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

Proceedings of the
ACM SIGKDD 2017 Full-day Workshop on

Interactive Data Exploration and Analytics

IDEA 2017

Halifax, Nova Scotia, Canada
August 14, 2017
poloclub.gatech.edu/idea2017

Microsoft Research
200 YEARS GHENT UNIVERSITY
Georgia Tech
max planck institut informatik
UNIVERSITÄT DES SAARLANDES
Universiteit Leiden
THE HEBREW UNIVERSITY OF JERUSALEM
Carnegie Mellon University

Chicago 2013

New York 2014

Sydney 2015

San Francisco 2016

Designed by: Fred Hohman

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

These proceedings are not included in the ACM Digital Library.

IDEA'17, August 14, 2017, Halifax, Nova Scotia, Canada.

ACM SIGKDD Workshop on Interactive Data Exploration and Analytics

General Chairs

Duen Horng (Polo) Chau (Georgia Tech)
Jefrey Lijffijt (Ghent University)
Jilles Vreeken (Max Planck Institute for Informatics and Saarland University)
Matthijs van Leeuwen (Universiteit Leiden)
Dafna Shahaf (Hebrew University of Jerusalem)
Christos Faloutsos (Carnegie Mellon University)

Program Committee

Acar Tamersoy (Symantec, USA)
Adam Perer (IBM, USA)
Aristides Gionis (Aalto U., Finland)
Bahador Saket (Georgia Tech, USA)
Bo Kang (Ghent U., Belgium)
Danai Koutra (U. Michigan, USA)
Emilia Oikarinen (Finnish Institute of Occupational Health, Finland)
Esther Galbrun (Loria, France)
Geoff Webb (Monash U., Australia)
George Forman (Amazon, USA)
Hannah Kim (Georgia Tech, USA)
Jaegul Choo (Georgia Tech)
James Abello (Rutgers, USA)
Jia-Yu (Tim) Pan (Google, USA)
Kai Puolamäki (Finnish Institute of Occupational Health, Finland)
Kashyap Popat (Max Planck Institute for Informatics, Germany)
Kevin Roundy (Symantec, USA)
Mario Boley (Max Planck Institute for Informatics, Germany)
Marti Hearst (UC Berkeley, USA)
Minsuk (Brian) Kahng (Georgia Tech, USA)
Nan Cao (NYU Shanghai, China)
Parikshit Ram (Skytree, USA)
Pauli Mietinnen (Max Planck Institute for Informatics, Germany)
Robert Pienta (Georgia Tech, USA)
Saleema Amershi (Microsoft Research, USA)
Steffen Koch (U. Stuttgart, Germany)
Stephan Günnemann (TU Munich, Germany)
Sucheta Soundarajam (Syracuse U., USA)
Thomas Gärtner (U. Nottingham, UK)
Thomas Seidl (LMU Munich, Germany)
Tijl De Bie (University of Bristol, UK)
Wouter Duivesteijn (U. Ghent, Belgium)

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 explanatorily 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, 2014, 2015, and 2016 the main program of IDEA'17 consists of twelve papers that cover various aspects of interactive data exploration and analytics. In addition there were three keynotes. Six papers were presented orally, and six were presented during the interactive poster and demo session. These papers were selected from a total of 20 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 Rich Caruana, Nathalie Riche, and Samuel Kaski for engaging keynote presentations on the fundamental aspects of interactive data exploration, analysis, and visualization.

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

Saarbrücken, July 2016

Table of Contents

Invited Talk

Interactive Machine Learning via Transparent Modeling: Putting Experts in the Driver's Seat

Rich Caruana
Deep Learning Foundations
Microsoft Research
rcaruana@microsoft.com

Abstract

In machine learning often a tradeoff must be made between accuracy and intelligibility: the most accurate models usually are not very intelligible (e.g., deep nets, boosted trees, and random forests), and the most intelligible models usually are less accurate (e.g., linear or logistic regression). This tradeoff often limits the accuracy of models that can be used in mission-critical applications such as healthcare where being able to understand, validate, edit, and ultimately trust a learned model is important. We have developed a learning method based on generalized additive models called GA2Ms that is often as accurate as full complexity models, but as intelligible as linear/logistic regression models. GA2Ms not only make it easy to understand what a model learned and how it makes predictions, but it also makes it easier to edit the model when it learns "bad" things. These bad things typically arise not because the learning algorithm is wrong, but because the data has unexpected "landmines" hidden in it. Making it possible for experts to understand a model and interactively repair it is critical for safe deployment because most data has such landmines. In the talk I'll present cases studies where these transparent, high-performance GAMs are applied to problems in healthcare and recidivism prediction, and explain what we're doing to make the models easier for experts to understand and edit.

Bio

Rich Caruana is a Senior Researcher at Microsoft Research. Before joining Microsoft, Rich was on the faculty in the Computer Science Department at Cornell University, at UCLA's Medical School, and at CMU's Center for Learning and Discovery. Rich's Ph.D. is from Carnegie Mellon University, where he worked with Tom Mitchell and Herb Simon. His thesis on Multi-Task Learning helped create interest in a new subfield of machine learning called Transfer Learning. Rich received an NSF CAREER Award in 2004 (for Meta Clustering), best paper awards in 2005 (with Alex Niculescu-Mizil), 2007 (with Daria Sorokina), and 2014 (with Todd Kulesza, Saleema Amershi, Danyel Fisher, and Denis Charles), co-chaired KDD in 2007 (with Xindong Wu), and serves as area chair for NIPS, ICML, and KDD. His current research focus is on learning for medical decision making, transparent modeling, deep learning, and computational ecology.

Invited Talk

Interactive Visual Interfaces to Think with Data

Nathalie Henry Riche
Extended Perception, Interaction & Cognition
Microsoft Research
nath@microsoft.com

Abstract

In this talk, I will reflect on eight years of research on the design of interfaces that help people **explore complex data** by representing information visually and providing interaction mechanisms to extract meaningful patterns. In particular, I will present insights gained from a multi-year collaboration with neuroscientists to help them understand dynamics in brain connectivity networks. In the second part of the talk, I will present research on the design of interfaces that help people **think with data** by seamlessly bridging data exploration and visual thinking via pen and touch interactions. I will conclude by reflecting on challenges and opportunities for the future.

Bio

Nathalie conducts research at the intersection of data visualization and human-computer interaction. She currently is a researcher at Microsoft Research member of the EPIC (Extended Perception Interaction Cognition) group led by Ken Hinckley and part of the broader HCI@MSR effort. Nathalie received a joint Ph. D. in Computer Science in 2008 from the University of Paris-Sud/ INRIA, France and the University of Sydney, Australia. Over the past ten years, she published over 50 refereed journal and conference articles in human-computer interaction and data visualization venues, receiving several outstanding awards for her contributions to the field. Nathalie is involved in program and organization committees of leading venues in visualization and is co-chairing the dedicated visualization sub-committee of the premier conference in human-computer interaction ACM SIGCHI in 2017 and 2018.

Invited Talk

Interactive Intent Modeling

Samuel Kaski
Aalto University, and Helsinki Institute for Information Technology (HIIT)
samuel.kaski@aalto.fi

Abstract

I will discuss our recent work on interactive machine learning in two closely related setups: (i) interactive intent modeling for information discovery and (ii) knowledge elicitation on features for improving predictive modelling given limited high-dimensional data. Both setups require balancing between exploration and exploitation in the interactions, and interactive modelling of the user which can be formulated as experimental design or multiarmed bandit problems. I will also discuss extensions to multimodal interfaces, including mind reading, and to inferring more advanced cognitive models from data with Approximate Bayesian Computation.

Bio

Samuel Kaski is an Academy (research) Professor of the Academy of Finland, Professor of Computer Science at Aalto University, and Director of the Finnish Center of Excellence in Computational Inference Research COIN. His field is probabilistic machine learning, with applications involving multiple data sources in interactive information retrieval, data visualization, health and biology.

Visualizing Wikipedia for Interactive Exploration

Ron Bekkerman
University of Haifa
Haifa, Israel
ronb@univ.haifa.ac.il

Olga Donin
University of Haifa
Haifa, Israel
olgad@univ.haifa.ac.il

ABSTRACT

We aim to visualize (almost) the entire Wikipedia as a two-level coarse-grained / fine-grained graph representation of Wikipedia categories, for which we customize a hierarchy. We face the challenge of visualizing large scale-free graphs and propose an effective method for edge elimination that preserves the topical locality property of the original graph. The resulting visualization is sensible, traversable, and therefore actionable. It is a big step towards establishing comprehensiveness of Wikipedia as the collective memory of our and future generations.

KEYWORDS

Data Visualization, Interactive Exploration, Wikipedia

ACM Reference format:
Ron Bekkerman and Olga Donin. 2017. Visualizing Wikipedia for Interactive Exploration. In *Proceedings of KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada, August 14th, 2017 (IDEA'17),* 7 pages.
DOI:

INTRODUCTION

Wikipedia has de-facto become the collective memory of our generation [10, 21]. Our ancestors did not have the luxury of accessing a comprehensive memory bank. Over generations, people were considered intellectuals if they remembered a variety of facts and had a mental ability to integrate them into a compelling story [16]. We no longer need to develop a strong declarative memory. The classic model of human intellect [8] is to be adjusted to the new reality when the memory retention operation is effectively "outsourced" to the Web, and to Wikipedia specifically. Facts are – from now on – always at the tips of our fingers. And, remarkably, the content of our new outsourced memory is roughly the same for everyone. It is safe to say that the humankind is developing a collective intellect as our cognition is now based on the common, shared memory source.

There are many advantages of the collective memory as represented in Wikipedia. First, it never fails on us (as soon as, naturally, the Wikipedia website is accessible). We can always retrieve a missing fact, provided that we remember what to search for. Admittedly, Wikipedia is being constantly changed, some pages deleted while new pages added, the content of others updated. However, Wikipedia is never fading as human memory is. We can retrieve the same fact twice, many years apart, and chances are good that the fact will not change, regardless of our physical and mental wellbeing. Moreover, an argument can be made that Wikipedia is updating more slowly than the human memory is fading. For all practical purposes, our new collective memory is pretty static.

Second, in contrast to our biological memory that always plays tricks on us, Wikipedia is not changing inadvertently. Wikipedia pages are being added and deleted for a good reason, which is to always improve the content quality. Wikipedia is known for tending to objectiveness – opinionated reasoning is being aggressively fought against. Actually, the notion of objectiveness is very new in the context of human memory – we are never objective in our choice of facts to remember, nor we are able to keep our memories unaffected by our attitude towards them. Wikipedia, however, is widely considered unbiased [18], and facts presented in Wikipedia are perceived as correct. Indeed, they are verified by a community of highly qualified editors. While pure objectiveness cannot be possibly achieved, Wikipedia might be the most objective source of information that the humanity has ever had access to.

Third, and probably foremost, there is nothing mysterious about Wikipedia. While human memory has not been fully researched and some biological processes in our brains are yet to be understood, Wikipedia is just a few (million) pages in the Web that are – conceptually – trivial to grasp. Wikipedia pages hyperlink each other so its underlying structure is a graph [28], which we – computer scientists – are intimately familiar with. And whoever believes that a graph with a few million nodes is too large should not forget that they carry a graph of about 100 billion neurons to the north of their neck.

Being a conceptually simple notion, Wikipedia as our new digital memory allows answering questions that would sound completely outrageous were they asked about the human memory. One of the most exciting questions is comprehensiveness: does Wikipedia contain all the world's knowledge? Needless to say, asking such a question would make no sense in the context of human memory – no one would doubt its selectivity. A skeptical reader would argue that the lack of comprehensiveness characterizes Wikipedia just as well as the human memory. The proof might be straightforward: it is enough to come up with an example of a piece of knowledge that Wikipedia lacks. We, however, would like to offer two counterarguments. First, not every piece of knowledge has to be included in the world's collective memory. In fact, Wikipedia editors meticulously assess the value of each piece of knowledge to be presented on Wikipedia pages. Information that might not be in the general public interest is cold-bloodedly erased. This does not necessarily jeopardize the comprehensiveness of Wikipedia as knowledge can be effectively summarized to obfuscate auxiliary details.

Our second counterargument is: given a specific piece of knowledge, how does one know that this knowledge is not already in

IDEA'17, Halifax, Nova Scotia, Canada

DOI:

Wikipedia? We are used to applying keyword search to document repositories such as Wikipedia, but knowledge is not always easy to describe in a few keywords. Even if we applied many searches of many keywords, and did not find anything, would this mean that the knowledge we were looking for is not in Wikipedia or our search methodology is just not good enough? Based on the two arguments above, we may conclude that Wikipedia comprehensiveness is not that easy to contradict. Apparently, it is not easy to prove either.

Some work has been done on assessing comprehensiveness of several topics in Wikipedia, by mapping topical pages on a set of books published on the topic [9]. An attempt was made of assessing comprehensiveness of the entire Wikipedia [23], however the proposed methodology did not go far beyond word frequency computations.

To claim comprehensiveness or lack of comprehensiveness, one needs to first understand what it is out there in Wikipedia. What does our collective memory actually contain? When referred to memory, this question sounds both lunatic and thrilling at the same time. On the one hand, no one has dared to overview the entire content of memory. This would be impossible in the context of human memory which is an ever-changing, intrinsically complex, only partly studied medium. Even in the context of (English only) Wikipedia, this question is hard to answer. On the other hand, once answered, this question may lead to a breakthrough in a global understanding of our intellectual and cultural heritage which we (and our children) are substituting for our long-term memory. So, what does Wikipedia know? That is the question that we aim to answer in this paper.

METHODOLOGY

We will show how to climb 30,000 feet and view (almost) the entire content of Wikipedia in a digestible and actionable format. After exploring the content, we will be able to make decisions about which topics are missing in Wikipedia, which are underrepresented, and how our efforts need to be allocated to make Wikipedia the ultimate source of truth in all areas of human interest.

At the time of our bulk download (September 9, 2016), English-language Wikipedia contained 16,857,586 pages out of which 7,785,959 were redirect pages, 162,236 were disambiguation pages, and 3,830,032 were auxiliary pages, such as pages of Wikipedia categories, files, templates etc. After removing redirect, disambiguation, and auxiliary pages, we ended up with 5,079,359 content pages. We are on a quest to summarize five million Wikipedia pages.

From the classical Text Classification perspective [17], summarizing five million pages is not too hard: each page can be automatically categorized to one of N categories. Once all the pages are categorized, we would be able to summarize the entire Wikipedia as a ranked list of categories sorted by their frequency: say, N_1 pages on the topic of chemistry, N_2 pages on politics, N_3 pages on arts, etc. There are a number of deficiencies in this approach: (a) categories have to be chosen beforehand and might not directly correspond to the topics covered in the data; (b) text classification is error-prone – some pages will be misclassified; (c) choosing too few categories will lead to coarse-grained, imprecise categorization, while choosing too many categories will overcomplicate the categorization algorithm which would result in a large amount of misclassifications.

Fortunately, most Wikipedia pages are already categorized by their contributors: at the time of creating a Wikipedia page, a set of relevant categories has to be provided. Out of the 5,079,359 content pages, 4,913,089 pages belong to at least one category. Unfortunately, the entire number of Wikipedia categories is 1,303,021 which is only four times less than the number of content pages. Overviewing those categories would be as tedious as overviewing Wikipedia pages themselves. Nevertheless, Wikipedia categories hold the aggregation property such that content pages can be overviewed in groups whenever the corresponding categories are considered.

Creating a ranked list of Wikipedia categories is not an ideal way of overviewing Wikipedia. A one-dimensional interface of the ranked list – while being intimately familiar to us from our everyday interactions with search engine results – suboptimally exploits the area of the computer screen, and misses the advantages of using visual primitives such as color and shape [7]. A two-dimensional, graph-based representation would be more plausible for overview and exploration purposes.

We build a graph of Wikipedia categories, with nodes being the categories themselves and the edges being the (weighted) semantic connections between the categories as captured on Wikipedia pages: in 83% cases, a Wikipedia page belongs to more than one category. The more pages belong both to category A and category B, the stronger the connection between A and B is. When spread over a two-dimensional surface, the graph of Wikipedia categories naturally holds the topical locality property [5]: similar categories will be shown close to each other – which will allow easy exploration. At this point, it appears that all we are left to deal with is the graph's enormous size.

Since the times Wikipedia first got measured [27], attempts were made to visualize Wikipedia. Holloway et al. [11] generated an image with 79 thousand Wikipedia categories – which at that time was the overall number of categories. Needless to say, given such an enormous number of represented categories, this visualization is not appropriate for exploration. Moreover, the number of categories has increased 16.5 times since then, which makes the visualization of all Wikipedia categories no longer feasible. Pang and Biuk-Aghai [20] proposed a Wikipedia visualization in the style of a geographical map, while not attempting to achieve the visualization comprehensiveness. Silva et al. [24] visualized small graphs of Wikipedia pages hyperlinking each other. Some previous works dealt with visualizing Wikipedia dynamics: Brandes et al. [3] visualized Wikipedia's edit network, Kimmerle et al. [14] visualized knowledge evolution in Wikipedia.

