	0.72	0.69	0.54	0.60	0.60	0.75	0.74	0.60	0.60	0.59	0.75	0.52	0.63	0.61	0.58	0.71	0.68	0.51	0.50	0.58	0.73	0.71	0.59	0.58	0.55	0.63	0.69	0.56	0.59
Drift=High	pos=10%	Queries=10	5	0.16	0.22	0.21	0.13	0.14	0.13	0.22	0.24	0.13	0.13	0.11	0.15	0.14	0.11	0.12	0.14	0.19	0.19	0.13	0.12	0.13	0.18	0.22	0.13	0.12	0.10	0.16	0.19	0.12	0.12
		Queries=100	6	0.25	0.41	0.40	0.24	0.25	0.27	0.41	0.42	0.25	0.24	0.28	0.39	0.22	0.24	0.23	0.24	0.40	0.39	0.23	0.21	0.26	0.40	0.40	0.25	0.23	0.27	0.38	0.39	0.21	0.22
	pos=20%	Queries=10	7	0.29	0.47	0.50	0.32	0.30	0.30	0.44	0.54	0.29	0.32	0.27	0.42	0.29	0.29	0.30	0.30	0.42	0.44	0.31	0.29	0.29	0.44	0.49	0.32	0.30	0.28	0.39	0.47	0.30	0.30
		Queries=100	8	0.48	0.70	0.69	0.50	0.53	0.50	0.69	0.71	0.52	0.53	0.48	0.67	0.48	0.51	0.50	0.52	0.69	0.66	0.48	0.47	0.49	0.69	0.69	0.51	0.51	0.48	0.60	0.67	0.45	0.49

Figure 11: Table indicating the indexing scheme for the Heatmaps in Figure 9. The data shown in the table correspond to Figure 9(a) - Precision@1 for 20 Newsgroup dataset. The abbreviated naming convention for the active learning strategies are: rand - random; cer - certainty; uncer - uncertainty; den - density; outl - density outlier

of differences is very different for the three metrics. '0' value in the heatmap relates to the best performing modeling scheme. The larger the value in this heatmap for the modeling strategy the worse it performs. The data in Figure 10 corresponds to raw values (prior to normalization) used to generate Figure 8(a) and the data in Figure 11 corresponds to Figure9(a). For instance, for Figure 8(a), row index 3 corresponds to row 3 of Figure 10, where the data has 'No Drift'; percentage of positive examples is 20% and number of queries per time period is 10. The best performing learning choice is for index '28', which corresponds to an ensemble model with uniform instance weighting scheme and using 'uncertainty' active sampling strategy.

Which active sampling strategy in general performs better? In general, uncertainty sampling is the best active sampling choice. For SSD, the second best choice for active sampling is certainty based sampling. Density based sampling is not well suited for SSD.

Which type of model in general performs better? For SSD, the best performing models are single models rather than ensembles. For FCD, ensemble models perform better than single models. The intuitive explanation is that since the true concept is not changing for SSD (only the subclass frequency within the broader concept is changing), learning a single model that represents the concept fares better. For FCD, where the true concept is changing, learning models for different time periods which represent the concept for that time period helps by possibly learning disjoint concepts, which is not possible with a single linear model.

Which weighting scheme in general performs better? For FCD, the exponential weighting scheme works better than the linear and uniform weighting schemes for both single and ensemble model types. For SSD, a linear weighting scheme works better than the exponential and uniform weighting schemes. This difference in weighting scheme for FCD and SSD seems intuitive because for SSD, history is more useful than for FCD and forgetting the history slowly (linearly) helps for SSD whereas forgetting the history faster (exponentially) helps for FCD.

These results provide crucial insights indicating that the optimal design choices for interactive systems, need to consider broader domain parameters rather than adopting a 'best practice' strategy. A sampling strategy that focuses on detecting the drift and then explicitly sample examples to reflect it, will perform better than drift-agnostic (or random) strategies.

5. CONCLUSION

We find that active sampling performs statistically better than random sampling in nearly half the temporal drift problem setups, while being comparable in the remaining setups. Further away the drift take the subclasses, less advantageous is using active learning stategies. Performance gain is more prominent for focused metrics and less so for aggregated metrics. In general, uncertainty based sampling strategy was the best active learning strategy. We found that

the techniques developed in temporal drift literature namely instance weighting and weighted ensembles combined with active learning gave better results. Some intuitive patterns that were validated by the results were: a) ensemble models are better for FCD whereas building a single model (with instance weighting) is better for SSD b) exponential weighting scheme is better for FCD whereas linear weighting scheme is more effective for SSD. The optimal design choices for interactive systems in the presence of temporal drift, needs to consider the broader domain parameters rather than adopting a 'best practice' strategy.

6. REFERENCES

[1] R. Bekkerman and M. Scholz. Data weaving: scaling up the state-of-the-art in data clustering. In *Proc of the 17th ACM CIKM*, 2008.

[2] C. C. Chang and C. J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[3] W. Chu, M. Zinkevich, L. Li, A. Thomas, and B. Tseng. Unbiased online active learning in data streams. In *Proc of the 17th ACM SIGKDD*, 2011.

[4] P. Donmez and J. G. Carbonell. Active sampling for rank learning via optimizing the area under the roc curve. In *Proc of the 31th ECIR*, 2009.

[5] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In *Applications of Data Mining in Computer Security*. Kluwer, 2002.

[6] Z. Ferdowsi, R. Ghani, and M. Kumar. An online strategy for safe active learning. In *ICML Workshop on Combining Learning Strategies to Reduce Label Cost*, 2011.

[7] G. Forman. Tackling concept drift by temporal inductive transfer. In *Proc of the 29th SIGIR*, 2006.

[8] R. Ghani and M. Kumar. Interactive learning for efficiently detecting errors in insurance claims. In *Proc of the 17th ACM SIGKDD*, 2011.

[9] T. R. Hoens, R. Polikar, and N. V. Chawla. Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artificial Intelligence*, 2012.

[10] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proc of the 7th ACM SIGKDD*, 2001.

[11] N. Japkowicz and M. Shah. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.

[12] KDD-Cup. Kdd cup, 1999.

[13] R. Klinkenberg and T. Joachims. Detecting concept drift with support vector machines. In *Proc of the 17th ICML*, 2000.

[14] J. Z. Kolter and M. A. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal Machine Learning Research*, 2007.

[15] J. Rennie. 20 newsgroups data set, 2007.

[16] M. Scholz and R. Klinkenberg. Boosting classifiers for drifting concepts. *Intelligent Data Analysis*, 2007.

[17] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[18] W. N. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Proc of the 7th ACM SIGKDD*, 2001.

[19] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes. Active learning with evolving streaming data. In *Proc of the ECML PKDD-part III*, 2011.