Wikipedia categories are power-law distributed over the pages, with a long tail of categories each covering very few pages. In fact, 64% Wikipedia categories cover 90% of Wikipedia content pages. We decided to ignore the long tail and to visualize only categories covering 90% of Wikipedia pages, however those categories are still too many to visualize. Literature offers a variety of methods for visualizing large graphs, by using techniques such as edge clustering [4], edge bundling [12], and edge compression [6]. We did not adapt those techniques due to their imprecision, high complexity, and low scalability. Instead, we got inspired by the wealth of research on visualization of hierarchical information (e.g. [22]). If we

impose a hierarchy on the Wikipedia categories, we could visualize the graph of top-level categories each covering a large number of pages, while each top-level category could in turn be visualized as a graph of second-level categories.

Let us emphasize the fact that we need to visualize only the top two levels of Wikipedia category hierarchy – because the overall number of categories to visualize is under one million. If the hierarchy is carefully designed, e.g. second-level categories are uniformly distributed among the top-level categories, at any time we may show a graph with under $\sqrt{1,000,000} = 1000$ nodes. This number is manageable in a visualization – both in terms of layout and explorability. In a real-world situation, however, the uniform distribution is too much to require. Nevertheless, the number of categories is not expected to grow fast beyond a million, so the two-level hierarchy design will hold water years from now.

As a matter of fact, Wikipedia already offers a category hierarchy: most category pages are themselves listing one or more categories. However, Wikipedia category hierarchy is extremely noisy and not appropriate for visualization. Consider, for example, category *"BioShock"* which is a first-person shooter video game series. Traversing one path of the Wikipedia category hierarchy from *"BioShock"* upwards, we can see the following categories: *"BioShock"* → *"Dieselpunk"* → *"Retro Style"* → *"Nostalgia"* → *"Melancholia"* → *"Romanticism"* → *"German Idealism"* → *"Rationalism"* → *"A priori"* → *"Latin Logical Phrases"* → *"Latin Philosophical Phrases"* → *"Latin Words and Phrases"* → *"Ancient Rome in Art and Culture"* → *"Culture in Rome"* → *"Tourism in Rome"* → *"Rome"* → *"Renaissance Architecture in Lazio"* → *"Italian Renaissance"*. Apparently, the Wikipedia hierarchy is not a hierarchy but rather a network of associations. The longest path we could detect in this graph is of the length of 881. Besides, we detected 32,678 cycles in the graph, the shortest being of length 2, the longest – 829.

It is clear that we need to construct the category hierarchy of our own. We consulted with Kittur et al. [15] who mapped all 277 thousand Wikipedia categories of that time to 26 top-level categories, and then used the top-level categories to overview the content of Wikipedia. While Kittur et al.'s result is the closest to ours, we find it too coarse-grained, not explorable, and therefore not actionable. Milne and Witten [19] present a visual tool for analyzing Wikipedia which is, in contrast, suitable for exploration but too fine-grained: it does not provide an overview of Wikipedia. Suchecki et al. [25] investigated the evolution of Wikipedia category structure and concluded that it is quite stable, which implies that our results are unlikely to become obsolete any time soon.

We preprocessed the set of Wikipedia categories by first removing "technical" categories (that auxiliary pages belong to), such as categories containing the following phrases: *"Archived"*, *"COI-Bot"*, *"Created"*, *"Defunct"*, *"Deprecated Parameters"*, *"Did You Know"*, *"Disambiguation"*, *"Draft"*, *"DYK"*, *"Infobox"*, *"Lists of"*, *"Missing"*, *"Navigational Boxes"*, *"Nominations"*, *"Redirects"*, *"Requests"*, *"Templates"*, *"Uncertain"*, *"Unknown"*, *"Wikipedia"*, and *"Wikiproject"*. We also removed 70 noisy categories (categories in foreign languages, personal names, etc). Examples of noisy categories are: *"Living People"*[1], *"Births"*, *"Deaths"*, *"Nacional"*, and *"Michael"*. We mapped all plural words onto their singular forms. We then manually added 9 aggregation rules for all categories belonging to *"US States"*, *"UK Counties"*, *"Canada Provinces"*, *"India States"*, *"Countries"*, *"Towns"*, *"Years"*, *"Centuries"*, and *"National"* (into the latter, we aggregated nationality categories, such as *"German"*, *"Brazilian"*).

We are now ready to build the hierarchy of Wikipedia categories. For a category A, we denote $W(A)$ the set of words in the category's name. We create the category hierarchy as follows: category A is included in a more general category A' if $W(A')$ is a proper subset of $W(A)$. The resulting hierarchy is a DAG – circles are not allowed by definition. The depth of the constructed hierarchy is 6. An example of a depth-6 hierarchy is: *"College of Charleston Cougars Women's Basketball Players"* → *"College of Charleston Cougars Women's Basketball"* → *"College of Charleston Cougars Basketball"* → *"College of Charleston Cougars"* → *"College of Charleston"* → *"College"*.[2]

The top level of the category hierarchy contains 441 largest categories covering 90% of the entire Wikipedia. Those categories will be the nodes in our top-level graph representation. If two categories appear together on at least one Wikipedia page, we connect them with an edge. We end up having 68,764 edges in the top-level graph – the number that is way beyond the boundaries of aesthetic appeal. Besides the problem of the enormous number of edges, we face another problem: the graph is scale-free.

Visualization of scale-free graphs is difficult. In the majority of cases, the graph looks like an image of an explosion whose epicenter is a tangled bundle of edges with many separate branches sticking out of it in all possible directions. The larger the epicenter is, the messier the graph appears. To our surprise, literature on visualizing scale-free graphs is very sparse (see e.g., [13, 26]). Accepted approaches are mostly related to stochastic edge sampling, which does not really solve the aesthetics problem if the sample is large, while breaking the graph to disconnected parts if the sample is small. We propose a different technique for eliminating unnecessary edges.

As a preprocessing step, we need to eliminate low-weight edges (edges between categories that rarely appear together on Wikipedia pages). Unfortunately, in a scale-free graph of categories, the two endpoints of an edge might have dramatically different coverage, such that the number of pages on which they appear together can be negligible for one and substantial for another. A standard approach of using a universal threshold to filter out low-weight edges is therefore not applicable in this case. We eliminate an edge between two categories if they appear together on less than 5% of pages covered by either of them. The motivation for this choice is that the eliminated edge needs to be negligible for both its endpoints. For example, let us say that category A covers 100 pages, and category B covers 10,000 pages. Say, A and B appear together on 4 pages, which is 4% of A's coverage, and 0.04% of B's coverage. We eliminate the edge between A and B because it is negligible for both nodes. For the top-level category graph, applying this heuristic led to eliminating 93% of edges. Still, the remaining 4815 edges are too many for an aesthetic visualization.

We noticed that both the top-level graph and second-level graphs contain many triangles. Triangles tangle nodes while creating extra

[1] *"Living People"* is the largest category in Wikipedia, covering over 786 thousand pages. It is simply too common to be meaningful.

[2] Despite its apparent superiority over the existing Wikipedia category hierarchy, our hierarchy is not 100% error-proof. For example, the category *"Ambassadors of the United Kingdom to the Ottoman Empire"* was identified as a subcategory of *"Ambassadors of the Ottoman Empire"*.

ties between them. If we break each triangle by eliminating one of its edges, the distance between two previously adjacent nodes will then be 2, which will still preserve the topical locality property. The remaining question is which edge out of the three edges of a triangle we need to eliminate. The power-law distribution of node degrees in a scale-free graph naturally splits up to the head, body, and tail. Nodes from the distribution's head are connected to many others, while nodes from the tail are connected to very few, with the body nodes staying in between. For simplicity, the sets of nodes belonging to the head, body, and tail of the degree distribution will be called the first layer, second layer, and third layer, respectively. Inspired by the Hamiltonian ball model of Asratian and Oksimets [1], we propose the following algorithm for eliminating triangles in scale-free graphs:

(1) Eliminate edges that connected nodes of the same layer.
(2) Eliminate edges between nodes of the first and third layer.
(3) If the process above resulted in isolating nodes, restore one (arbitrary) edge per such node.

The logic behind this algorithm is in taking into account only connections between the first and the second layers, as well as between the second and the third layers. All the other edges would not matter: nodes of the second layer are likely to be connected to each other through the nodes of the first layer, while each node from the third layer is likely to be connected at least one node from the second layer (and if not, a connection will be kept to one node from the first or third layer).

Theorem 0.1. *The algorithm proposed above eliminates all triangles in the graph.*

Proof. Assume a triangle remained in the resulting graph. According to step 1 of the algorithm, there cannot be two nodes of the triangle that belong to the same layer. Thus, the only option for the triangle to exist would be when each of its nodes belongs to a different layer. However, according to step 2 of the algorithm, the resulting graph does not contain edges drawn from layer 1 to layer 3, which means that the triangle with nodes at each of the three layers is not possible. Edges restored at step 3 of the algorithm increase node degrees from 0 to 1, which implies that those nodes cannot participate in any triangle. □

Figure 1 is an example of a subgraph from the top-level graph before and after applying the triangle elimination graph – clearly, the resulting graph is more comprehensible. After applying the algorithm to the top-level graph, we eliminated 60% edges – and all 19,412 triangles. The distance between two previously adjacent nodes became 2.1 on average (that is, the topical locality of the graph is almost fully preserved).

RESULTS

The resulting visualization of the top-level graph is in Figure 2. All visualizations are obtained using the Gephi graph visualization tool with Fruchterman-Reingold rendering preprocessed by Force Atlas [2]. Larger nodes correspond to categories with higher coverage. As can be seen in Figure 2, the top-level categories split to four large groups: *Science and Society* (including history, religion, and technology), *Arts and Culture* (including films and television), *Places and Nature* (including flora and fauna), and *Sports*, while some ambiguous categories are referred to as *Other*. Percentage-wise, *Science and Society* covers 32.7% of Wikipedia, *Arts and Culture* 25.6%, *Places and Nature* 76.7%, *Sports* 16.0%, and *Other* 24.4% (obviously enough, these topics heavily overlap). It is not a surprise that *Places and Nature* covers more than 3/4 of Wikipedia – the majority of Wikipedia pages are location-bound. What is more of a surprise is that as much as 1/6 of Wikipedia deals with sports.

Figure 3 shows four examples of visualizing the top-level categories as graphs of their subcategories. Analogously to the top-level, in the second-level visualizations we decided to present only the largest subcategories covering together at least 90% of the category's pages. The top graphs in Figure 3 show two large categories (*"Districts"* and *"Descent"*) with over a thousand subcategories each, while the bottom graphs show two small categories (*"Models"* and *"Gold"*) with under a hundred subcategories each.

Our edge elimination methodology (low-weight edge elimination + triangle elimination) split the *"Districts"* graph to many small subgraphs, each representing a separate type of a district. Many such subgraphs look like flowers – those often correspond to a specific country and its districts (the central category is global, such as *"Districts of India"*, while the peripheral categories cover local districts). In the case of *"Descent"*, the vast majority of categories shown are quite homogeneous in their meaning: they cover pages of people of a certain descent. In this situation, separation of the graph to smaller subgraphs is infeasible. Our edge elimination methodology can, however, substantially detangle the complex network of connections between people of various descents. In the resulting visualization, areas can be clearly identified that correspond to people of European, Asian, Hispanic, and Middle Eastern descent. *"Models"* is an ambiguous category that got split in our visualization to two main subgraphs: scientific models and fashion models, with the latter being significantly larger in size. Category *"Gold"* was split to many more subgraphs, the largest of which is related to gold medals in sports. The average number of nodes in the second-level visualizations is 321, the average coverage is 94%. The original number of edges (before edge elimination) is 7942 on average, it goes down to 1321 after the low-weight edge filtering, and down to 609 after applying our triangle elimination algorithm. Each edge eliminated by the algorithm became a path of length 2.4 on average.

The website[3] presents the interactive system where each top-level node from Figure 2 is clickable and once clicked it unfolds into the second-level visualization, examples of which are shown in Figure 3. We invite encyclopedians, library and information scientists, philosophers, and subject matter experts to use our visualization for assessing the comprehensiveness of Wikipedia. Underrepresented topics can now be identified, and additional content may be created. Content creation in overrepresented topics might be slowed down. If rebuilt periodically, our visualization can capture the dynamics of content creation, which may lead to defining the general strategy of maintaining Wikipedia as our main source of factual knowledge.

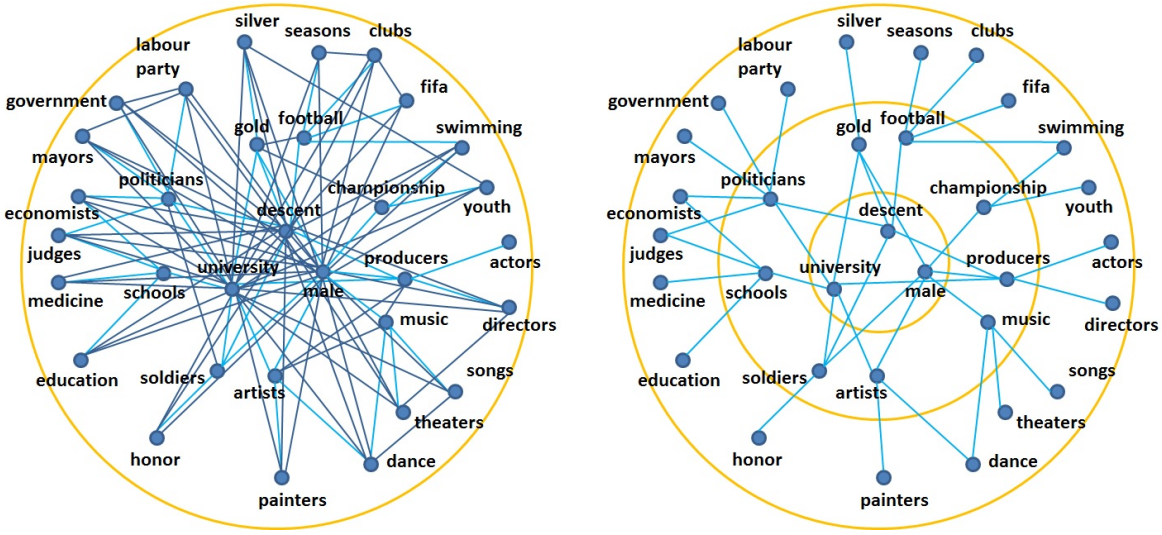
ACKNOWLEDGMENTS

We thank Dr. Roi Krakovski and Prof. Asher Koriat for fruitful discussions.

[3] http://the-wikipedia-viz-project.s3-website-eu-west-1.amazonaws.com/

Figure 1: A subgraph of the top-level Wikipedia category graph before (left) and after (right) applying the triangle elimination algorithm. Circles represent node layers.

-1in0in



REFERENCES

[1] A Asratian and N Oksimets. 1998. Graphs with Hamiltonian balls. *Australasian Journal of Combinatorics* 17 (1998), 185–198.

[2] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: an open source software for exploring and manipulating networks. In *Proceedings of the International Conference on Web and Social Media*. 361–362.

[3] Ulrik Brandes, Patrick Kenis, Jürgen Lerner, and Denise van Raaij. 2009. Network analysis of collaboration structure in Wikipedia. In *Proceedings of the 18th international conference on World Wide Web*. 731–740.

[4] Weiwei Cui, Hong Zhou, Huamin Qu, Pak Chung Wong, and Xiaoming Li. 2008. Geometry-based edge clustering for graph visualization. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1277–1284.

[5] Brian D Davison. 2000. Topical locality in the Web. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and Development in Information Retrieval*. 272–279.

[6] Tim Dwyer, Nathalie Henry Riche, Kim Marriott, and Christopher Mears. 2013. Edge compression techniques for visualization of dense directed graphs. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2596–2605.

[7] Wilbert O Galitz. 2007. *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley & Sons.

[8] Joy Paul Guilford. 1956. The structure of intellect. *Psychological bulletin* 53, 4 (1956), 267.

[9] Alexander Halavais and Derek Lackaff. 2008. An analysis of topical coverage of Wikipedia. *Journal of Computer-Mediated Communication* 13, 2 (2008), 429–440.

[10] Maurice Halbwachs. 1992. *On collective memory, edited and translated by Lewis Coser*. University of Chicago Press.

[11] Todd Holloway, Miran Bozicevic, and Katy Börner. 2007. Analyzing and visualizing the semantic coverage of Wikipedia and its authors. *Complexity* 12, 3 (2007), 30–40.

[12] Danny Holten and Jarke J Van Wijk. 2009. Force-Directed Edge Bundling for Graph Visualization. In *Computer graphics forum*, Vol. 28. 983–990.

[13] Yuntao Jia, Jared Hoberock, Michael Garland, and John Hart. 2008. On the visualization of social and other scale-free networks. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1285–1292.

[14] Joachim Kimmerle, Johannes Moskaliuk, Andreas Harrer, and Ulrike Cress. 2010. Visualizing co-evolution of individual and collective knowledge. *Information, Communication & Society* 13, 8 (2010), 1099–1121.

[15] Aniket Kittur, Ed H Chi, and Bongwon Suh. 2009. What's in Wikipedia?: mapping topics and conflict using socially annotated category structure. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1509–1512.

[16] Patrick C Kyllonen and Raymond E Christal. 1990. Reasoning ability is (little more than) working-memory capacity?! *Intelligence* 14, 4 (1990), 389–433.

[17] David D Lewis, Robert E Schapire, James P Callan, and Ron Papka. 1996. Training algorithms for linear text classifiers. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 298–306.

[18] Sorin Adam Matei and Caius Dobrescu. 2010. Wikipedia's fiNeutral Point of Viewfi: Settling Conflict through Ambiguity. *The Information Society* 27, 1 (2010), 40–51.

[19] David Milne and Ian H Witten. 2013. An open-source toolkit for mining Wikipedia. *Artificial Intelligence* 194 (2013), 222–239.

[20] Cheong-Iao Pang and Robert P Biuk-Aghai. 2011. Wikipedia world map: method and application of map-like wiki visualization. In *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*. 124–133.

[21] Christian Pentzold. 2009. Fixing the floating gap: The online encyclopaedia Wikipedia as a global memory place. *Memory Studies* 2, 2 (2009), 255–272.

[22] George G Robertson, Jock D Mackinlay, and Stuart K Card. 1991. Cone trees: animated 3D visualizations of hierarchical information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 189–194.

[23] Cindy Royal and Deepina Kapila. 2009. What's on Wikipedia, and what's not...? Assessing completeness of information. *Social Science Computer Review* 27, 1 (2009), 138–148.

[24] Filipi Nascimento Silva, Matheus Palhares Viana, Bruno Augusto Nassif Travençolo, and L da F Costa. 2011. Investigating relationships within and between category networks in Wikipedia. *Journal of Informetrics* 5, 3 (2011), 431–438.

[25] Krzysztof Suchecki, Alkim Almila Akdag Salah, Cheng Gao, and Andrea Scharnhorst. Evolution of wikipedia's category structure. *Advances in Complex Systems* 15 (????).

[26] Tatiana Von Landesberger, Arjan Kuijper, Tobias Schreck, Jörn Kohlhammer, Jarke J van Wijk, J-D Fekete, and Dieter W Fellner. 2011. Visual analysis of large graphs: state-of-the-art and future research challenges. In *Computer graphics forum*, Vol. 30. 1719–1749.

[27] Jakob Voß. 2005. Measuring wikipedia. In *Proceedings of 10th International Conference of the International Society for Scientometrics and Informetrics*.

[28] Vinko Zlatić, Miran Božičević, Hrvoje Štefančić, and Mladen Domazet. 2006. Wikipedias: Collaborative web-based encyclopedias as complex networks. *Physical Review E* 74, 1 (2006).

Figure 2: Top-level Wikipedia category graph.

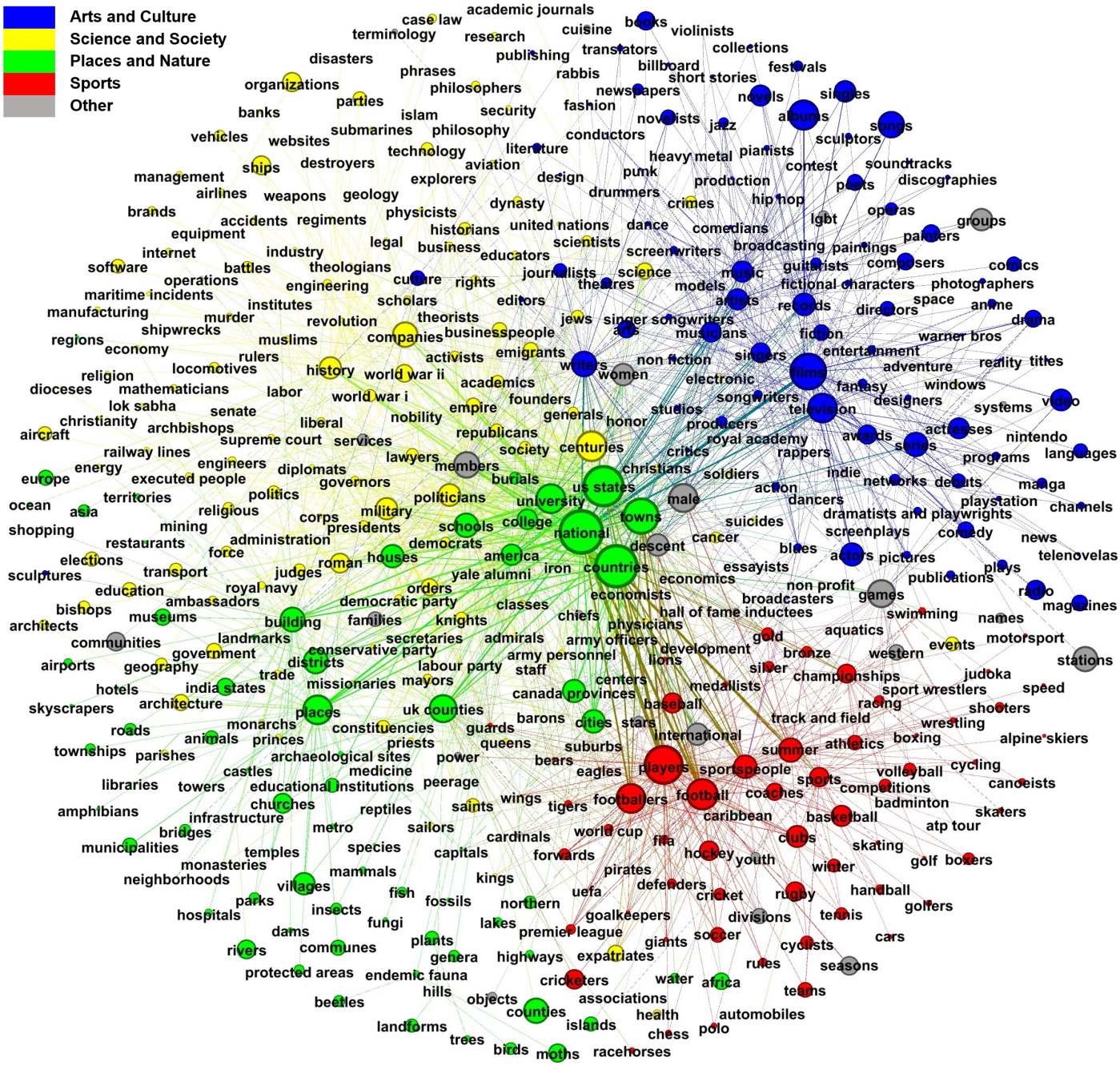
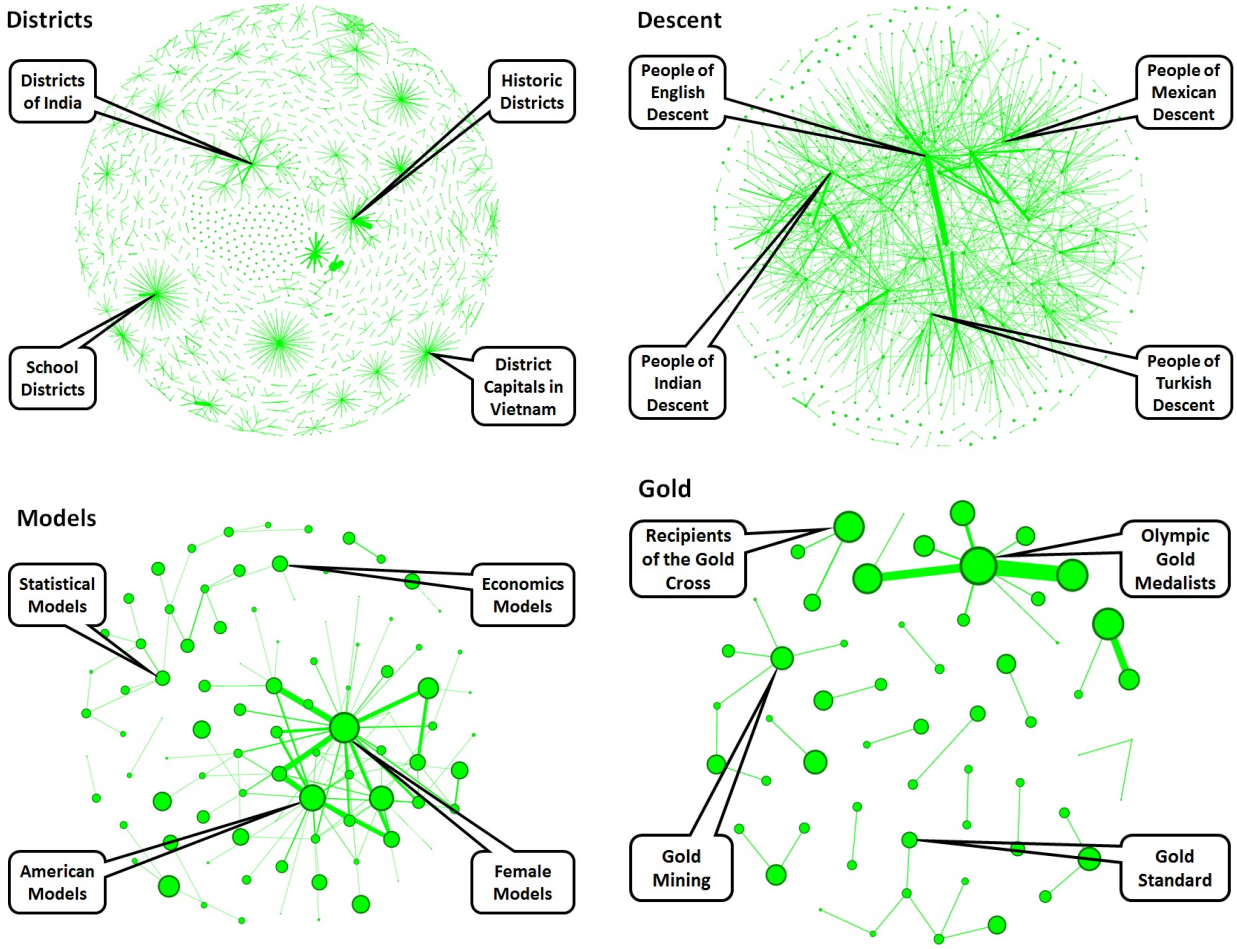


Figure 3: Examples of second-level visualization (of top-level categories *"Districts"*, *"Descent"*, *"Models"*, and *"Gold"*).



Exploring Dimensionality Reductions with Forward and Backward Projections

Marco Cavallo
IBM Research
mcavall@us.ibm.com

Çağatay Demiralp
IBM Research
cagatay.demiralp@us.ibm.com

ABSTRACT

Dimensionality reduction is a common method for analyzing and visualizing high-dimensional data across domains. Dimensionality-reduction algorithms involve complex optimizations and reduced dimensions but generally lack clear relation to the initial data dimensions, so that interpreting and reasoning about dimensionality reductions can be difficult. In this work, we introduce two interaction techniques, *forward projection* and *backward projection*, for reasoning dynamically about scatter plots of dimensionally reduced data. We also contribute two related visualization techniques, *prolines* and *feasibility map*, to facilitate and enrich the effective use of the proposed interactions, which we integrate in a new tool called *Praxis*. To evaluate our techniques, we first analyze their time and accuracy performance across varying sample and dimension sizes. We then conduct a user study in which twelve data scientists use *Praxis* so as to assess the usefulness of the techniques in performing exploratory data analysis tasks. Results suggest that our visual interactions are intuitive and effective for exploring dimensionality reductions and generating hypotheses about the underlying data.

KEYWORDS

Dimensionality reduction, interaction, bidirectional binding, visual embedding, forward projection, backward projection, PCA, autoencoder, prolines, feasibility map, what-if analysis, Praxis

This paper is published as an arXiv preprint. It can be accessed at: https://arxiv.org/pdf/1707.04281.pdf.

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), August 14th, 2017, Halifax, Nova Scotia, Canada
2017. ACM ISBN .
https://doi.org/

DycomDetector: Discover topics using automatic community detections in dynamic networks

Tommy Dang*
Texas Tech University
P.O. Box 43104
Lubbock, Texas 79409-3104
tommy.dang@ttu.edu

Vinh Nguyen†
Texas Tech University
P.O. Box 43104
Lubbock, Texas 79409-3104
vinh.nguyen@ttu.edu

Md. Yasin Kabir‡
Texas Tech University
P.O. Box 43104
Lubbock, Texas 79409-3104
yasin.kabir@ttu.edu

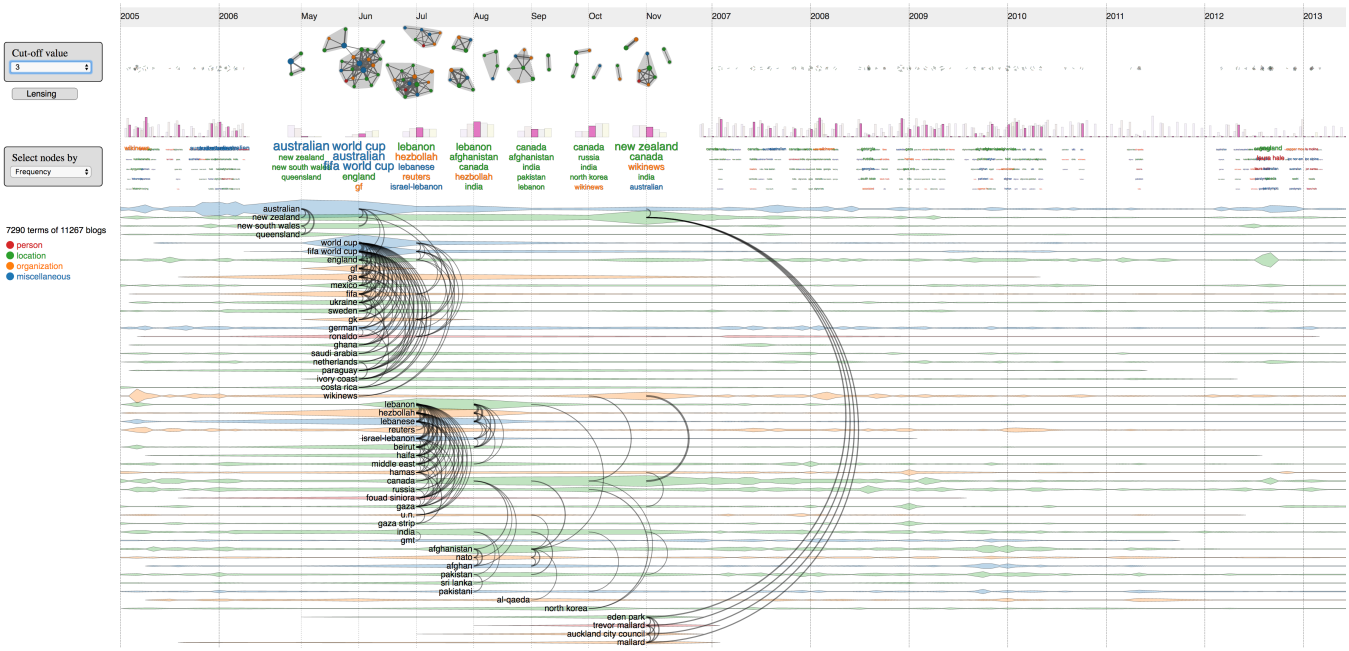


Figure 1: *DycomDetector* visualization: nodes are popular terms from Wikinews which are color-coded by categories, and links indicate the co-allocation of terms in news. The term networks for each month are displayed on the top with their features and details at the bottom.