[20] H. Wang, W. Fan, P. S. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *Proc of the 9th ACM SIGKDD*, 2003.

[21] I. Zliobaite. Learning under concept drift: an overview. *CoRR*, abs/1010.4784, 2010.

RedRock: Interactive Analysis and Visualization of Very Large Data Sets

Hao Wang
IBM Silicon Valley Lab
555 Bailey Avenue
San Jose, CA 95141
haowang@us.ibm.com

Albith Joel Colon Figueroa
IBM Silicon Valley Lab
555 Bailey Avenue
San Jose, CA 95141
ajcolonf@us.ibm.com

ABSTRACT

In this paper, we describe a system for interactively analyzing and visualizing very large data sets and demonstrate a prototype we have built so far[1]. In recent years the requirement for analyzing and visualizing very large data sets is emerging rapidly in both industries and academics due to the large volume of digital data generated by human and machines. Therefore, there is an urgent need for good tools that enable users (e.g., data analysts, data scientists and researchers) to accomplish such tasks. A variety of systems and tools have emerged to solve this problem in the last decade but unfortunately most of them require special skills such as programming or SQL. For users with extensive domain knowledge but have limited or no programming skills, their toolbox for dealing with very large data sets is extremely limited. We designed and developed a system that aims to mitigate this gap. The system includes two major components: 1) a server based on Apache Hadoop and Spark that provides a web service for quick analysis of very large data sets. It supports common data transformations, text extraction, training and applying machine-learning models; 2) a GUI that allows the user to interactively carry out operations supported by the server as well as creating dynamic visualizations.

Keywords

Big data, analytics, visualization, Hadoop, Spark, GUI, exploratory analysis, information extraction

1. INTRODUCTION

In the last decades, the volume of data generated by human and machines has been growing exponentially. Some examples of very large data sets (VLDS) are Wikipedia, Twitter, weather, machine logs, sensor data and also the entire web as indexed by the search engines. Wikipedia now contains more than 36 million pages [1]. Twitter users generate about 400 – 500 million short posts per day according to our Twitter data collection. VLDS like these often have millions or billions of records. Storing and processing of such huge amount of data was a challenge in the past. With the dramatic decrease of the cost of data storage and computing power, Apache Hadoop has emerged to allow distributed processing of VLDS. Because of its scalability and some other advantages, the Hadoop eco-system is growing fast and becoming exuberant. Since 2009, Apache Spark was developed to provide a better distributed computing engine. Comparing to Hadoop, one particular improvement in Spark is the support of in-memory caching of data between operations so it is much faster in many tasks. It has been shown that "Spark can outperform Hadoop by 10x in iterative machine learning jobs, and can be used to interactively query a 39 GB dataset with sub-second response time" [2]. With Apache Spark interactive analysis of VLDS becomes a real possibility. However, it is only accessible to users through an API via one of the three programming languages (Scala, Java and Python)[2]. There needs a tool to bring the power of distributed data processing to users without programming skills.

Visualizing VLDS is a problem because it is difficult to show all of the data points in a chart and it is difficult for human to make sense of what that means. It is necessary to reduce the data to meaningful size or subsets so visualization of such would make sense and easy to consume for the user.

There are a rich set of tools for data analysis and visualization, including tools with a graphical user interface (such as, Many eyes, Wolfram Alpha and Trifacta) as well as programming languages and tools (such as, R, Excel, Spark, Map reduce, Hive and BigSQL). Each of them has its own advantages and disadvantages. For example, Excel is very good at quickly manipulating data and creating charts. But it has a limit of 1 million rows. Spark is very good at scaling to VLDS and is also known for its processing speed because it can cache data in memory between operations. But one can only access it through one of the three programming languages it supports.

We believe it is extremely important to have a tool that is scalable to VLDS and accessible to users without programming skills. Such a tool has to satisfy the following requirements:

1. Scales well with VLDS
2. Quick response to allow interactive analysis and visualization
3. An intuitive interface designed for users with and without programming skills

In the next section, we describe in detail the design of a novel system based on Spark that fulfills those three requirements. And we will show a prototype via two use cases that it is very promising for interactive analysis and visualization of VLDS.

[1]Demo recording available at https://vimeo.com/128706075 (password: redrock4you).

[2]Spark also has some support for R and SQL but some programming is still required.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15). August 10th, 2015, Sydney, Australia.

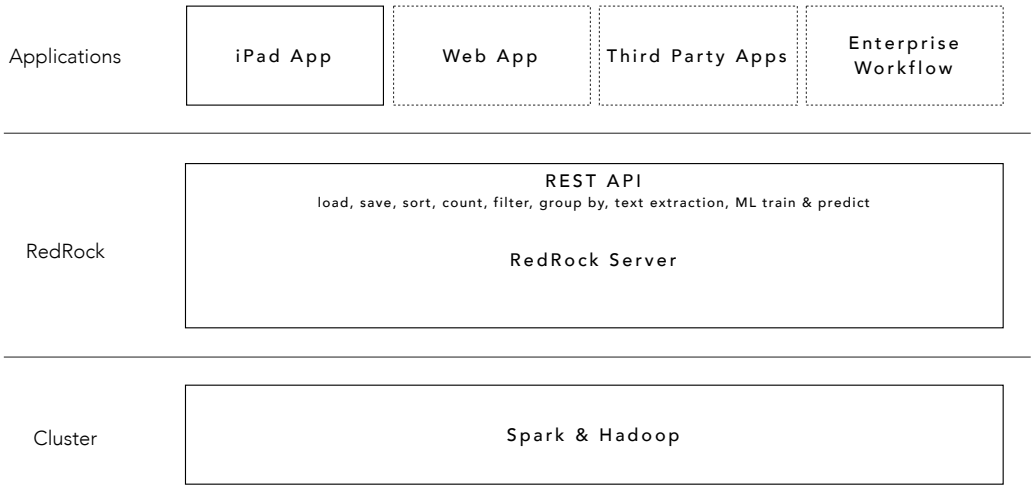


Figure 1. System architecture (the dotted boxes are for illustration purpose)

2. SYSTEM DESIGN

2.1 Overview

The system has the typical client-server architecture (see Figure 1). The server runs on a Hadoop and Spark cluster and provides a web service. It exposes a set of REST APIs for common data transformations (e.g., sort, group by and pivot) and analytics operations (e.g., text extraction, train/apply machine learning models). The server receives a request via the REST API (such as, group by), executes the corresponding operation using Spark, and then sends back the results or a sample of that.

The front-end we built for this prototype is a native iPad app, which interacts with the user, sends an operation to the server and displays the result from the server on a spreadsheet-like UI. It supports creating common visualizations on the device.

Because the server exposes a set of REST APIs, the client could also be a web app, a desktop application from a third party, or an enterprise workflow. The system is in active development so that what we are describing here is an evolving prototype.