ABSTRACT

Due to the rapid expansion and heterogeneity of the data, it is a challenging task to discover the trends/patterns and relationships in the data, especially from a corpus of texts from published documents, news, and social media. In this paper, we introduce *DycomDetector*, a novel approach for topic modeling using community detections in dynamic networks. Our algorithm extracts the important terms/phrases, formulates a network of collocated terms, and then automatically refines the network on various features (such as *term/relationship frequency*, *sudden changes* in their time series, or vertex *betweenness centralities*) to reveal the structure/communities in the given network. These communities are corresponded to different hidden topics in the input texts. *DycomDetector* provides an intuitive interface and supports a range of interactive features, such as lensing or filtering, allowing users to quickly narrow down events of interest. We also demonstrate the applications of *DycomDetector* on several real world datasets to evaluate its capabilities.

Paper type: Novel research paper

KEYWORDS

Topic Modeling, Latent Dirichlet Allocation, community detections, dynamic networks, Storyline Visualization

ACM Reference format:
Tommy Dang, Vinh Nguyen, and Md. Yasin Kabir. 2017. *DycomDetector*: Discover topics using automatic community detections in dynamic networks. In *Proceedings of , Halifax, Nova Scotia, Canada, August 14th, 2017 (KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17))*, 6 pages.

*Dr. Tommy Dang is with the Department of Computer Science at Texas Tech University.
†Vinh Nguyen is with the Department of Computer Science at Texas Tech University.
‡Md. Yasin Kabir is with the Department of Computer Science at Texas Tech University.

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada
2017. .
DOI:

DOI:

1 INTRODUCTION

We are living in the age of big data in which a vast amount of digitalized information being collected grows exponentially. The expansion of IT infrastructure along with broadband uses helps us access information instantly. Although more data becomes available, accessing to what information we are looking for is still a challenging task due to the level of the details we are trying to achieve.For example in 2016, Twitter had approximately 500 million tweets per day, while Facebook had 216 million posts within the same time scale [21]. One might be interested in various specific questions such as: what were the hottest topics on Twitter last month? Are there any commonalities among different demographics of people? Or how did interests about a particular topic change over time? These intriguing questions have been motivating researchers to look for answers over the years. Thus the need to have automated tools and techniques to filter, organize, and explore vast quantities of information is highly desirable.

Our contributions in this paper thus are:

- We present a new approach for discovering topics based on graph theory, specifically community detection in networks.
- We provide an interactive data analytics prototype, called *DycomDetector*, for visualizing and analyzing topic abstractions and how they change overtime.
- We provide use cases on other application domains and make comparisons to a classic topic modeling algorithm.

The rest of this paper is organized as follows. Section 2 reviews related work and existing methodologies and Section 2.3 discusses our approach for community detection in networks. Section 3 introduces the *DycomDetector* prototype and illustrates it on real datasets. In Section 4, we present our experiments on real-world topic discoveries in dynamic networks. Finally, we conclude our paper with future work.

2 RELATED WORK

2.1 Topic Modeling

Latent Dirichlet Allocation (LDA) [3] is a flexible generative probabilistic model for text mining. LDA aims to find potential topics in the text corpora. In this mining approach, documents are treated as random mixtures of certain number of topics, and LDA categorizes the topics based on the distribution over the words through a three-level hierarchical Bayesian model. Doyle et al. [13] implement basic LDA for financial topics modeling for stock market to detect the companies which tend to move together. Wang et al. [25] extend LDA into *Spatial Latent Dirichlet Allocation* (SLDA), which encodes spatial structures among visual words into the same topic. *LDAAnalyzer* [30], a tool designed for software engineering researchers, used for source code modeling and numerical data visualization was developed based on LDA.

Topic modeling enables us to organize, re-order, and summarize large text corpora in an effective way. Hence, it is used highly for the visualization of large data in a time efficient manner. Many good visualization tools and techniques have been developed for the visualization of topics. Wei et al. [27] present TIARA, which determines time-sensitive keywords to portray the content evolution of each topic over time using stack graph metaphor. ParallelTopics [12] was also developed to represent the temporal changes of topics using Parallel Coordinates.

2.2 Dynamic Network Visualization

Improved technologies and devices enable us to gather data which are changing in every moment. The large temporal data from various fields motivate the creation of novel visualization techniques. Beck et al. [2] present state of the art in visualizing dynamic data which provide an overview of the novel techniques for representing relational data. This survey provides a hierarchical taxonomy of dynamic graph visualization and classifies the existing techniques into the taxonomy based on a systematic literature review. The survey shows that time-line based techniques (time-to-space mapping) are becoming more popular in dynamic visualization.

For visualizing the temporal changes in dynamic networks [6, 23, 29] matrices are very useful. When visualizing dense graphs [15] adjacency matrices are particularly effective as they avoid edge-crossing problem in node-link diagrams [10, 14, 18]. *TimeMatrix* [29] displays a modest temporal bar chart inside each cell of the matrix which allow comparing the changes of edge weights for the two corresponding vertices. Alternatively, *gestaltmatrix* [7] uses *gestaltlines*, intra-cell lines that encode various metrics utilizing the angle and length. *Matrix Cubes* [1] stacks adjacency matrices in chronological order to represent the dynamic networks based on the space-time cube metaphor.

2.3 Community detection in networks

One of the most widely used algorithms for community detection in practice is the Louvain method [4] because of its speed and desired modularity value. This is a greedy optimization method, that is, it first looks for "small" communities by optimizing modularity locally, then each small community is grouped into one node and the first step is repeated iteratively until a maximum of modularity is attained and a hierarchy of communities is produced.

3 *DYCOMDETECTOR* VISUALIZATION

To enable users to explore the vast temporal text corpora in an interactive and efficient way, *DycomDetector* visualization introduces several components and following steps:

- **Compute and extract the terms**: Our method extracts terms from text data based on frequency and standardized net frequency of the entities at each time point. We then rank them and filter only the top 200 terms. We describe this step in Section 3.1.
- **Construct relationships**: This step constructs the relationships between terms/phrases based on the contexts that they are situated together in Section 3.2.
- **Redefine and reorder the vertices**: This process allows reordering the vertices based on a user selected parameter in Section 3.3.
- **Generate visualizations**: The visualization is generated in this step. Due to the limited screen display, each network (at each time point) is represented as a thumbnail which summarizes the structure of the network in Section 3.4.

- **Interations**: Users can explore popular terms and dynamic relationships between them via various interactions and selections supported within *DycomDetector*. (Section 3.5).

The *DycomDetector* implements four low-level analysis tasks:

- **T1**: Provide a summary view of text corpus over time [17]. *DycomDetector* provides a quick overview of important topics using the network thumbnails. Moreover, we also display a summary histogram of network modularity on different settings as well as the top 5 popular terms in these communities (see Section 3.2).
- **T2**: Mouse over timeline to expand several consecutive snapshots of the network and the relationships of collocated terms (see Section 3.3).
- **T3**: Filter terms/ topics on user request (see Section 3.3). For example, users may want to see political events at a specific geographic area (see Section 3.5).
- **T4**: Sort terms based on a selected measure: *term frequency*, *sudden increase in frequency*, *vertex degree*, or *betweenness centrality* (see Section 3.4).

3.1 Extract terms

The input text documents are preprocessed into entities and further classified into different categories: people, locations, organizations, and others. We use colors to encode these categories: red for person, green for location, blue for organization, and orange for miscellaneous. This color-encoding is used consistently for all figures in the paper.

3.2 Construct networks for each time stamp

There are several methods to generate relationships among two given text element entities. For instance, a link can be generated by the similarity between two documents, or the frequency with which two entities are mentioned together [16], or defined logical forms[19]. The weight for each link is calculated accordingly. Since *DycomDetector* works directly on individual terms to obtain community-based coherence, the relationship is determined based on the collocation of the terms in the same articles/blogs. A link with high weight indicates that two terms are frequently situated together in a given period of time, such as within one day or within one month.

3.3 Redefine the vertices

The *DycomDetector* allows users to narrow down the text network using different parameters (visualization task **T3**). In particular, users can redefine vertices in the relationship network using several properties which are divided into non-network properties (including *term frequency* and *sudden increase in frequency*) and network-related properties (including *vertex degree* and *betweenness centrality*). Depending on the user selection, *DycomDetector* recalculates the networks and their communities' formulations accordingly.

3.4 Generate visualizations

In *DycomDetector* visualization, we align network snapshots horizontally from left to right. This design is widely adopted in many time series visualizations [8, 9, 26]. A histogram below each network displays modularity Q on different filtering settings (called *relationship cut*). *Relationship cut* can extend from 1 to the highest weight of vetices in all terms networks.

The term timelines are shown at the bottom. In particular, *CloudLines* style visualization [20, 22] can be overlaid to highlight the evolution of terms over time. Arcs are used to connect collocated terms which are ordered by month. Terms in the same month are ordered by communities to reduce edge-crossings (visualization task **T4**) since the community detection algorithm groups highly connected nodes (frequently collocated terms) and loosely connected nodes in other clusters. The quality of produced cluster formation is reflected in modularity Q presented in histograms above.

3.5 Interactions

DycomDetector supports a range of interactions, such as lensing or filtering (based on the four features above), allowing users to quickly drill down events of interest. Moreover, users can input into a search box to provide a topic of interest (visualization task **T3**). Figure 2 shows an example of topics related to the inputted geographic location "Tucson". Data is political blogs from the *Huffington Post* which contain 75,293 blogs, and we extracted 33,528 terms in total. Notice that the networks and their cluster formations (number of clusters as well as members in each cluster) are very different for different months. Consequently, their modularity histograms vary significantly. In the example, only the second bar in each histogram is highlighted since users have set the *relationship cut* to 2.

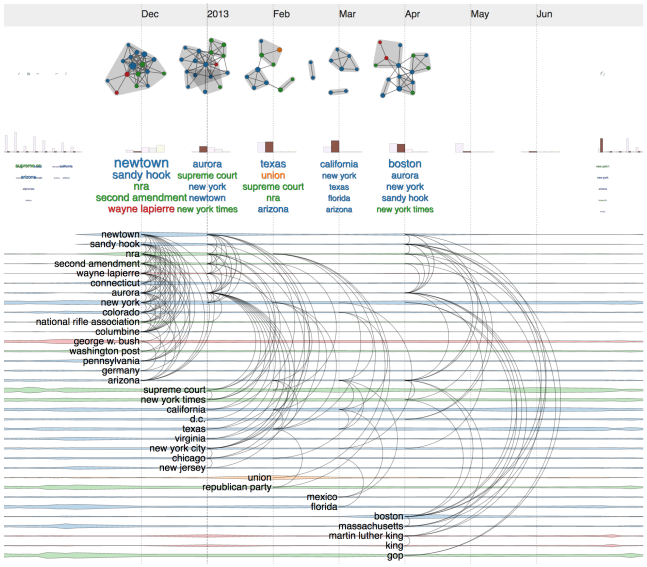


Figure 2: Related topics in the *Huffington Post* for an input term "Tucson". Terms in the list are ordered by *betweenness centrality*. For example, *Newtown* and *Sandy Hook* are the two central terms appearing on many political blogs due to the shooting event at the Sandy Hook Elementary School shooting which occurred on December 14, 2012, in Newtown, Connecticut [28].

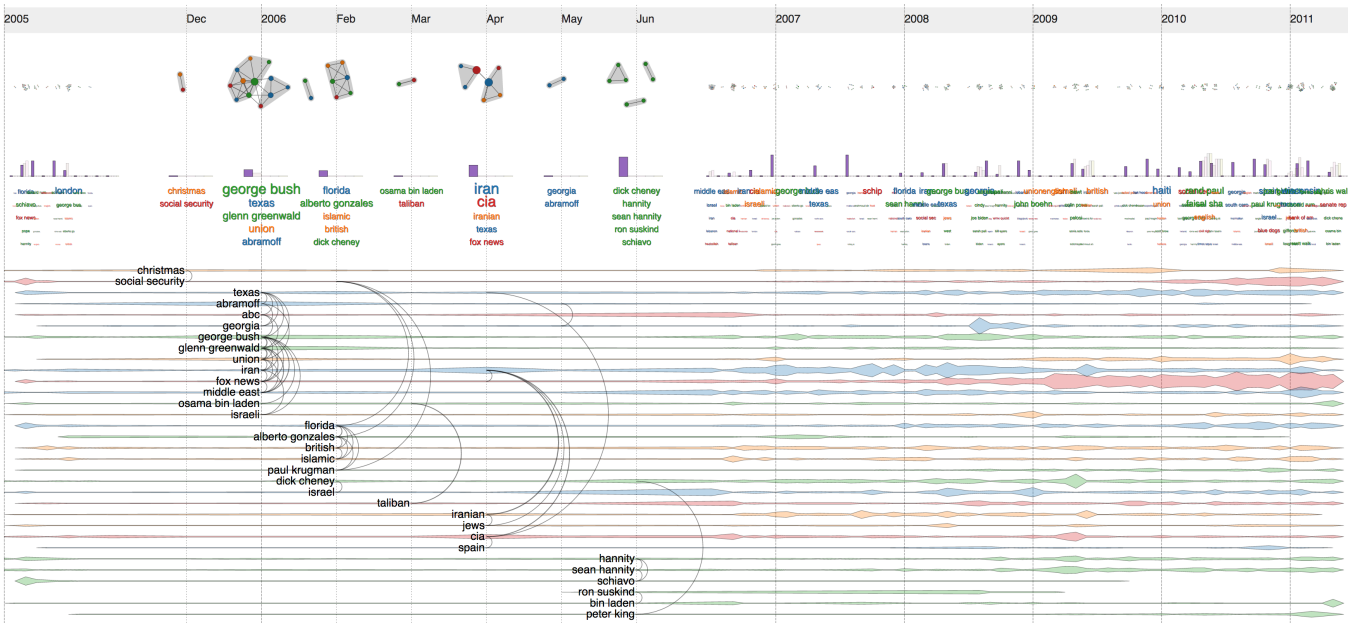


Figure 3: *DycomDetector* on Crooks And Liars blogs with lensing into 2006. The top 5 terms under modularity histograms are ordered by *betweenness centrality*. For example, *iran* and *cia* are the hottest terms in April 2006.

Since it is difficult to explain the interactions with *DycomDetector* from static images, we advise the viewers to conduct the demo video on our *DycomDetector* project page.

4 EXPERIMENTS

4.1 Datasets and use cases

We will illustrate the features of *DycomDetector* mainly through examples. We use datasets retrieved from different political blogs, such as Americablog, Crooks and Liars, the Huffington Post, and other sources to demonstrate the performance of *DycomDetector*.

Figure 3 shows an example of *DycomDetector* on Crooks And Liars blogs (which contains 9,663 blogs and 4,935 terms) while Figure 4 uses the Ensquire data (which contains 2,208 blogs and 2,117 terms). The figures depict lensing effects at different time stamps but with the same settings: terms are selected and ranked based on *net frequency* while connections are filtered by *relationship cut* to maximize the modularity scores on each network snapshot. In Figure 4, notice that the nodes have different sizes depending on their *betweenness centrality*. Moreover, the best *relationship cut* for revealing network structure is different (highlighted in different bar colors underneath each network).

More examples and other use cases are provided on our project, available at https://github.com/iDVLTTU/DycomDetector.

4.2 Comparisons with LDA

In order to validate the performance of our proposed prototype, we make a comparison between *DycomDetector* and a well-known LDA model [3]. The result in Figure 5 shows that our proposed *DycomDetector* model generates some overlapping terms with the LDA model. For example, in July 2014 some overlapping terms are *hamas*, *gaza*, and *strip*. As depicted, it is easier to keep track of the evolution of terms over time in *DycomDetector*. Notice that the term *russia* mentioned most in March 2014 (highest frequency value) does not show up in the top ranking words in the next two months then is mentioned again in June and becomes a hot topic yet again in September.

Figure 5(c) allows users to look at the filtered topics at a different angle. The network is constructed by *sudden change* in terms frequency. *Betweenness centrality* is applied to highlight important terms (terms serve as bridges in these networks). Notice that node sizes are computed to reflect their connecting roles (*betweenness centrality*).

4.3 Implementation

DycomDetector is implemented in D3.js [5]. The application, source code, sample data, and demo video are provided via our GitHub project repository, located at https://github.com/iDataVisualizationLab/DycomDetector.

5 CONCLUSION

In this paper, we present a novel approach that incorporates a community detection algorithm to find topics and reveal network structure in temporal data automatically. We also introduce an interactive data analytics prototype which helps users to visualize and analyze topic abstractions and how they change over time. Our experiments on various datasets show that *DycomDetector* provides a better lens into large corpus of texts obtained from news/blogs. Furthermore, our study demonstrates the usefulness of each component of our *DycomDetector* model. Our model also supports a wide range of capabilities such as dynamic clustering

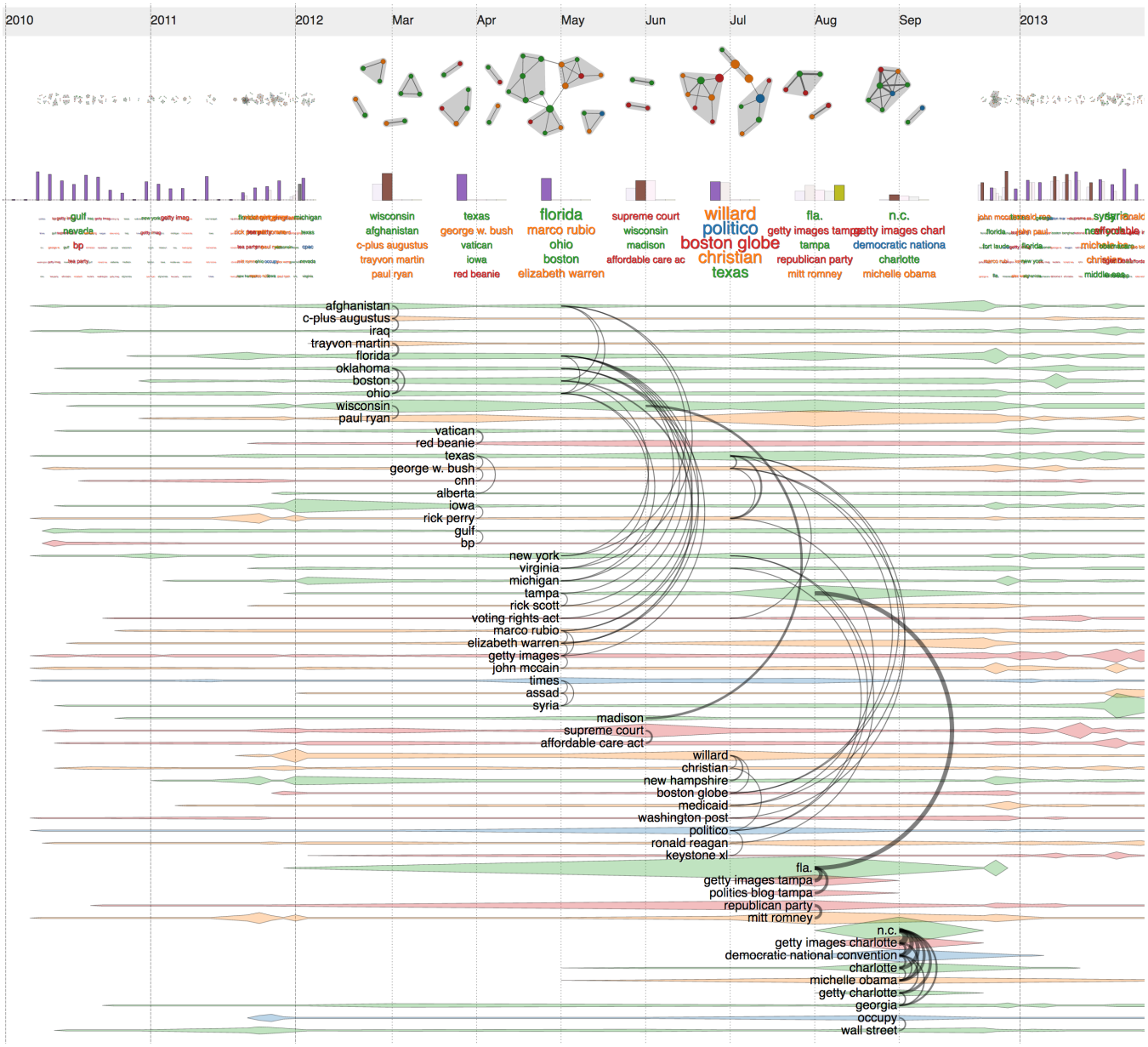


Figure 4: *DycomDetector* on Ensquire blogs with lensing into 2012. The top 5 terms under modularity histograms are ordered by *betweenness centrality*. For example, *willard*, *politico* and *boston globe* are the hottest terms in July 2012.

community over time and providing additional informative filtering selection.

In future work, we are planning to introduce topic recommendations into our prototype (the complete inferred sentence extracted from clustered topics). This new direction is very promising since it helps users gain a better understanding of given topics from an extensive collection of text documents. Moreover, applying community detection into dynamic network to automate the process of revealing network structures has many applications in other domains where tracking the structural changes is a vital part [11, 24].

REFERENCES

[1] Benjamin Bach, Emmanuel Pietriga, and Jean-Daniel Fekete. 2014. Visualizing Dynamic Networks with Matrix Cubes. In *Proc. ACM Conf. on Human Factors in Computing Systems*. 877–886.

[2] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. 2016. A taxonomy and survey of dynamic graph visualization. In *Computer Graphics Forum*. Wiley Online Library.

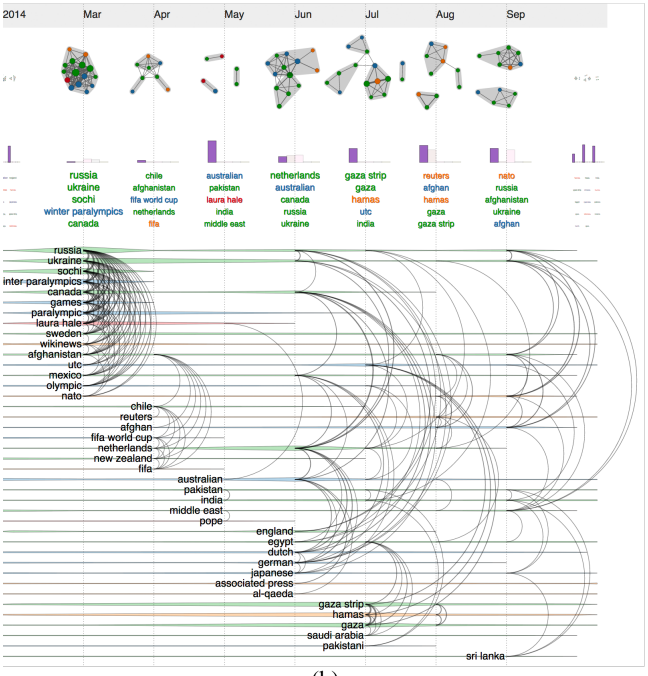
[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

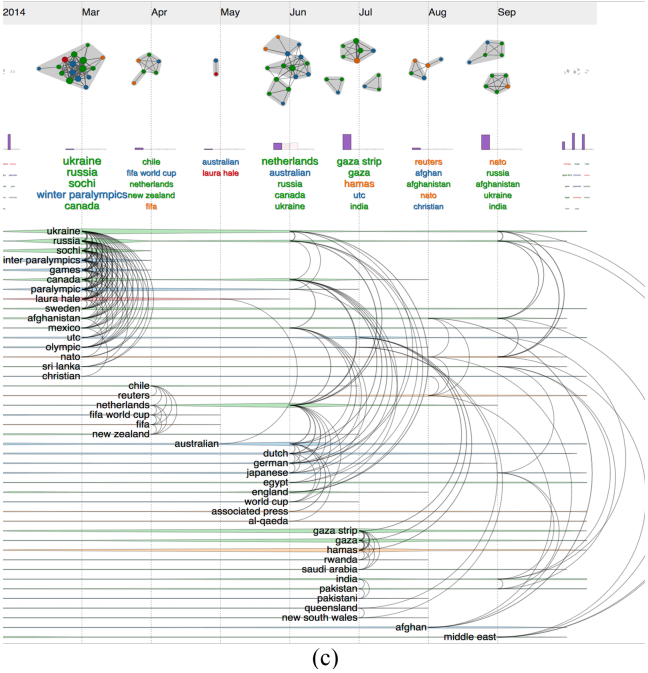
[5] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D3 Data-Driven Documents. *IEEE Trans. Vis. Comput. Graph.* 17, 12 (2011), 2301–2309.

Mar	Apr		May	Jun			Jul			Aug		Sep	
Topic Term/Freq	Topic Term/Freq		Topic Term/Freq	Topic Term/Freq			Topic Term/Freq			Topic Term/Freq		Topic Term/Freq	
1	1	2	1	1	2	3	1	2	3	1	2	1	2
russia	police	species	gillett	governme nt	canada	team	israeli	coaching	university	center	forces	students	university
united	team	wikipedia	patrick	gold	united	ndash	gaza	basketball	stage	mcneil	reported	police	nato
hale	black	race	university	people	game	world	hamas	kyle	try	brian	military	people	state
team	macdo nald	genus	brian	speech	australia	netherland s	border	started	ozone	thumb	asteroid	noosa	final
mdash	united	stumira	whale	record	party	points	military	tom	minutes	street	security	paisley	toads
senator	univers ity	committee	mcneil	men's	australian	play	israel	coach	assistant	university	state	minister	afghanistan
laura	people	school	protesters	col	national	germany	strip	i'd	noosa	festival	passport	adams	president

(a)



(b)



(c)

Figure 5: A comparison between *DycomDetector* and LDA model based on frequency with *relationship cut* = 1 and data from March to September in 2014: a) topics obtained by LDA model b) topics generated by *DycomDetector* model based on *sudden changes* in term frequency c) topics generated by *DycomDetector* model based on *betweenness centrality*.

[6] U. Brandes and S. R. Corman. 2002. Visual unrolling of network evolution and the analysis of dynamic discourse. In *IEEE Symp. on Information Visualization*. 145–151.

[7] Ulrik Brandes and Bobo Nick. 2011. Asymmetric Relations in Longitudinal Social Networks. *IEEE Trans. Vis. Comput. Graph.* 17, 12 (2011), 2283–2290. https://doi.org/10.1109/TVCG.2011.169

[8] Lee Byron and Martin Wattenberg. 2008. Stacked Graphs – Geometry & Aesthetics. *IEEE Trans. Vis. Comput. Graph.* 14, 6 (2008), 1245–1252. https://doi.org/10.1109/TVCG.2008.166

[9] Tuan Nhon Dang, A. Anand, and L. Wilkinson. 2013. TimeSeer: Scagnostics for High-Dimensional Time Series. *IEEE Trans. Vis. Comput. Graph.* 19, 3 (March 2013), 470–483. https://doi.org/10.1109/TVCG.2012.128

[10] Tuan Nhon Dang, Paul Murray, and Angus G. Forbes. 2015. PathwayMatrix: Visualizing Binary Relationships between Proteins in Biological Pathways. *BMC Proceedings* 9, 6 (2015), S3.

[11] Tuan Nhon Dang, Nick Pendar, and Angus G. Forbes. 2016. TimeArcs: Visualizing Fluctuations in Dynamic Networks. *Computer Graphics Forum* (2016). https://doi.org/10.1111/cgf.12882

[12] Wenwen Dou, Xiaoyu Wang, Remco Chang, and William Ribarsky. 2011. Paralleltopics: A probabilistic approach to exploring document collections. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*. IEEE, 231–240.

[13] Gabriel Doyle and Charles Elkan. 2009. Financial topic models. In *Working Notes of the NIPS-2009 Workshop on Applications for Topic Models: Text and Beyond Workshop*.

[14] Mohammad Ghoniem, Jean-Daniel Fekete, and Philippe Castagliola. 2005. On the Readability of Graphs Using Node-link and Matrix-based Representations: A Controlled Experiment and Statistical Analysis. *Inf. Vis.* 4, 2 (2005), 114–135. https://doi.org/10.1057/palgrave.ivs.9500092

[15] N. Henry and J. d. Fekete. 2006. MatrixExplorer: A Dual-Representation System to Explore Social Networks. *IEEE Trans. Vis. Comput. Graph.* 12, 5 (2006), 677–684. https://doi.org/10.1109/TVCG.2006.160

[16] Fang Jin, Rupinder Paul Khandpur, Nathan Self, Edward Dougherty, Sheng Guo, Feng Chen, B Aditya Prakash, and Naren Ramakrishnan. 2014. Modeling mass protest adoption in social network communities using geometric brownian motion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1660–1669.

[17] Daniel A. Keim, Christian Panse, and Mike Sips. 2004. Information Visualization : Scope, Techniques and Opportunities for Geovisualization. In *Exploring Geovisualization*, J. Dykes (Ed.). Elsevier, Oxford, 1–17.

[18] René Keller, Claudia M. Eckert, and P. John Clarkson. 2006. Matrices or Node-link Diagrams: Which Visual Representation is Better for Visualising Connectivity Models? *Inf. Vis.* 5, 1 (2006), 62–76. https://doi.org/10.1057/palgrave.ivs.9500116

[19] Stanley Kok and Pedro Domingos. 2008. Extracting semantic networks from text via relational clustering. *Machine Learning and Knowledge Discovery in Databases* (2008), 624–639.

[20] M. Krstajic, E. Bertini, and D.A. Keim. 2011. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. *IEEE Trans. Vis. Comput. Graph.* 17, 12 (2011), 2432–2439.

[21] Lisa Lowe. 2016. Socialpilot. https://socialpilot.co/blog/125-amazing-social-media-statistics-know-2016/. (March 2016).

[22] Dongning Luo, Jing Yang, Milos Krstajic, William Ribarsky, and Daniel Keim. 2012. Eventriver: Visually exploring text collections with temporal references. *IEEE Trans. Vis. Comput. Graph.* 18, 1 (2012), 93–105.

[23] Chihua Ma, Robert V. Kenyon, Angus G. Forbes, Tanya Berger-Wolf, Bernard J. Slater, and Daniel A. Llano. 2015. Visualizing Dynamic Brain Networks Using an Animated Dual-Representation. In *Proc. Eurographics Conf. on Visualization*. 73–77.

[24] Chayant Tantipathananandh and Tanya Y Berger-Wolf. 2011. Finding communities in dynamic social networks. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 1236–1241.

[25] Xiaogang Wang and Eric Grimson. 2008. Spatial latent dirichlet allocation. In *Advances in neural information processing systems*. 1577–1584.

[26] Martin Wattenberg. 2005. Baby Names, Visualization, and Social Data Analysis. In *Proc. IEEE Symp. on Information Visualization*. 1–7.

[27] Furu Wei, Shixia Liu, Yangqiu Song, Shimei Pan, Michelle X Zhou, Weihong Qian, Lei Shi, Li Tan, and Qiang Zhang. 2010. Tiara: a visual exploratory text analytic system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 153–162.

[28] wikipedia. 2017. Sandy Hook Elementary School shooting. (2017). https://en.wikipedia.org/wiki/Sandy_Hook_Elementary_School_shooting

[29] Ji Soo Yi, Niklas Elmqvist, and Lee Seungyoon. 2010. TimeMatrix: Analyzing Temporal Social Networks Using Interactive Matrix-Based Visualizations. *Int. J. Hum. Comput. Int.* 26, 11-12 (2010), 1031–1051.

[30] Chunyao Zou and Daqing Hou. 2014. LDA analyzer: A tool for exploring topic models. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*. IEEE, 593–596.

Incorporating Feedback into Tree-based Anomaly Detection

Shubhomoy Das
Oregon State University
Corvallis, Oregon 97330
dassh@oregonstate.edu

Weng-Keen Wong
Oregon State University
Corvallis, Oregon 97330
wongwe@oregonstate.edu

Alan Fern
Oregon State University
Corvallis, Oregon 97330
Alan.Fern@oregonstate.edu

Thomas G. Dietterich
Oregon State University
Corvallis, Oregon 97330
tgd@oregonstate.edu

Md Amran Siddiqui
Oregon State University
Corvallis, Oregon 97330
siddiqmd@oregonstate.edu

ABSTRACT

Anomaly detectors are often used to produce a ranked list of statistical anomalies, which are examined by human analysts in order to extract the actual anomalies of interest. Unfortunately, in real-world applications, this process can be exceedingly difficult for the analyst since a large fraction of high-ranking anomalies are false positives and not interesting from the application perspective. In this paper, we aim to make the analyst's job easier by allowing for analyst feedback during the investigation process. Ideally, the feedback influences the ranking of the anomaly detector in a way that reduces the number of false positives that must be examined before discovering the anomalies of interest. In particular, we introduce a novel technique for incorporating simple binary feedback into tree-based anomaly detectors. We focus on the Isolation Forest algorithm as a representative tree-based anomaly detector, and show that we can significantly improve its performance by incorporating feedback, when compared with the baseline algorithm that does not incorporate feedback. Our technique is simple and scales well as the size of the data increases, which makes it suitable for interactive discovery of anomalies in large datasets.