2.2 Data model

The data model in the system is represented as a 2D array, which effectively represents rows and columns in a data set. It was chosen for the following reasons: 1) a table with rows and columns is a natural way of representing structured data; 2) this concept is popularly used in many software, such as Excel, SPSS and Spark's DataFrame. In many big data use cases, the input data is often unstructured (e.g., customer feedback in the form of plain text files) or semi-structured (e.g., Twitter data in JSON format). The system is designed to easily handled data in those forms as well. For example, the user can load text files in a directory as whole files from Hadoop (i.e., the content of a text file is loaded into one cell of the 2D array). This is particularly useful in many machine learning use cases as demonstrated in Section 3.2. As another example, any JSON data, which may have objects in multiple levels, is flattened out and converted into a table of rows and columns. This makes it easier for the front-end to display the data. Besides the 2D array that contains the actual data, the system also keeps certain meta-data, such as the name of the columns (e.g., the header of each column).

Since the system is for interactive data analysis, it also has the concept of a session that keeps the history of the operations from a user so the user can easily undo and redo any previous operation.

2.3 The REST APIs

The server exposes a set of REST APIs for common data transformations and analytics. Table 1 lists the operations that are currently supported. Within this framework, adding new operations is a straightforward task.

Table 1. The REST APIs currently supported by the server

API	Description
load	Load data from Hadoop and return a sample
save	Save current data to Hadoop
count	Count number of rows
sort	Sort based on the values in a column
filter	Filter by keywords or regular expressions
group by	Aggregate based on the one or two columns
pivot	Create a pivot table from two columns
extract	Extract sentiment, location or tokens from text
substring	Apply substring to a column of text
ml/train	Train a machine learning classifier
ml/predict	Apply an existing machine learning classifier
hdfs/list	List files in a directory on Hadoop
history/back	Undo the last operation

The REST APIs are the communication protocol between the server and clients. Any client that conforms to this API will be able to fully utilize the data processing and analytics capabilities of the server. For instance, when the need arises for a client that runs on a desktop web browser (a.k.a., web app), it can be built without requiring any change to the server, which is very desirable in software engineering. In another case, the server could be easily integrated into some enterprise workflow for automation.

On the server side, if we need to switch to a new technology for storing or processing big data in the future a significant part of the server may need to be changed. There will be no effect to the clients if the server provides the same REST APIs. In summary, this REST-API-based architectural design allows the separation of the server and clients. It nicely hides the internals of the server and client from each other.

2.4 Server and data processing

Conceptually the server's tasks are straightforward: it receives a HTTP request from a client, parse the parameters, execute the request as a set of Spark operations, and send the result (sometimes a sample of it) back to the client.

For many operations, it is a direct mapping to Spark's operations (e.g., count, group by, join). For some other operations, their implementation is more complicated. For example, to create a machine-learning model the server needs to do feature extraction, recoding and training of the model; to apply a model, it needs to do feature extraction, recoding, applying the model and convert the output to labels.

In the cases when the server only sends back a sample of the result, it actually does "lazy" computation to just get the sample result so the server responds "instantly". The server only executes on the entire data set when necessary. This feature is provided by Resilient Distributed Datasets (RDDs) in Spark [3]. For example, applying substring on a data set of 100 million of rows will not actually compute the substrings of all the rows but just the first 1000 rows if the client is only displaying 1000 rows initially. If computation is required on the entire set in the next operation, Spark will automatically starts the computation of the dependent operations. This feature is important and favorable because it allows the analysis on big data to be real interactive and instant for many operations.

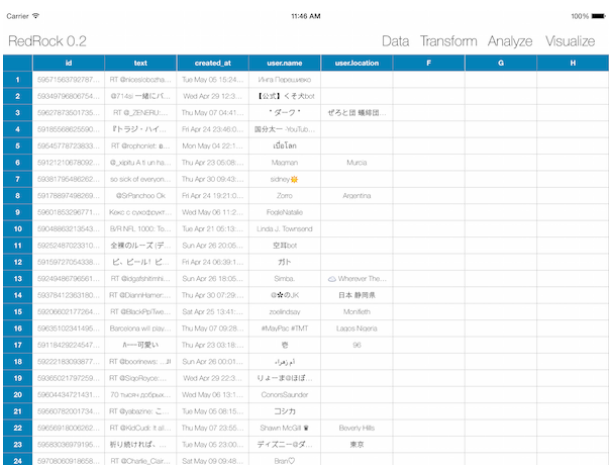


Figure 2. Main UI of the iPad app

2.5 Front-end and user interaction

For the purpose of demonstration, we built the client as a native iPad app. The app has a spreadsheet-like UI. Similar to Excel and SPSS, the grid in the UI displays data in rows and columns. The operations a user can execute are grouped into four general categories (as the tabs on top right): *Data*, *Transform*, *Analyze* and *Visualize*.

Under *Data*, it is loading data from different sources including Hadoop, Twitter, Weather and more in the future (Figure 3). It also allows the user to save results to Hadoop.

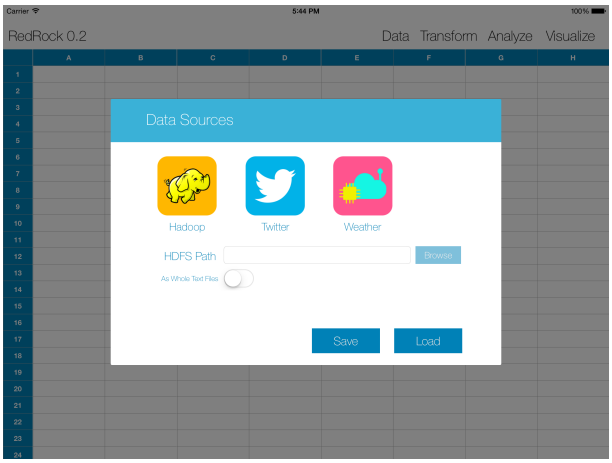


Figure 3. Load data from Hadoop and other data sources

Transform includes all the data transformations listed in Table 1 except the machine learning ones (Figure 4).

Analyze is for training and applying machine-learning models (Figure 5). We intend to provide the most common machine-learning models and features in a way that is understandable to novice users. As an experimental feature, it currently only supports the Naïve Bayes classifier and a limited set of features (unigrams and bigrams).

Visualize is discussed in the next sub-section.