CCS CONCEPTS

• **Computing methodologies** → **Active learning settings**; **Semi-supervised learning settings**;

KEYWORDS

Anomaly Detection, Active Learning, User Feedback, Semi-supervised Learning, Optimization

ACM Reference format:
Shubhomoy Das, Weng-Keen Wong, Alan Fern, Thomas G. Dietterich, and Md Amran Siddiqui. 2017. Incorporating Feedback into Tree-based Anomaly Detection. In *Proceedings of , Halifax, Nova Scotia, Canada, August 14th, 2017 (KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17))*, 9 pages.
https://doi.org/

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), August 14th, 2017, Halifax, Nova Scotia, Canada

ACM ISBN .
https://doi.org/

1 INTRODUCTION

We define an *anomaly* as a data instance generated by a different process than the process generating the nominal data. On the other hand, we define an *outlier* as a data instance that has low likelihood according to a model. Anomaly detectors are in general very good at detecting outliers. However, not all outliers are anomalies. Some outliers are statistical noise, while others might not interest the end-users. A class of the state-of-the-art anomaly detectors dependent on unsupervised tree-based methods [2, 5, 8, 11] are not naturally immune to this problem. These detectors usually partition the feature space into multiple (sometimes overlapping and hierarchical) regions and assign scores to each region individually. When the scores computed for some of the regions do not reflect their true relevance to the user's notion of an anomaly, it creates a semantic mismatch between what the user considers an anomaly and what the algorithm considers an outlier. In order to avoid this mismatch, we need expert-feedback to make outliers more in line with expert's idea of an anomaly.

Active Anomaly Discovery (AAD) [3] is one of the most recent methods for incorporating analyst-feedback into an ensemble of anomaly detectors. In this paper, we show that tree-based anomaly detectors can also be treated as *ensembles* such that we can incorporate feedback into them by employing AAD. We present an implementation of this concept in the specific context of the tree-based anomaly detector Isolation Forest [5], which is competitive with other state-of-the-art anomaly detectors [4, 5]. One advantage of the proposed approach is that it allows incorporating feedback at a finer level than simply combining the outputs of multiple detectors linearly.

In Section 2, we present our view of the general structure of tree-based anomaly detectors, and illustrate this view with Isolation Forest as an example. Section 3 presents an overview of AAD and then extends AAD to incorporate feedback into the Isolation Forest. We refer to this new algorithm as *IF-AAD*. Section 4 presents quantitative empirical results on eight benchmark datasets and provides a visualization of the feedback process in order to gain further insight into how the feedback affects which instances are queried. Finally, we summarize the contributions and results in Section 5.

2 TREE-BASED ANOMALY DETECTORS

We consider an anomaly detection setting where an anomaly detector is used to assign anomaly scores to data instances, which are assumed to be feature vectors in $\mathcal{R}^n$. The instances can then

Table 1: Node weights for tree-based algorithms.

Name	Internal node weight	Leaf node weight
Isolation Forest [5]	-1	-1
HS-Trees [8]	0	anomaly score as defined in Tan et al. [8]
RS-Forest [11]	0	anomaly score as defined in Wu et al. [11]
RPAD ('AVG' variant) [7]	normalized pattern frequency [7]	normalized pattern frequency [7]
Random Projection Forest [2]	log-probability at the node [2]	log-probability at the leaf [2]

be presented to an analyst in ranked order, starting with the most anomalous instance. Our work is motivated by the observation that a number of state-of-the-art anomaly detectors are based on decision-tree ensembles, or forests. The internal nodes of each tree correspond to threshold tests on selected features. Thus, a given instance x will follow a unique path from the root to a leaf in each tree.

Each tree node v in the tree-based anomaly detector stores a real-valued weight w_v, which is used to calculate the anomaly scores. The anomaly score of an instance x is simply equal to the average over weights of all tree nodes that the instance follows in the forest. Note that each node in the forest can be viewed as defining a distinct volume in $\mathcal{R}^n$ and thus the total score is a combination of the weights of these overlapping volumes.

Despite the simplicity of this anomaly detection structure, a number of state-of-the-art algorithms can be represented as a particular choice of weight values and methods to construct the trees (generally highly randomized trees). Table 1 illustrates the weight values that correspond to a number of algorithms. As one example and as described in detail below, the Isolation Forest [5] algorithm assigns a constant weight of -1 to all tree nodes.[1] The anomaly score then evaluates to be the average path length traversed by an instance across trees in the forest.

As another example, the HS-Trees[8] algorithm, assigns a weight of $v_r \times 2^{v_k}$ to each **leaf** node v, where v_r is the number of training instances at the node v, and v_k is the node's depth. In addition, HS-Trees assigns weight 0 to all non-leaf nodes. Thus, in this case the anomaly score of an instance x is the average of the weights at leaves it reaches.

In practice, there is no uniformly best anomaly detector (or equivalently, a fixed setting of the weights) across the possible applications. Rather, the best performing detector for a given application will depend on how well a detector's notion of "outlier" matches the analyst's notion of "interesting anomaly". This is difficult to predict for a given application. Further, it is unlikely that any of the weight settings corresponding to state-of-the-art detectors will be optimal for a given application when considering the entire range of possible weight settings.

The above motivates incorporating user feedback during use of an anomaly detector to attempt to tune the weights toward the ideal application-specific detector. In Section 4, we show that this approach often increases the number true anomalies discovered within a particular budget of instances that can be examined by an analyst. In this paper, we treat Isolation Forest as a representative tree-based anomaly detector, and explain our method for incorporating feedback, where the detector is initialized to the Isolation Forest weights. Below we describe in detail the Isolation Forest algorithm for concreteness and illustrate how it is easily captured in our tree-based anomaly detection framework.

[1] This assumes the trees are grown to a depth where instances are isolated. Otherwise the leaf nodes would have alternative weights that depend on the amount of data arriving at each leaf.

2.1 Isolation Forest (IF)

Isolation Forest (IF) [5] comprises of a set of t trees denoted by $\mathbf{T} = \{T_1, ..., T_t\}$ constructed in a randomized manner as outlined in Algorithm 1, and illustrated in Figure 1a. Each tree is constructed from the root to leaves by randomly partitioning the data at each node by selecting a feature and a threshold both at uniformly random. The trees are grown until each instance is isolated in a leaf. IF is based on the idea that anomalous instances are few, and they are well-separated from clusters of nominal instances in the feature space. Because of this, anomalous instances very quickly reach leaf nodes through random partitioning. On the other hand, nominal instances, which form dense clusters, require many more splits to finally reach leaf nodes. Therefore, the length of the path traversed by an instance from the root node to the leaf, also known as the isolation depth, is shorter (on average) for anomalous instances than it is for nominal instances. The anomaly score assigned to an instance is simply the average isolation depth across the forest.

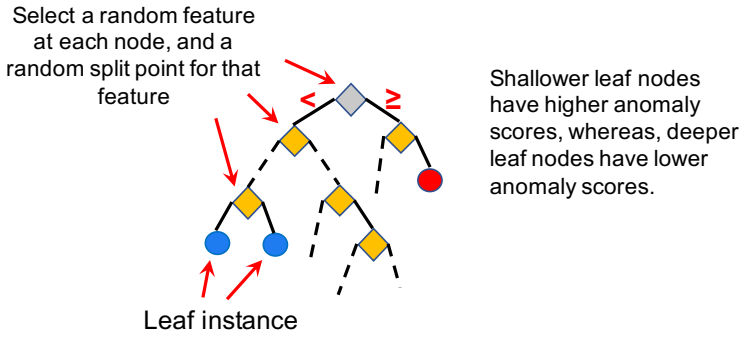
It is straightforward now to describe IF as a particular way of setting the weights of a tree-based anomaly detector. In particular, the weight of each node v is $w_v = -1$ (constant). Given an instance x, it is easy to see that the anomaly score assigned by the tree-based detector is simply negative of the average number of nodes on paths traversed by x in the forest, i.e. negative of the average isolation depth. Note that, the main purpose to make scores negative is to ensure that higher scores indicate more anomalous and lower scores indicate more nominal.

In order to describe our algorithm for feedback, it is convenient to view the score assigned by the detector as a linear score function. To do this, for each tree node v define an indicator feature $z_v \in \{0, 1\}$. The anomaly score is then simply the dot product of feature and weight vectors, that is,

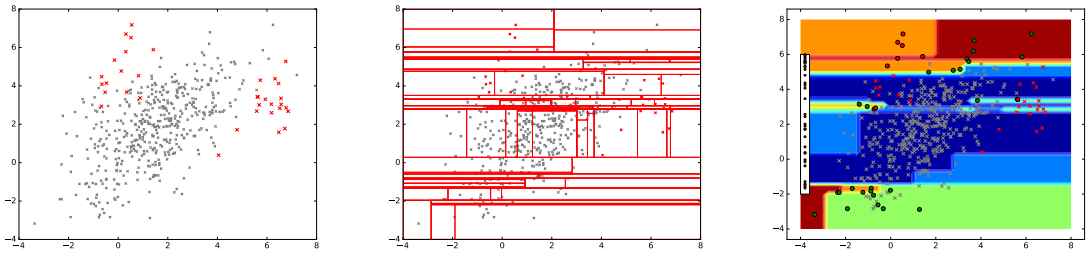
$$\text{score}(x) = z \cdot w,$$

where the dimension of each vector is the number of nodes in the forest.

Figure 1 illustrates the anomaly score contours for IF with a single tree on synthetic data. The anomaly score contours in Figure 1d show that a single isolation tree is not very informative. However, if we increase the number of trees in the ensemble, their combined scores can be fairly accurate even without feedback. This is illustrated in Figure 2a where the number of trees is 100.



(a) Isolation Tree Construction



(b) Synthetic dataset (c) 1 Tree (d) Contours with 1 Tree

Figure 1: Random trees in Isolation Forest (IF) for synthetic data. The points in red are true anomalies; points in gray are true nominals. Figure 1c shows the leaf node regions for a single tree generated by random IF splits. Figure 1d shows the contours of anomaly scores assigned to the nodes of this tree. Deeper red means more anomalous; deeper blue means more nominal. The red circles are the true anomalies among the top ranked 35 instances. The green circles are the true nominals among the top ranked 35 instances. The left sidebar in Figure 1d shows the ranking of true anomalies (red dots). Ideally, true anomalies should be near the top on this bar.

Algorithm 1 Generating randomized trees in Isolation Forest

Input: **D**, sub-sample size: N, number of trees: t
$\mathbf{T} = \emptyset$
for $i = 1 \ldots t$ **do**
  Let S_i = a sub-sample of N instances from **D**
  Build tree T_i as follows, by starting with all instances in S_i at the root node:
    Let $U \subseteq S_i$ be the set of instances at the current node
    if $|U| == 1$ **then**
      return
    else
      Let f be a feature sampled at random
      Let f_{min} = min. value of f across all instances in U
      Let f_{max} = max. value of f across all instances in U
      Let p_f = value sampled unif. random in $[f_{min}, f_{max}]$
      Partition U into two parts on the basis of p_f and recurse on both partitions
    end if
  $\mathbf{T} = \mathbf{T} \cup T_i$
end for

3 RE-WEIGHTING TREE PARTITIONS

We now describe our approach for adjusting the weights in the above score function based on feedback from the analyst.

3.1 Active Anomaly Discovery (AAD)

AAD is an algorithm (Algorithm 2) that tries to maximize the number of true anomalies presented to the analyst in an interactive feedback loop. It assigns an anomaly score to each instance such that a higher score means more anomalous. The instances are internally ranked in descending order of the scores. In each feedback iteration, AAD presents the most anomalous instance to the analyst and asks for its true label, either *anomalous* or *nominal*. In prior work, the AAD algorithm was developed to learn the weighting among an ensemble of anomaly detectors, in particular ensembles produced by the LODA [6] anomaly detector. Here we show that the same approach can be used to re-weight nodes within the trees of a forest.

Assume that we have a dataset instance $\mathbf{H} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$, where $\mathbf{z}_i \in \mathbb{R}^M$. Note that here we think of the instances as being represented by the vector of indicator features corresponding to tree

nodes. When the label is known for an instance $\mathbf{z}_i$, we will denote the label by $y_i \in \{anomaly, nominal\}$. Let $\mathbf{H}_F \subseteq \mathbf{H}$ be the set of instances for which the analyst has already provided feedback, $\mathbf{H}_A \subseteq \mathbf{H}_F$ be the set of labeled anomalies, and let $\mathbf{H}_N \subseteq \mathbf{H}_F$ be the set of labeled nominals. The anomaly score of an instance $\mathbf{z}$ is score($\mathbf{z}$) = $\mathbf{z} \cdot \mathbf{w}$, and our goal is to learn the weights w that will most likely rank the true anomalies near the top.

The AAD algorithm takes a quantile parameter as input $\tau \in [0, 1]$. The instance that has the τ-th ranked score (in descending order) is denoted by $\mathbf{z}_\tau$, and its corresponding score is denoted by q_τ. The weight vector $\mathbf{w}$ must ensure that scores of labeled anomalies $\mathbf{z} \in \mathbf{H}_A$ are higher than q_τ while, at the same time, the scores of labeled nominals $\mathbf{z} \in \mathbf{H}_N$ are lower than q_τ. Additionally, AAD adds soft pairwise constraints which encourage every labeled anomaly to have a higher score than every labeled nominal under the new weights that are learned.

The weight vector $\mathbf{w}$ is learned through a constrained optimization problem (described below). This problem is the same as the one introduced for the original AAD algorithm [3], except for the following differences:

(1) Instead of introducing all pairwise constraints between anomalies and nominals, we only add constraints relative to the current τ-th ranked instance. We found that this change does not degrade the accuracy of AAD in detecting anomalies, but makes the computation significantly faster.
(2) Since the pairwise constraints are 'soft', each violated constraint is multiplied by a slack penalty term C_ξ. We can then re-formulate the objective by adding additional terms to the loss function that correspond to the constraints. This allows optimization by gradient descent, which is helpful when the number of features is very high — as will be the case in our proposed algorithm.

Before formulating the optimization problem, we first define the following hinge loss $\ell(q, \mathbf{w}; (\mathbf{z}_i, y_i))$:

$$\ell(q, \mathbf{w}; (\mathbf{z}_i, y_i)) = \begin{cases} 0 & \mathbf{w} \cdot \mathbf{z}_i \geq q \text{ and } y_i = \text{'anomaly'} \\ 0 & \mathbf{w} \cdot \mathbf{z}_i < q \text{ and } y_i = \text{'nominal'} \\ (q - \mathbf{w} \cdot \mathbf{z}_i) & \mathbf{w} \cdot \mathbf{z}_i < q \text{ and } y_i = \text{'anomaly'} \\ (\mathbf{w} \cdot \mathbf{z}_i - q) & \mathbf{w} \cdot \mathbf{z}_i \geq q \text{ and } y_i = \text{'nominal'} \end{cases} \quad (1)$$

The modified unconstrained optimization problem for learning the optimal weights is then formulated as:

$$\begin{aligned} \mathbf{w}^{(t)} = \arg\min_{\mathbf{w}, \xi} & \frac{C_A}{|\mathbf{H}_A|} \left(\sum_{\mathbf{z}_i \in \mathbf{H}_A} \ell(\hat{q}_\tau(\mathbf{w}^{(t-1)}), \mathbf{w}; (\mathbf{z}_i, y_i)) \right) \\ & + \frac{1}{|\mathbf{H}_N|} \left(\sum_{\mathbf{z}_i \in \mathbf{H}_N} \ell(\hat{q}_\tau(\mathbf{w}^{(t-1)}), \mathbf{w}; (\mathbf{z}_i, y_i)) \right) \\ & + \frac{C_\xi}{|\mathbf{H}_A|} \left(\sum_{\mathbf{z}_i \in \mathbf{H}_A} \ell(\mathbf{z}_\tau^{(t-1)} \cdot \mathbf{w}, \mathbf{w}; (\mathbf{z}_i, y_i)) \right) \\ & + \frac{C_\xi}{|\mathbf{H}_N|} \left(\sum_{\mathbf{z}_i \in \mathbf{H}_N} \ell(\mathbf{z}_\tau^{(t-1)} \cdot \mathbf{w}, \mathbf{w}; (\mathbf{z}_i, y_i)) \right) \\ & + \|\mathbf{w} - \mathbf{w}_p\|^2 \end{aligned} \quad (2)$$

where, $\mathbf{w}_p = \frac{\mathbf{w}_U}{\|\mathbf{w}_U\|} = [\frac{1}{\sqrt{m}}, \ldots, \frac{1}{\sqrt{m}}]^T$, $\mathbf{z}_\tau^{(t-1)}$ and $\hat{q}_\tau(\mathbf{w}^{(t-1)})$ are computed by ranking anomaly scores with $\mathbf{w} = \mathbf{w}^{(t-1)}$. C_A and C_ξ are constant weight hyper-parameters. When C_A is set to a value larger than 1, as is typically the case, it causes the hinge loss for anomalies in $\mathbf{H}_A$ to be higher than those associated with nominals. C_ξ encourages a) the scores of anomalies in $\mathbf{H}_A$ to be higher than that of the τ-th ranked instance from the previous iteration, and b) the scores of nominals in $\mathbf{H}_N$ to be lower than that of the τ-th ranked instance from the previous iteration.

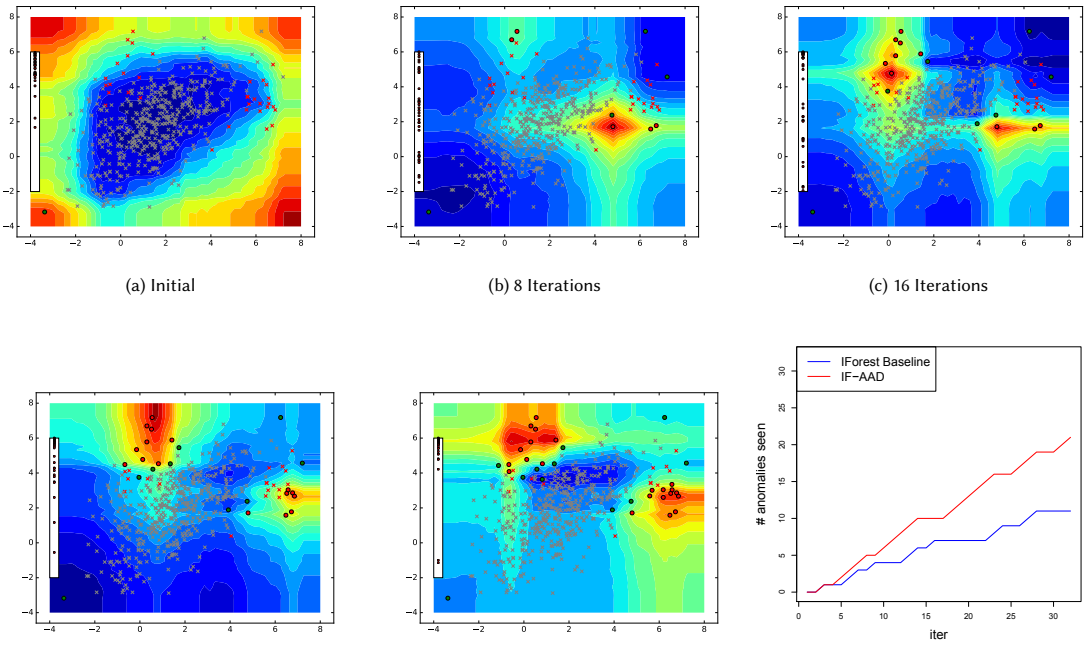
We apply gradient descent to learn the optimal weights $\mathbf{w}$ for Equation 2, in Line 15 of Algorithm 2.

Algorithm 2 Active Anomaly Discovery (AAD)

Input: Dataset $\mathbf{H}$, budget B
Initialize the weights $\mathbf{w}^{(0)} = \{\frac{1}{\sqrt{m}}, ..., \frac{1}{\sqrt{m}}\}$
Set $t = 0$
Set $\mathbf{H}_A = \mathbf{H}_N = \emptyset$
while $t \leq B$ **do**
  $t = t + 1$
  Set $\mathbf{a} = \mathbf{H} \cdot \mathbf{w}$ (i.e., $\mathbf{a}$ is the vector of anomaly scores)
  Let $\mathbf{z}_i$ = instance with highest anomaly score (where $i = \arg\max_i(a_i)$)
  Get feedback {'*anomaly*'/'*nominal*'} on $\mathbf{z}_i$
  if $\mathbf{z}_i$ is *anomaly* **then**
    $\mathbf{H}_A = \{\mathbf{z}_i\} \cup \mathbf{H}_A$
  else
    $\mathbf{H}_N = \{\mathbf{z}_i\} \cup \mathbf{H}_N$
  end if
15: $\mathbf{w}^{(t)}$ = compute new weights; normalize $\|\mathbf{w}^{(t)}\| = 1$
end while

3.2 Re-weighting IF Partitions (IF-AAD)

Our experiments consider starting with IF and tuning the weights based on feedback. This can simply be done by initializing the weights to all be constant values. The AAD algorithm can then be employeed with a regularization term that encourages weights to not depart too far from those initial values. We will refer to



(a) Initial (b) 8 Iterations (c) 16 Iterations

(d) 24 Iterations (e) 32 Iterations (f) Anomalies discovered

Figure 2: Incorporating feedback in Isolation Forest (IF) for synthetic data (Figure 1b). Figures 2a – 2e show anomaly score contours in the same way as explained in Figure 1. The red and green circles are the instances that have been presented for labeling. The x-axis in Figure 2f represents the number of instances presented to the analyst, and the y-axis represents the number of true anomalies discovered. The red curve in Figure 2f shows the number of true anomalies discovered when we incorporate feedback; the blue curve in Figure 2f shows the number of true anomalies discovered when no feedback was incorporated.

this algorithm as *IF-AAD*. We assume that the forest is constructed exactly as in the original IF algorithm and the trees are kept fixed throughout the entire interaction with the analyst. That is, the feedback is employed only to re-weight the tree-partitions; the partitions themselves are never modified.

Figure 2 shows the result of incorporating feedback on the synthetic data. As the algorithm receives feedback, it alters the contours of the anomaly scores and focuses on the more relevant regions of the feature space. In all experiments we have set the number of trees $t = 100$. For the AAD parameters, we set $\tau = 0.03$, and $C_A = 100$, as recommended in Das et al. [3]. We set $C_\xi = 0.001$ in all experiments. A very large C_ξ makes the algorithm focus more on regions where anomalies have already been found previously, and discourages exploration.

4 EXPERIMENTS

In our experiments, we used the *Mammography* [10] dataset as well as seven datasets from the UCI repository [1]: *Abalone*, *Cardiotocography*, *Thyroid (ANN-Thyroid)*, *Forest Cover (Covtype)*, *KDD-Cup-99*, *Shuttle* and *Yeast*. For each dataset, the classes were divided into two sets, one representing the nominal instances and a smaller set representing the anomlous instances. For the *Cardiotocography* dataset, we retained all instances from the *nominal* class as in the original dataset, but down-sampled the *anomaly* instances so that they represent only around 2% of the total data. The rest of the datasets were used in their entirety. The number of true anomalies and true nominals in each dataset along with the division of classes into nominals and anomalies are shown in Table 2.

We evaluate an anomaly detector based on the rate that a simulated analyst is able to find true anomalies. In particular, each iteration of anomaly detection involves giving the analyst the top ranked instance and then receiving the feedback as *anomalous* or *nominal*. We compare our proposed algorithm, IF-AAD, against the following baselines:

(1) **IForest Baseline:** For the baseline, we present instances in decreasing order of anomaly score computed with the IF algorithm with uniform weights. This algorithm ignores the analyst feedback and thus the ranking is constant across iteration. This baseline captures the performance of an unsupervised anomaly detector that does not incorporate expert feedback. The trees were constructed by the original IF implementation available as part of the *Python scikit-learn* library.
(2) **LODA-AAD:** This corresponds to the original AAD approach [3], where AAD was applied to the ensemble of anomaly detectors created by the LODA anomaly detector [6]. Each anomaly detector in the ensemble corresponds to a random

Table 2: Datasets used in our experiments, along with their characteristics.

Dataset	Nominal Class	Anomaly Class	Total	Dims	# Anomalies(%)
Abalone	8, 9, 10	3, 21	1920	9	29 (1.5%)
ANN-Thyroid-1v3	3	1	3251	21	73 (2.25%)
Cardiotocography	1 (Normal)	3 (Pathological)	1700	22	45 (2.65%)
Covtype	2	4	286048	54	2747 (0.9%)
KDD-Cup-99	*'normal'*	*'u2r', 'probe'*	63009	91	2416 (3.83%)
Mammography	-1	+1	11183	6	260 (2.32%)
Shuttle	1	2, 3, 5, 6, 7	12345	9	867 (7.02%)
Yeast	*CYT, NUC, MIT*	*ERL, POX, VAC*	1191	8	55 (4.6%)

projection that maps each instance to 1D, bins the data to form a histogram, and then measures the anomaly score according to frequency of the histogram bin an instance falls into.

Figure 3 shows the quantitative results for all of the data sets. Each graph plots the number of discovered anomalies versus the number of iterations. The best possible result is a line with slope 1, indicating that an anomaly is discovered at each iteration. The curves are averaged over 10 independent runs of the algorithm and 95% confidence intervals are shown. Overall, we see that IF-AAD never hurts the performance of IF and in most cases significantly increases the number of anomalies discovered over time compared to both IF and LODA-AAD.

In order to gain more insight into how the feedback influences the algorithm on real-world datasets, we computed the two-dimensional representations of the datasets with *t-SNE* [9] for visualization. Figure 4 shows the t-SNE plots of two representative datasets, *Abalone* and *ANN-Thyroid-1v3*. We then marked the points on which the algorithm focused its queries in the first 60 feedback iterations. We observe two ways by which the feedback influenced the queries. First, it reduced focus on the regions where the queried outliers were labeled nominal (e.g., location $(30, -50)$ in *Abalone*, and $(60, -60)$ in *ANN-Thyroid-1v3*). Second, it increased focus on regions that contained previously labeled true anomalies (e.g., $(-20, -20)$ in *Abalone* and $(0, -10)$ in *ANN-Thyroid-1v3*).

The time taken by IF-AAD in each feedback iteration depends on the particular data set and increases linearly with the number of labeled instances. As an example, for *ANN-Thyroid-1v3*, IF-AAD took less than one second for the first feedback which involved one labeled instance, and took approx. 40 seconds to incorporate 100 labeled instances.

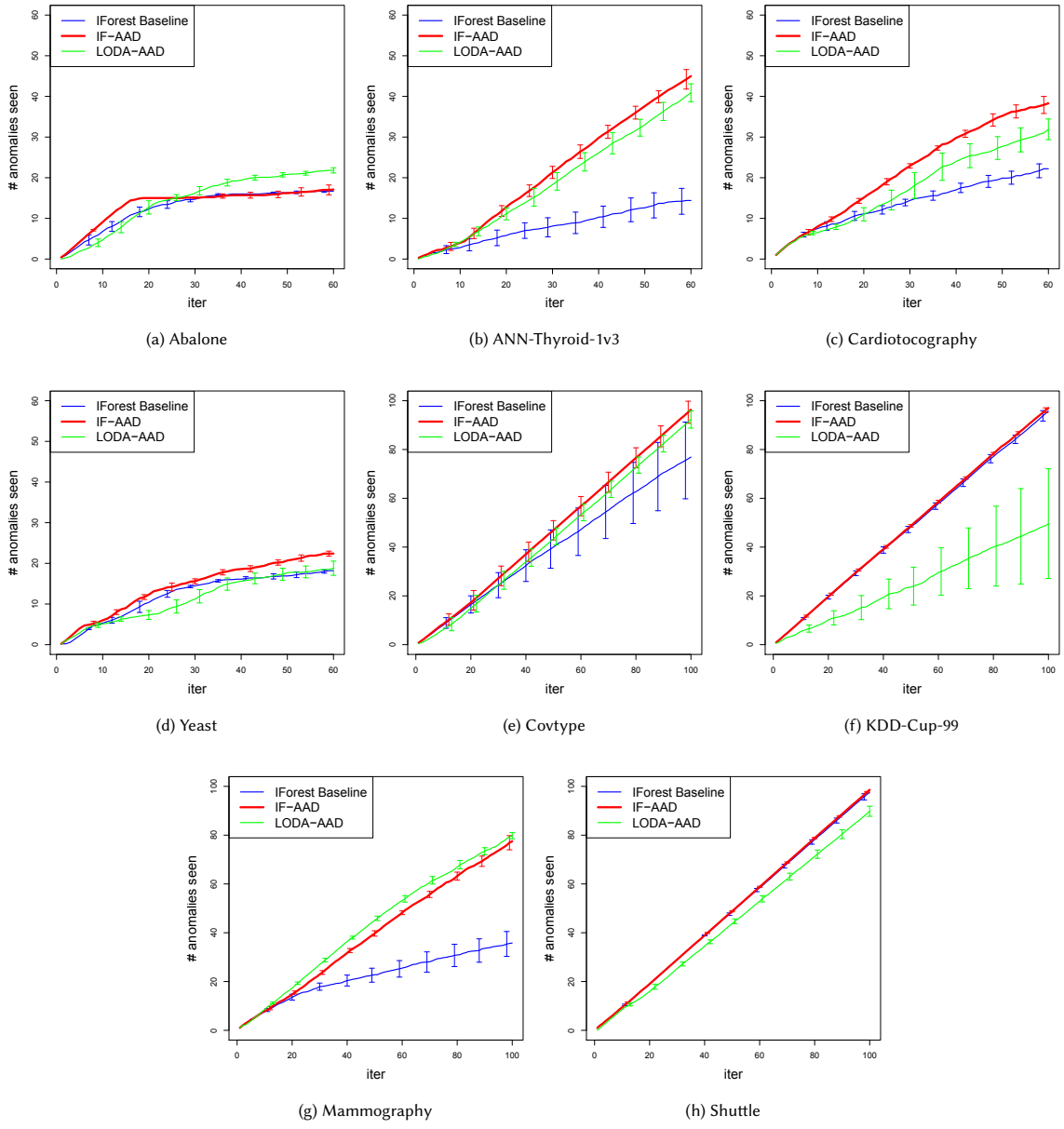
Finally, we note that a number of tree-based anomaly detectors are based on having non-zero weights only at the leaves (see Table 1). In order to evaluate the importance of having non-zero weights on internal nodes, we evaluated a version of IF-AAD that keeps all weights equal to zero except for the leaf nodes, which are updated by AAD. This new algorithm is called IF-AAD-Leaf and is implemented by only including indicator features and weights for leaf nodes in our formulation. Figure 5 shows a comparison between IF-AAD and IF-AAD-Leaf on three data sets that are representative of the results across all data sets. We observed that IF-AAD-Leaf has slightly worse performance than IF-AAD, showing that there is utility in weighting internal nodes, but the majority of the impact of feedback can be achieved by focusing just on leaf nodes.

5 SUMMARY

We presented a new anomaly detection algorithm, IF-AAD, which fine-tunes the output of an Isolation Forest in a feedback loop. It treats the regions defined by the nodes of the isolation trees as components of an ensemble and re-weights them on the basis of feedback received from an analyst. IF-AAD is consistently one of the top performers in our experiments with real-world data. It sometimes detects twice the number of true anomalies as the baseline isolation forest algorithm. In future work we intend to extend our approach to other tree-based anomaly detectors.

ACKNOWLEDGMENTS

Funding was provided by Defense Advanced Research Projects Agency Contracts W911NF-11-C-0088 and FA8650-15-C-7557. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on. This paper is based upon work while Weng-Keen Wong was serving at the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.



(a) Abalone (b) ANN-Thyroid-1v3 (c) Cardiotocography

(d) Yeast (e) Covtype (f) KDD-Cup-99

(g) Mammography (h) Shuttle

Figure 3: The total number of true anomalies seen vs. the number of queries for all datasets. Total number of queries for the smaller datasets (*Abalone*, *Cardiotocography*, *ANN-Thyroid-1v3*, and *Yeast*) is 60. Total number of queries for the larger datasets (*Covtype*, *KDD-Cup-99*, *Mammography*, and *Shuttle*) is 100. Results were averaged over 10 runs. The error-bars represent 95% confidence intervals.

REFERENCES

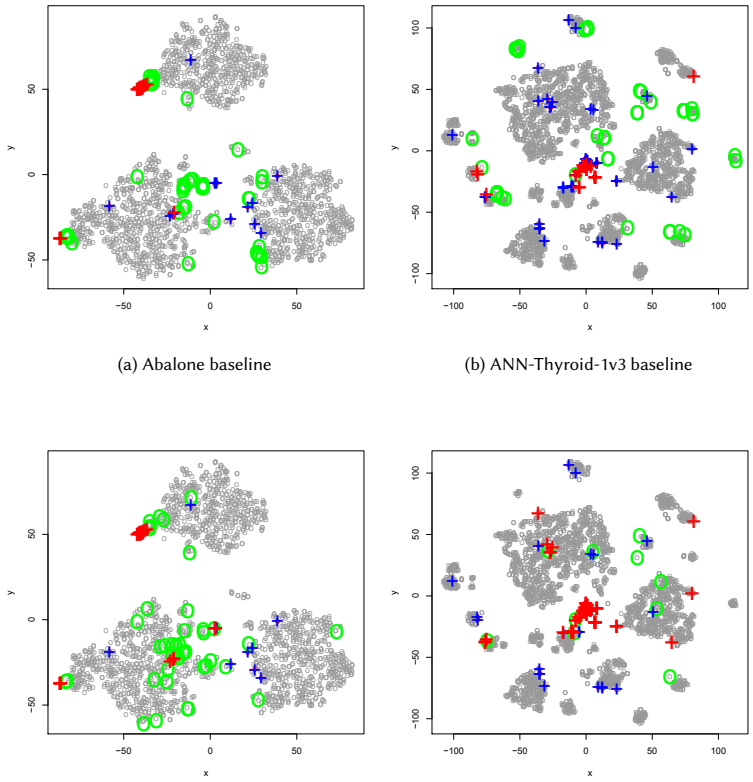
[1] 2007. UC Irvine Machine Learning Repository. http://archive.ics.uci.edu/ml/. (2007).