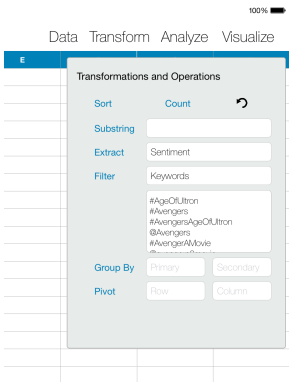


Figure 4. Data transformations

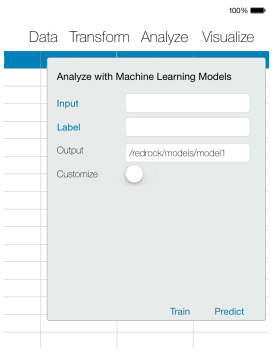


Figure 5. Analyze using machine-learning models

2.6 Visualization

Visualize is the central place for creating various charts and map. Currently it supports line, bar, pie, donut charts and world map (see Section 3 for examples). The charts were developed based on the D3 JavaScript library [4]. D3 is a very powerful library and is widely used to create different types of visualizations.

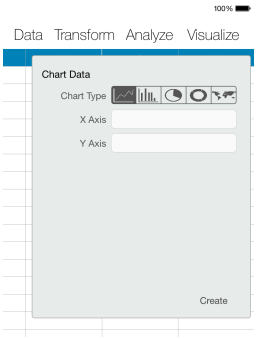


Figure 6. Visualization tab for creating charts and map

In the app, when the user creates visualization, the app passes the data (in the form of a 2D array) and parameters (such as X and Y axes) to the particular chart or map. Within this framework, developers or even advanced users can easily add new charts or adapt existing D3 visualizations.

The charts currently support limited interactions with the user. For example, on the bar chart, the user can choose grouped or stacked bars. On the map, the user can "play" it or move the slide to select a certain date. More interactions may be added since a chart is relatively independent from other parts of the app.

3. USE CASE STUDIES

3.1 Twitter Data Analysis

3.1.1 Problem

The goal of this use case is to analyze the expectations and feedback to the movie *Avengers: Age of Ultron* on Twitter before and after its release.

3.1.2 Data

The movie was released on May 1, 2015 in the US. We collected Twitter data via the Decahose API (which provides 10% of entire Twitter traffic) from Apr 20 to May 9. The total number of tweets we have collected in this period is about 800 million. Due to the size of our cluster, we randomly selected a subset of the tweets plus all tweets that mentioned the movie, which are about 100 million tweets in total.

The cluster includes 10 nodes (8 cores and 16 GB of memory each) and runs Hadoop 2.2 and Spark 1.3.0.

3.1.3 Analysis

To understand Twitter users' feedback on the movie, we analyzed the volume, sentiment of the tweets related to the movie and the locations of those users. To assess the interactivity of the system, the time taken for the system to respond to each operation was recorded.

3.1.4 Results

Figures 7, 8 and 9 illustrate the visualizations created in this analysis. Figure 7 shows the volume of tweets per day. Figure 8 shows the volume of positive and negative sentiments per day. Figure 9 shows a map of tweet volume for each country over time with animated transition for the bubble size.

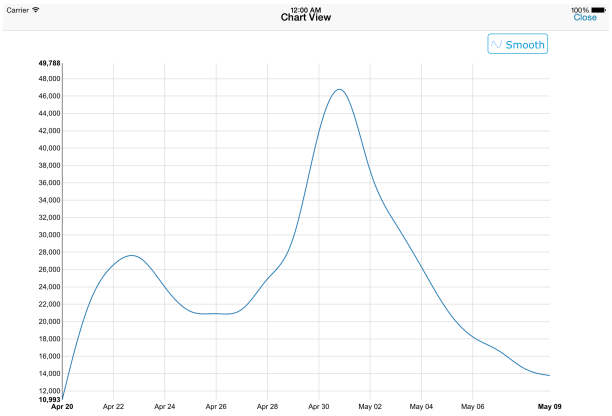


Figure 7. Volume of tweets per day

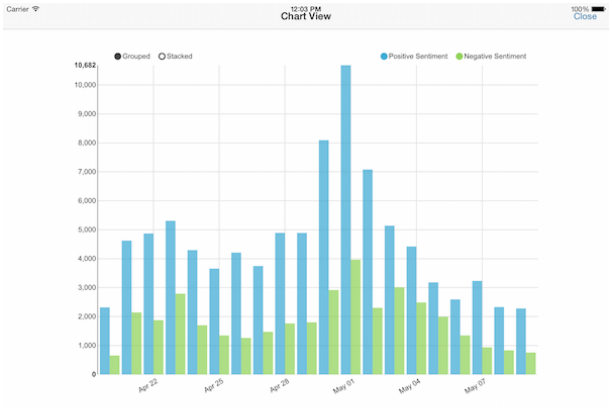


Figure 8. Volume of positive and negative sentiments

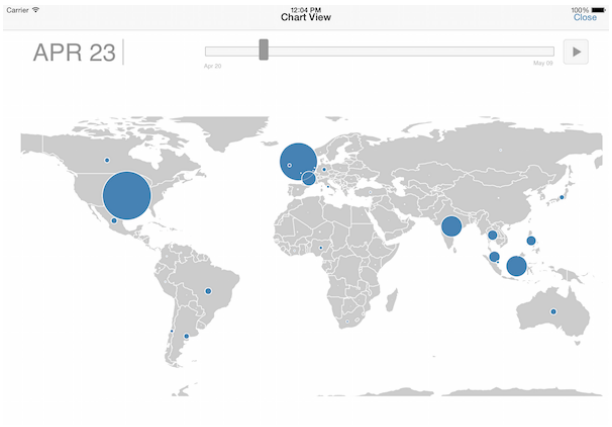


Figure 9. Volume of tweets by country over time

Table 2 shows the response time for each operation. Note that creating visualizations is a client side operation so it is basically instant. Loading the data took only 1.4 seconds. While the user is inspecting the first 1000 samples of the data, the server was caching the entire data set to an in-memory cache in the background. When the data is cached in memory, the rest of the operations take at most a few seconds to complete. Pivot and group-by took little longer because these operations require computation on the entire data set. Overall, this analysis was extremely responsive and interactive. In just a few minutes, we were able to start with 100 millions tweets, filter it down to about half a million relevant ones, apply text extraction algorithms, and visualize the results. For conducting exploratory analysis of VLDS like the current one, this tool is very valuable particularly for users who have no experiences in programming languages or SQL.

Table 2. Response time of the operations for the Twitter data analysis use case

Operation	Response Time (seconds)
Load data (save to in-memory cache in the background)	1.4 (60.0)
Count	6.3
Filter by keywords (save results to in-memory cache in the background)	0.6 (22.0)
Count	1.4
Extract date from timestamp	0.3
Group by on date	1.4
Sort by date	1.5
Create a line chart for volume	0*
Undo	0.5
Extract sentiment	1.2
Extract location	1.4
Pivot by date and sentiment	14.3
Create a bar chart for sentiment	0*
Undo	0.8
Group by date and location	14.7
Create a map for user location	0*

* Client side operation

3.2 Document Classification

3.2.1 Problem

Classifying text documents based on the content is a very common use case yet it requires knowledge of machine-learning algorithms, feature extraction and programming using existing ML libraries.

3.2.2 Data

We will use a subset of the 20 Newsgroups data set [5]. The original data set is a collection of approximately 20,000 newsgroup documents, partitioned almost evenly across 20 different newsgroups. Each newsgroup corresponds to a different topic. Table 3 lists the 20 topics. In this use case, we randomly select 5,000 documents as the training set and 2,500 documents as the test set due to the cluster size.

3.2.3 Analysis

Because machine learning is an experimental feature, the system currently supports only Naïve Bayes and a limited set of features, such as unigram and bigram. The purpose of this demonstration is to showcase that even a novice user can quickly apply a machine-learning algorithm in this system instead of finding the best classifier. This analysis trained a Naïve Bayes classifier with the bag-of-words model (using the most frequent 1000 unigrams) and applied the classifier to the test set. Feature extraction and recoding were carried out by the server through Spark. The server uses the SystemML library as the implementation of the Naïve Bayes algorithm [6].

Table 3. The 20 newsgroups

alt.atheism	rec.sport.hockey
comp.graphics	sci.crypt
comp.os.ms-windows.misc	sci.electronics
comp.sys.ibm.pc.hardware	sci.med
comp.sys.mac.hardware	sci.space
comp.windows.x	soc.religion.christian
misc.forsale	talk.politics.guns
rec.autos	talk.politics.mideast
rec.motorcycles	talk.politics.misc
rec.sport.baseball	talk.religion.misc

3.2.4 Results

Table 4 shows the response time for each operation. Loading the text files from Hadoop and returning a sample of the data to the front-end is very quick (0.2 seconds). It took 41 seconds to cache all the files in memory in the background while the user was inspecting the sample files. Training and applying the classifier are the longest running operations, which were expected since these operations include feature extraction, recoding and the actual training and applying of the classifier on the entire data set. The rest of the operations were nearly instant. Most importantly, it only took a few steps to train and apply a machine-learning model. This ease of use is critical for novice users because they do not have to learn programming to unleash the power of common machine-learning algorithms. Note that, although this analysis used only thousands of documents, it could scale to very large number of documents without any effort from the user.

Table 4. Response time of operations for the document classification use case

Operation	Response Time (seconds)
Load text files of the training set (and save to in-memory cache in the background)	0.2 (41.0)
Count	0.2
Train a Naïve Bayes Classifier and save it to Hadoop	20.2
Load text files of the test set (and save to in-memory cache in the background)	0.2 (24.0)
Count	0.2
Apply the classifier	12.0
Group by the predicted label	0.1
Sort by the number of documents with each predicted label	0.5
Create a pie chart for the number of documents with each predicted label	0*

* Client side operation

4. CONCLUSION AND FUTURE WORK

We described the design of a system for interactively analyzing and visualizing very large data sets. We demonstrated a prototype through two use cases: Twitter data analysis and document classification. In the first use case, it has shown that a user was able to start with 100 millions tweets, filter it down to about half a million relevant ones, apply text extraction algorithms, and visualize the results all in just a few minutes through a GUI. The second use case showed a simple user interface for novice users to quickly apply machine-learning algorithms to their data.

The system is aimed to help users with limited or no programming skills to quickly and interactively analyze and visualize very large data sets. It supports common data transformations, visualizations and limited machine learning capabilities. Although the current prototype is limited in some aspects, it is straightforward to add the support for more transformations, visualizations and machine learning algorithms. Nevertheless, it shows the potential that such a system would be extremely useful for a huge number of users who have no programming knowledge.

To improve the current prototype, a number of features would be very desirable. For example, predicting and automatically caching data is preferable than having the user to manually cache the data after an operation. Another example would be integrating data profiling capabilities to the tool so the user can quickly understand the shape and quality of a data set. Having a web app (running in a web browser instead of an iPad) also appeared frequently in the feedback from our internal demonstrations.

In summary, the system demonstrated here is very promising as a basic tool for interactively analyze and visualize very large data sets. It is particularly valuable for users with no programming skills but need to work with very large data sets.

5. REFERENCES

[1] Wikipedia, 2015. Size of Wikipedia. http://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia

[2] Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., and Stoica, I., 2010. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, 10-10.

[3] Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., and Stoica, I., 2012. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation* USENIX Association, 2-2.

[4] Bostock, M., Ogievetsky, V., and Heer, J., 2011. D³ data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12, 2301-2309.

[5] The 20 Newsgroups data set. http://qwone.com/~jason/20Newsgroups/

[6] Ghoting, A., Krishnamurthy, R., Pednault, E., Reinwald, B., Sindhwani, V., Tatikonda, S., Tian, Y., and Vaithyanathan, S., 2011. SystemML: Declarative machine learning on MapReduce. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on* IEEE, 231-242.

Interactive Data Repositories: From Data Sharing to Interactive Data Exploration & Visualization

Ryan A. Rossi
Dept. of Computer Science
Purdue University
rrossi@purdue.edu

Nesreen K. Ahmed
Dept. of Computer Science
Purdue University
nkahmed@purdue.edu

ABSTRACT

Scientific data repositories have historically made data widely accessible to the scientific community, and have led to better research through comparisons, reproducibility, as well as further discoveries and insights. Despite the growing importance and utilization of data repositories in many scientific disciplines, the design of existing data repositories has not changed for decades. In this paper, we revisit the current design and envision interactive data repositories, which not only make data accessible, but also provide techniques for interactive data exploration, mining, and visualization in an easy, intuitive, and free-flowing manner.

Categories and Subject Descriptors

G.2.2 [**Graph theory**]: Graph algorithms; H.2.8 [**Database Applications**]: Data Mining; H.3.3 [**Information Storage and Retrieval**]: Relevance feedback; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

Keywords

Interactive data repository, data archive, digital library, visual analytics, interactive visualization, interactive graph mining, graph visualization, network science, sensemaking, network repository

1. INTRODUCTION

Scientific progress often relies on standard data sets for which claims, hypotheses, and algorithms can be compared and evaluated. In recent years, scientific data repositories have made data widely accessible to the broader scientific community, and have led to better research practices through comparisons, reproducibility, as well as further discoveries and innovations. Such data repositories are proving to be increasingly valuable to many scientific disciplines (e.g., computer science, bioinformatics, etc.) [6, 3], while other disciplines have only recently considered data sharing (e.g., ecology, evolutionary biology, and psychology) [15, 9]. Furthermore, the recent hype of big data has fueled the importance of sharing data for the greater good (e.g., healthcare, climate change). Hence, sharing data and making it accessible is quickly becoming a standard, and in many disciplines is now a requirement for funding [7, 8]. All of these reasons have led to the growing number of data repositories and their widespread use across a variety of disciplines.

Despite the growing importance and utilization of data repositories in many scientific disciplines, the design of existing data repositories has not changed for decades. Most existing data repositories are designed for data sharing and management rather than for scientific inquiry, which impedes the possibility to easily explore the data and ask novel questions beyond the questions that sparked data collection. This is due to the current design of data repositories, which lacks interactive visual analytics [13, 5, 2], mining, and statistical tools that make it easier to understand, explore, and find new and important patterns in the data.

In this paper, we revisit the traditional data repository concept that has been widely used for decades, and instead, we envision *interactive data repositories* (IDR) [10, 11], an alternative approach for the design of future data repositories, which not only makes data accessible, but also provides techniques and tools to find, understand, and explore data in an easy, intuitive, and free-flowing manner. Interactive

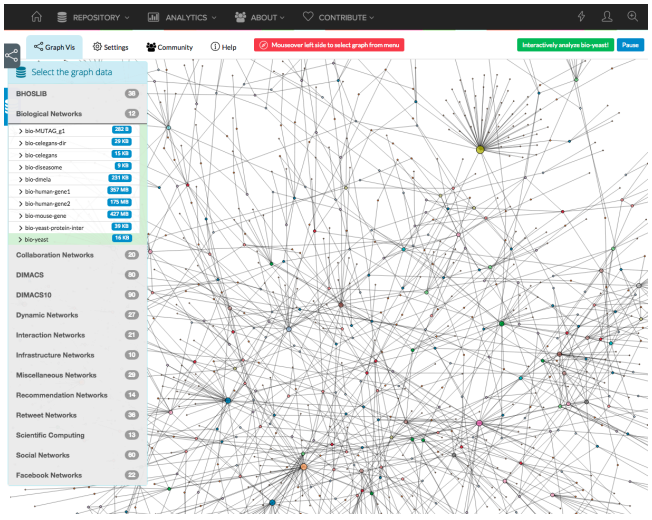


Figure 1: Users can interactively explore the topology of graphs in NetworkRepository.com and discover valuable insights.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

data repositories combine interactive visualizations with analytic techniques to reveal important patterns and insights for sense-making, reasoning, and decision-making. In addition, they facilitate research, education, training, and scientific discovery. These repositories allow the user to explore a single data set, or compare and contrast multiple data sets. In particular, interactive data repositories integrate visual analytic tools, which give the user full control to explore and understand the data in real-time. Data can be explored and visualized through user-defined and free-flowing transformations, queries, filtering, among other possibilities. We posit that the proposed interactive data repository concept will replace existing data repositories that have been used for decades.

We argue that interactive data repositories will significantly speedup scientific progress and discovery by making data accessible, but more importantly, by making data more discoverable, interpretable, and reusable. This would provide the broader scientific community with tools to quickly validate research findings, helping the peer-review process, and understand the caveats of published approaches based on the data and its characteristics. These tools would make it easier and more intuitive to explore the data in real-time, without the overhead of downloading the data, formatting, writing code/loading it, among others.

The remainder of this paper is organized as follows: Section 2 proposes an interactive data repository for graphs and provides a prototype, whereas Section 3 investigates independent and identically distributed (IID) data. Finally, Section 4 concludes.

2. INTERACTIVE GRAPH REPOSITORY

This section discusses the design of an interactive data repository for graphs (a.k.a relational data, networks), where the nodes represent entities (e.g., objects, people) and the links represent the dependencies among them. Graphs arise as a natural data representation, and allow us to study phenomena in a variety of domains, including social, behavioral, biological, transportation, communication, and financial domains. Studying these real-world graphs is crucial for solving numerous problems that lead to high-impact applications. For example, identifying the behavior and interests of users in online social networks (e.g., viral marketing, online advertising), monitoring and detecting virus outbreaks in human contact networks, predicting protein functions in biological networks, and detecting anomalous behavior in computer networks.

For demonstration, we discuss Network Repository (NR[1]) — the first graph data repository with a web-based interactive platform for real-time graph analytics. NR has hundreds of graphs for users to download and share. However, the key factor that differentiates NR from other repositories [12, 14] is the interactive graph analytics and visualization platform.

Network repository aims to improve and facilitate the scientific study of graphs by making it easy to interactively explore, visualize, and compare a large number of graphs across many different dimensions and facets. NR currently has 500+ graphs from 19 general collections (social, information, and biological networks, among others) that span a wide range of types (e.g., bipartite, temporal) and domains (e.g., social science, physics, biology). In addition to exploring the data in the repository, we also make it easy for users to upload and quickly explore and visualize their own data using the platform.

Next, we discuss some of the key features that that differentiate NR from other repositories.

2.1 Interactive Graph Topology Visualization

The interactive platform gives users the unique ability to interactively explore and visualize the topology of graphs in seconds. Figure 1 demonstrates this feature, where users have the flexibility to visualize any graph in the repository by simply selecting it from the left menu. The left menu displays a variety of graph collections which users can then click to display all graphs in a given collection. Once a graph is selected, we can then get a global view of the structural patterns by zooming-out completely. Similarly, users can drill-down on the regions of the graph that are of interest. For instance, suppose a user is interested in large cliques, then after spotting such regions from the global view, they can zoom into these regions to obtain additional information on the members of the clique and their connections and graph characteristics.

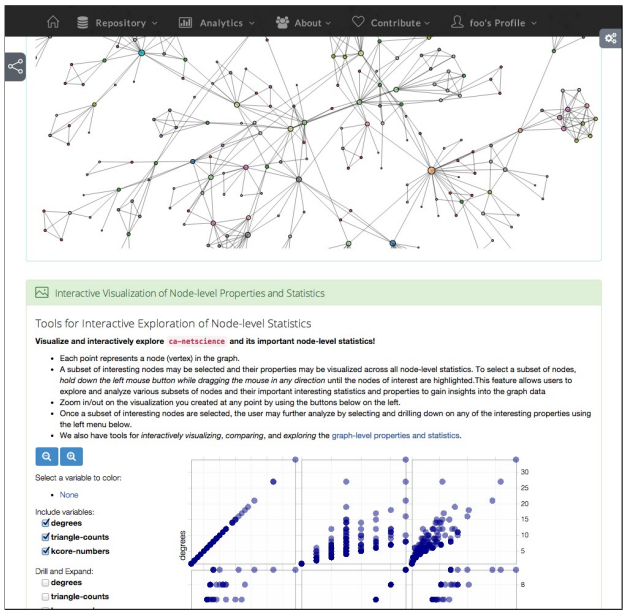


Figure 2: A snapshot of the online page of the network science coauthorship graph (ca-netscience), showing the interactive graph visualization and scatter plot. Note that each graph is automatically processed and assigned a unique URL for reference. This URL makes it easy for others to download the exact data, but also contains documentation and metadata, as well as numerous interactive visualization tools, graph statistics, as well as node-level statistics and distributions.

2.2 Multi-scale Interactive Graph Analytics

In order to provide the most flexibility for exploring data, NR provides a multi-scale graph analytics engine. This allows for each graph property to be easily analyzed at various levels of granularity and aggregation, which leads to a large space of possibilities for exploring and querying the data.

[1] http://networkrepository.com

Such an approach has many other advantages beyond providing users with a large space of possibilities for exploring and querying the data. In particular, NR provides an intuitive and meaningful approach that facilitates exploring and understanding graphs and their structure, both at the global macro-level as well as the local micro-level. For instance, at the global macro-level, NR maintains a number of global graph statistics and properties (e.g., total number of triangles, average clustering coefficient, max k-core number, etc). Alternatively, NR uses node-level (link-level) graph properties to explore graphs at the local micro-level.

In addition, the multi-scale analytics engine leverages visual analytics tools that facilitate graph exploration. For example, an interactive scatter plot matrix to analyze the correlation between pairs of node/link statistics (see an example in Figure 2), which supports brushing to allow users to highlight interesting nodes (and links) across the various measures. Furthermore, semantic zooming can be used to drill-down in order to understand the differences between individual nodes and links.

Further, NR leverages node and link summarization techniques (e.g., binning/histograms, statistical distributions) to obtain fast, meaningful and useful data representations. For instance, NR provides interactive plots of the cumulative distribution function (CDF) and the complementary CDF for important graph properties (e.g., degree distribution). These are known to be important for networks, capturing interesting structural properties such as heavy-tailed distributions (see an example in Figure 3).

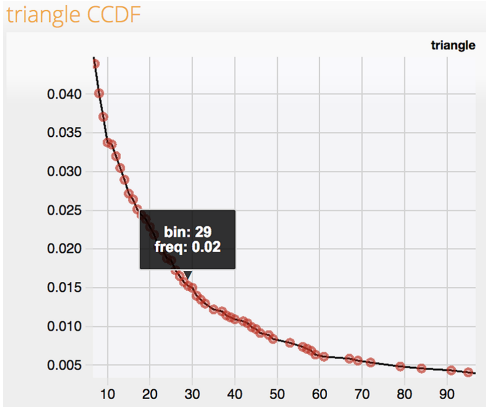


Figure 3: Interactive plot of the triangle count complementary cumulative distribution function (CCDF)

2.3 Interactive Graph Search & Comparisons

Graphs are easily compared across a wide range of important and fundamental graph statistics and properties (e.g., max k-core number, total number of triangles, degree, max clique size, motif counts, etc.). Figure 4 demonstrates how graphs can be interactively compared using an interactive scatter plot matrix and gives intuition for the types of queries and questions that can be explored. Clearly, as we show in Figure 4, there is a collection of data points where each point represents a graph, and users can use brushing to filter graphs via any user-selected constraint(s) and then highlights all such graphs (or nodes/edges) that satisfy it across all other interactive plots. In essence, NR supports interactive techniques such as brushing, linking, highlighting, as well as semantic zooming, to give the user full control to explore, query, and compare large collections of graphs across many dimensions. Finally, NR provides search tools to search for graphs by keywords and types.

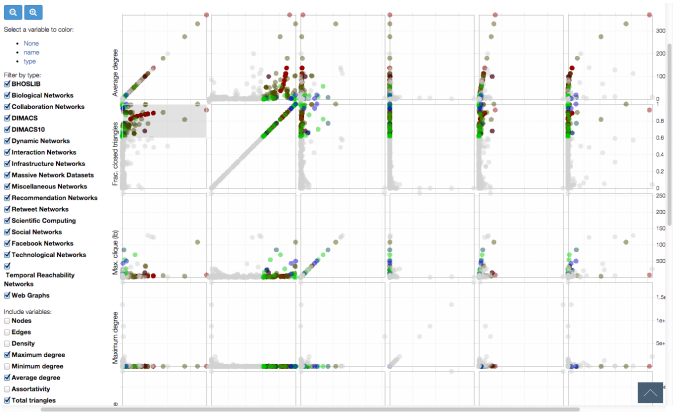


Figure 4: Interactive scatter plot matrix for large-scale graph comparisons. Interactively comparing graphs across a wide range of fundamental graph properties. Each data point represents a graph and each unique color represents the graph collection (e.g., social networks). In this example, we filter all graphs that have a global clustering coefficient (κ) greater than 0.6. Thus, all graph datasets that satisfy this query are highlighted in all other interactive plots. Further queries and research questions may be explored using this set of graphs that satisfy $\kappa \geq 0.6$.

Attribute	Range	Mean	Mode	Median	Variance	Skewness	Kurtosis	Median Dev.	Mean Dev.	Coeff. Var.	Missing values
sepal length	3.6	5.84	5	5.8	0.68	0.31	-0.57	0.68	0.69	0.14	0
sepal width	2.4	3.06	3	3	0.19	0.32	0.18	0.33	0.34	0.14	0
petal length	5.9	3.76	-	4.35	3.1	-0.27	-1.4	1.49	1.56	0.47	0
petal width	2.4	1.2	0.2	1.3	0.58	-0.1	-1.34	0.64	0.66	0.63	0

Figure 5: Univariate statistics are updated on-the-fly after any data filtering or querying/selection.

2.4 Scalability & Big Data Considerations

Big graph data may also be interactively explored and visualized using NR. We don't just provide users with summary or graph-level statistics, but allow a much deeper exploration of the data while sending a significantly smaller amount of data. For instance, users can interactively explore a range of distributions from a wide variety of important graph properties and statistics. Whenever necessary, we utilize state-of-the-art graph sampling methods to ensure fast and efficient loading and processing of the data while being as accurate as possible [1]. These techniques are extremely effective for sampling node features and visualizing the structure and connectivity of the graphs.

Furthermore, at the heart of the interactive platform lies a high-performance parallel graph analytics engine, which is written in C/C++ and designed to be fast and scalable for extremely large graphs. We note that it outperforms other

libraries such as GraphLab and igraph (e.g., on triangle and motif counting).

3. STATISTICAL VISUAL ANALYTICS

This section discusses the design of interactive data repositories for IID data. Since visual analytic techniques for iDR largely depend on the scientific discipline, we discuss general guidelines and provide examples from a recent visual analytic platform for such data.

Interactive univariate analysis offers a quick assessment of a variable (e.g., see Figure 5 for point statistics). Further, Figure 6 provides interactive box-and-whisker plots, histograms, outlier detection/visualization, etc. To quantify the relationship between two variables, one may use bivariate point statistics (e.g., correlation coeff. denoted by r in Figure 6). A variety of visual bivariate analytic techniques are shown in the lower-triangular region of Figure 6. In particular, interactive scatter plots, loess curves (non-parametric, non-linear) [4], and regression lines. Categorical variables may be used to color the data points in each scatter plot as well as the loess curves and regression lines.

It is also important to provide interactive multidimensional analytic techniques to understand the relationships between variables simultaneously (e.g., the interactive scatter plot matrix in Figure 6 with brushing and linking).

Interaction techniques such as brushing, linking, zooming, panning, filtering are used heavily in iDR. All data normalization and transformations in Figure 6 are interactive, rapid, incremental, and reversible.

4. CONCLUSION

This paper revisits existing scientific data repositories, and instead, proposes the concept of an *interactive data repository* that aims to facilitate scientific progress by incorporating interactive visual analytic techniques for the exploration, mining, and understanding of data in real-time. The paper also discusses a prototype of interactive data repositories for both graph data and independent and identically distributed (IID) data.

Acknowledgments

We thank the reviewers for helpful comments and all of the donors who contributed data to the repository.

5. REFERENCES

[1] N. K. Ahmed, J. Neville, and R. Kompella. Network sampling: From static to streaming graphs. *Transactions on Knowledge Discovery from Data (TKDD)*, 8(2):7:1–7:56, June 2014.

[2] N. K. Ahmed and R. A. Rossi. Interactive visual graph analytics on the web. In *ICWSM*, pages 566–569, 2015.

[3] A. Brazma, H. Parkinson, U. Sarkans, M. Shojatalab, J. Vilo, N. Abeygunawardena, E. Holloway, M. Kapushesky, P. Kemmeren, G. G. Lara, et al. Arrayexpress–a public repository for microarray gene expression data at the EBI. *Nucleic acids research*, 31(1):68–71, 2003.

[4] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979.

[5] D. Ebert, K. Gaither, Y. Jang, and S. Lasher-Trapp. Cross-scale, multi-scale, and multi-source data visualization and analysis issues and opportunities. In *Scientific Visualization*, pages 353–360. 2014.

[6] P. Murphy and D. W. Aha. UCI repository of machine learning databases–a machine-readable repository. 1995.

[7] NSB. Digital research data sharing and management. Technical Report NSB-11-79, National Science Foundation, 2011.

[8] N. A. of Sciences. *Ensuring the Integrity, Accessibility, and Stewardship of Research Data in the Digital Age*. The National Academies Press, 2009.

[9] H. Pashler and E.-J. Wagenmakers. Editor's introduction to the special section on replicability in psychological science a crisis of confidence? *Perspectives on Psychological Science*, 7(6):528–530, 2012.

[10] R. A. Rossi and N. K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[11] R. A. Rossi and N. K. Ahmed. Networkrepository: An interactive data repository with multi-scale visual analytics. In *arXiv:1410.3560v2*, 2015.

[12] SNAP. Stanford network dataset collection. http://snap.stanford.edu/data/index.html.

[13] J. J. Thomas, K. Cook, et al. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26(1):10–13, 2006.

[14] UCI ML Repository. UCI machine learning repository. http://archive.ics.uci.edu/ml.

[15] M. C. Whitlock, M. A. McPeek, M. D. Rausher, L. Rieseberg, and A. J. Moore. Data archiving. *The American Naturalist*, 175(2):145–146, 2010.

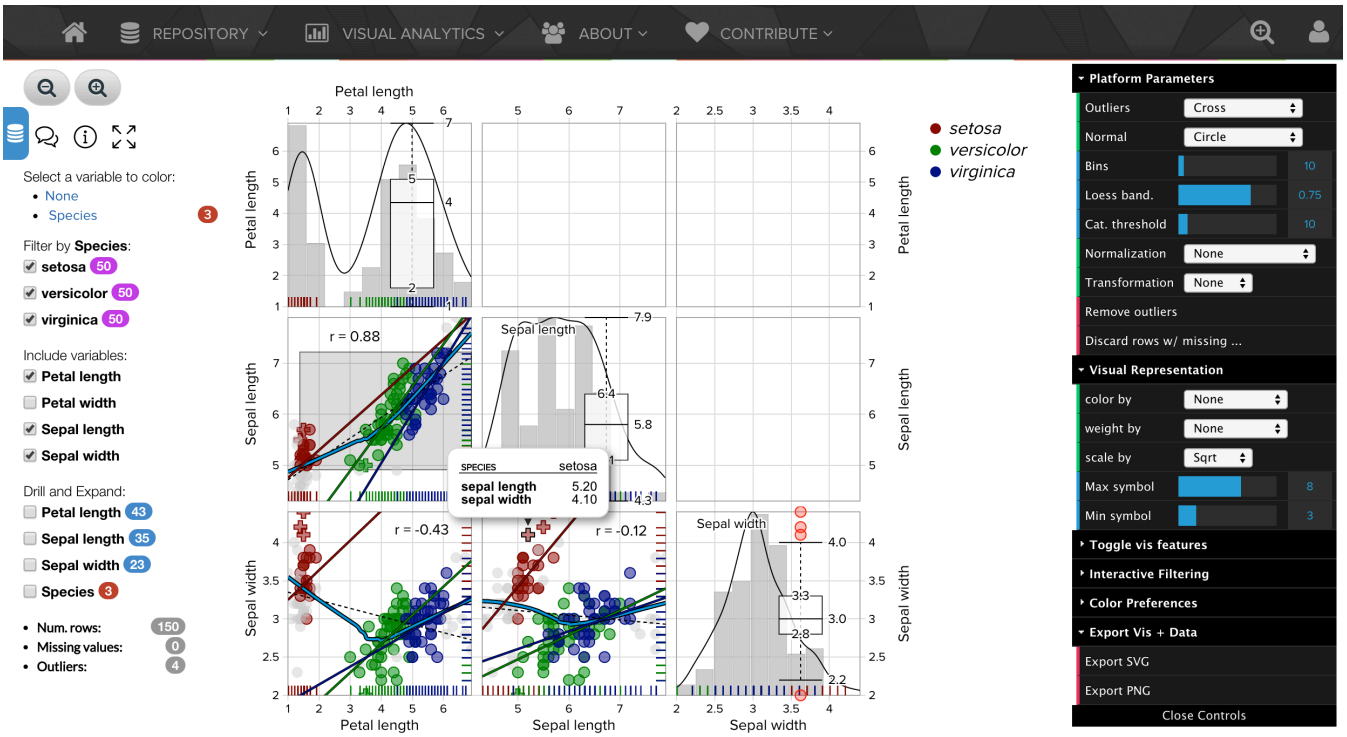


Figure 6: Interactive scatter plot matrix. The screenshot above is of the iris data where data points are colored by species. The lower triangular shows the relationship between pairs of variables, whereas the diagonal provides univariate analysis (e.g., histograms, whisker plots, and outliers).

Author Index