[2] Fan Chen, Zicheng Liu, and Ming-ting Sun. 2015. Anomaly detection by using Random Projection Forest. In *2015 IEEE International Conference on Image Processing (ICIP)*. 1210–1214.

[3] Shubhomoy Das, Weng-Keen Wong, Thomas G. Dietterich, Alan Fern, and Andrew Emmott. 2016. Incorporating Expert Feedback into Active Anomaly Discovery. In *Proceedings of the IEEE International Conference on Data Mining*. 853–858.

[4] Andrew Emmott, Shubhomoy Das, Thomas G. Dietterich, Alan Fern, and Weng-Keen Wong. 2015. Systematic Construction of Anomaly Detection Benchmarks from Real Data. *CoRR* abs/1503.01158 (2015). http://arxiv.org/abs/1503.01158

[5] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *Proceedings of the Eighth IEEE International Conference on Data Mining*. 413–422.



(a) Abalone baseline

(b) ANN-Thyroid-1v3 baseline

(c) Abalone IF-AAD

(d) ANN-Thyroid-1v3 IF-AAD

Figure 4: Low-dimensional visualization of *Abalone* and *ANN-Thyroid-1v3* using t-SNE. Plus signs are anomalies and circles are nominals. A red coloring indicates that a true anomaly point was queried. A green indicates a nominal point was queried. Grey circles correspond to unqueried nominals. To make unqueried anomalies stand out visually, we indicate them with blue plus signs.

[6] Tomáš Pevny. 2016. Loda: Lightweight On-line Detector of Anomalies. *Mach. Learn.* 102, 2 (Feb. 2016), 275–304.

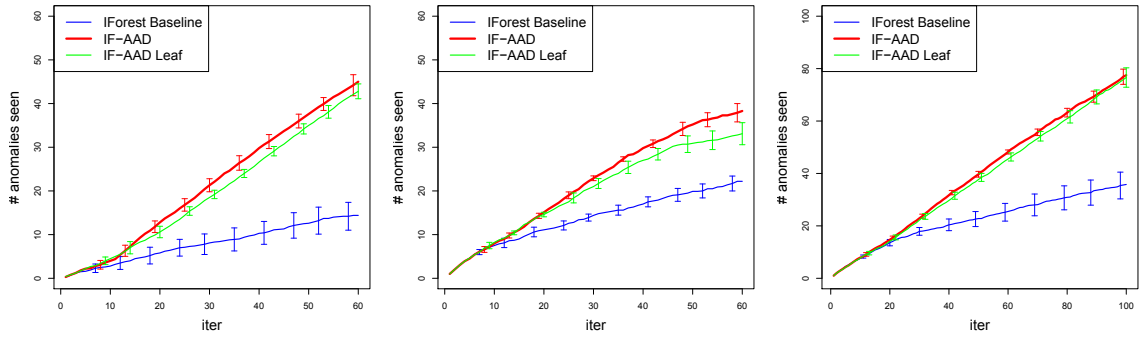
[7] Md Amran Siddiqui, Alan Fern, Thomas Dietterich, and Shubhomoy Das. 2016. Finite Sample Complexity of Rare Pattern Anomaly Detection. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

[8] Swee Chuan Tan, Kai Ming Ting, and Tony Fei Liu. 2011. Fast Anomaly Detection for Streaming Data. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Two*. 1511–1516.

[9] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. (2008), 2579–2605 pages.

[10] Kevin S. Woods, Christopher C. Doss, Kevin W. Bowyer, Jeffrey L. Solka, Carey E. Priebe, and W. Philip Kegelmeyer. 1993. Comparative Evaluation of Pattern Recognition Techniques for Detection of Microcalcifications in Mammography. *International Journal of Pattern Recognition and Artificial Intelligence* 07, 06 (1993), 1417–1436.

[11] Ke Wu, Kun Zhang, Wei Fan, Andrea Edwards, and S Yu Philip. 2014. Rs-forest: A Rapid Density Estimator for Streaming Anomaly Detection. In *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 600–609.



(a) ANN-Thyroid-1v3 (b) Cardiotocography (c) Mammography

Figure 5: Comparison between assigning weights only at the leaf nodes (IF-AAD Leaf), and assigning weights at both the leaf and the intermediate nodes (IF-AAD). The curves show the total number of true anomalies seen vs. the number of queries. The weight at each leaf node in *IF-AAD Leaf* was set to be the negative of the path length from the root, while the intermediate nodes were ignored. The weight at each node in *IF-AAD* (leaf and intermediate) was set to -1.

Foresight: Recommending Visual Insights

Çağatay Demiralp
IBM Research
cagatay.demiralp@us.ibm.com

Peter J. Haas
IBM Research
phaas@us.ibm.com

Srinivasan Parthasarathy
IBM Research
spartha@us.ibm.com

Tejaswini Pedapati
IBM Research
tejaswinip@us.ibm.com

ABSTRACT

Current tools for exploratory data analysis (EDA) require users to manually select data attributes, statistical computations and visual encodings. This can be daunting for large-scale, complex data. We introduce Foresight, a system that helps the user rapidly discover visual insights from large high-dimensional datasets. Formally, an "insight" is a strong manifestation of a statistical property of the data, e.g., high correlation between two attributes, high skewness or concentration about the mean of a single attribute, a strong clustering of values, and so on. For each insight type, Foresight initially presents visualizations of the top k instances in the data, based on an appropriate ranking metric. The user can then look at "nearby" insights by issuing "insight queries" containing constraints on insight strengths and data attributes. Thus the user can directly explore the space of insights, rather than the space of data dimensions and visual encodings as in other visual recommender systems. Foresight also provides "global" views of insight space to help orient the user and ensure a thorough exploration process. Furthermore, Foresight facilitates interactive exploration of large datasets through fast, approximate sketching.

KEYWORDS

Exploratory data analysis, guided data exploration, recommendation, statistical, insight, visualization, sketching, scalable analytics

This paper was previously accepted for publication at the demo track of VLDB'17. The original draft can be accessed at: https://arxiv.org/pdf/1707.03877.pdf.

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), August 14th, 2017, Halifax, Nova Scotia, Canada
2017. ACM ISBN .
https://doi.org/

Portable In-Browser Data Cube Exploration

Kareem El Gebaly, Lukasz Golab, and Jimmy Lin
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
[kareem.elgebaly,lgolab,jimmylin]@uwaterloo.ca

ABSTRACT

Data cubes, which summarize data across multiple dimensions, have been a staple of On Line Analytical Processing (OLAP) for well over a decade. While users typically access data cubes through data warehouse systems or business intelligence tools, we demonstrate that data cubes can be explored effectively and efficiently *inside a browser*. We provide an overview of the two recent technologies that enable our portable data cube exploration approach: 1) Afterburner, an in-browser relational database management system, and 2) explanation tables, an information-theoretic technique for guided data cube exploration.

1 INTRODUCTION

Since their introduction [6], data cubes have become a staple of On Line Analytical Processing (OLAP) and decision support. Given a dataset with multiple *dimension attributes* and one or more *measure attributes*, data cubes compute aggregate functions of the measure attribute over all subsets of the dimension attributes. Users typically explore data cubes by selecting different subsets of dimension attributes and viewing the resulting aggregates: e.g., total sales by store, total sales by product type, total sales by day, total sales by store and product type, etc.

Data cubes may be very large; e.g., millions of distinct products, multiplied by hundreds of stores, multiplied by hundreds of days, etc. Typically, data warehouse systems and business intelligence tools allow users to "start small" and zoom in (i.e., drill down) to different dimensions; e.g., a user may start by viewing total sales and then view a breakdown of sales by store.

We ask the following question: *can data cube exploration be performed effectively and efficiently inside a browser?* There are several compelling reasons for doing this. As evidenced by tools such as Jupyter Notebook, which integrate code, output, and visualization, the browser is no longer a dumb rendering endpoint and has become the de-facto front end for data science applications. It is therefore reasonable to ask if the browser can also eliminate the need to maintain a local data management system or to obtain access to a remote database server, at least for some types of tasks and users. Furthermore, in keeping with the recent trend of *data democratization*, in-browser analytics can facilitate *data analysis democratization* as a cross-platform, easy-to-share (across users and organizations) data analysis framework for non-expert users such as journalists.

In this paper, we show that the answer to the above question is yes, at least for moderately-sized datasets that fit in the browser's memory. We demonstrate our portable in-browser data cube exploration tool and explain its design, which leverages two recent technologies:

(1) **Afterburner:** an in-browser relational database management system (RDBMS) recently demonstrated at the SIGMOD conference [4]. Afterburner is implemented in JavaScript and runs inside a browser with no external dependencies, taking advantage of column-oriented storage using typed arrays and query compilation into asm.js, a strictly-typed and easy to optimize subset of JavaScript. On modestly-sized datasets, the performance of Afterburner was shown to be similar to that of the columnar RDBMS MonetDB [4] on the well-known TPC-H benchmark.
(2) **Explanation tables:** an information-theoretic technique for explaining a measure attribute using combinations of dimension attributes [3], which, as we will demonstrate, provides a useful starting point for interactive data cube exploration.

The remainder of this paper is organized as follows: Section 2 gives an overview of data cubes and the information-theoretic cube exploration framework we use. In Section 3, we explain the implementation details of our data cube exploration tool built on top of Afterburner. Section 4 presents an outline of our demonstration, Section 5 discusses related work, and Section 6 concludes the paper with directions for future work.

2 DATA CUBE EXPLORATION

2.1 Illustrative Example

We illustrate data cubes and their exploration with a simple example. Suppose we have collected a dataset from smart refrigerators indicating which food items were eaten while they were still fresh, before their expiry dates, and which ones were expired and had to be thrown away. The dataset is shown in Table 1. Each row consists of a numeric `id` which serves as a key but is not relevant for this example, and the following three dimension attributes: an item `name`, the `season` when the item was stored, and the `location` of the refrigerator. Additionally, the binary measure attribute `expires` identifies items which were expired (in general, measure attributes can be numeric).

Suppose we want to understand the reasons why some food items are consumed and some expire: is it the item type, the season, the location, or some combination of these? To do so, we compute a data cube over Table 1, as shown in Table 2. The two aggregate functions are count and average of the `expires` values, denoting how likely a subset of items is to expire. The first

IDEA'17, August 14th, 2017, Halifax, Nova Scotia, Canada

Table 1: Example dataset

id	item	season	location	expires?
1	Cheese	Winter	Kitchen	0
2	Cherries	Summer	Summer house	1
3	Chocolate	Summer	Summer house	0
4	Chocolate	Spring	Bedroom	0
5	Chocolate	Winter	Office	0
6	Chocolate	Summer	Basement	0
7	Chocolate	Fall	Winter house	0
8	Eggs	Fall	Kitchen	1
9	Eggs	Winter	Winter house	1
10	Juice	Spring	Office	0
11	Milk	Spring	Office	1
12	Milk	Summer	Winter house	1
13	Veggies	Spring	Summer house	1
14	Veggies	Winter	Winter house	1

Table 2: Fragments of a data cube over the example dataset

id	item	season	location	count	AVG(exp.)
1	*	*	*	14	0.5
2	Cheese	*	*	1	0
3	Cherries	*	*	1	1
9	*	Winter	*	4	0.5
10	*	Summer	*	4	0.5
13	*	*	Kitchen	2	0.5
14	*	*	Bedroom	1	0
19	Cheese	Winter	*	1	0
20	Chocolate	Summer	*	2	0
32	Cheese	*	Kitchen	1	0
33	Eggs	*	Kitchen	1	0
46	*	Winter	Kitchen	1	0
47	*	Spring	Office	2	0.5
57	Cheese	Winter	Kitchen	1	0
58	Chocolate	Spring	Bedroom	1	0

row of the data cube corresponds to the values of the two aggregates over the entire dataset, with stars denoting all possible values. Row ids 2 and 3 correspond to an SQL `GROUP BY` query over the `item` column; row ids 9 and 10 to `GROUP BY season`; row ids 13 and 14 to `GROUP BY location`; row ids 19 and 20 to `GROUP BY item, season`, and so on. The entire data cube consists of 70 rows and corresponds to a union of aggregation queries over all possible subsets of dimension attributes.

Two fundamental data cube operations are *drill down* and *roll up*, corresponding to adding or removing a dimension attribute, respectively. For example, starting from row id 1, we can drill down into the `item` dimension attribute and obtain more details about different items. Conversely, starting, say, from row id 19, we can roll up the `item` attribute, leaving only `season` (row ids 9 and 10).

2.2 Explanation Tables

Since data cubes may be very large, a useful data exploration technique is to identify interesting or informative parts of a data cube

Table 3: An explanation table over the example dataset

item	season	location	count	AVG(expires)
*	*	*	14	0.5
Chocolate	*	*	5	0
*	*	Winter house	4	0.75
*	*	Summer house	3	0.67

that users should examine. One such technique is the *explanation table* [3], which identifies rows from the data cube that provide the most information about the distribution of the measure attribute.

Table 3 shows an explanation table over our example dataset. Each row of an explanation table is a row from the data cube and includes the aggregates computed over the measure attributes; from now on, we refer to explanation table rows as *patterns*. The first pattern states that, on average, half of the items in the dataset have expired. The second pattern states that none of the five chocolates have expired and is included in the explanation table because it provides the most additional information about the distribution of the measure attribute. Intuitively, this is because chocolates are far less likely to expire than other products, and there are sufficiently many chocolates in the dataset. The third and fourth patterns, respectively, indicate that items located in the winter house and the summer house are more likely to expire. Again, they are included because they provide the most additional information about the distribution of the measure attribute.

An explanation table provides a useful starting point for data cube exploration: each of its patterns is *informative* and may be explored further by the user for additional insight. For example, the second pattern in Table 3 reveals that chocolates tend not to expire. The user may then drill down by `season` and/or `location` to see if chocolates purchased in different seasons or stored in different locations are more or less likely to expire. This is exactly how we leverage explanation tables in our data cube exploration tool.

2.3 Constructing Explanation Tables

We now give a brief overview of the algorithm for constructing informative explanation tables; see El Gebaly et al. [3] for full details.

The idea is to maintain *maximum-entropy* estimates for the values of the measure attribute and refine them as new patterns are added to the explanation table. In each iteration of the algorithm, we greedily add a pattern to the explanation table that gives the greatest information gain, more precisely, the greatest reduction in the *Kullback-Leibler Divergence* between the actual measure values and the estimated ones. We stop after k iterations, where k is a user-supplied parameter, yielding an explanation table with k patterns.

Consider Table 3. Based on the first pattern (the all-stars pattern), the maximum-entropy estimate of the true `expires` values is to set each value (of each of the 14 rows in Table 1) to 0.5. Of course, the actual `expires` values are binary, but we allow the estimates to be real numbers between zero and one. This maximum-entropy estimate only uses the information implied by the first pattern of the explanation table, which is that `AVG(expires)=0.5`, without making any other assumptions.

As it turns out, the pattern (Chocolate,*,*) is then added to the explanation table because it provides the greatest information gain. As a result, we update the estimates of `expires` as follows. Since the second pattern implies that all chocolates have `expires`=0, we set the estimates for rows 3 through 7 to zero. Next, for consistency with the first pattern, which requires the average value of `expires` over the whole dataset to be 0.5, we must set the estimates for all non-chocolate rows to $\frac{7}{9}$ each. This gives us a maximum-entropy estimate that only considers the information contained in the first two patterns of the explanation table. As in El Gebaly et al. [3], we use *iterative scaling* to compute updated estimates whenever a new pattern is added to the explanation table.

To summarize, the greedy algorithm for constructing an explanation table works as follows. We iterate k times, once for each pattern. In each iteration, we 1) compute the information gain of a set of candidate patterns, 2) add to the explanation table the pattern with the greatest gain, and 3) update the maximum-entropy estimates. To compute the information gain in step 1), we build a data cube with the average of the *estimated* measure attribute as the aggregate function, and compare the estimates with the actual values. For efficiency, our implementation uses sampling to compute these data cubes, following El Gebaly et al. [3].

3 IN-BROWSER IMPLEMENTATION

At a high level, our data cube exploration tool issues and consumes the output of SQL queries executed by the Afterburner RDBMS. Both explanation table construction and subsequent drill-down into individual patterns are accomplished with a series of SQL group-by/aggregation queries. Our implementation handles interactive data cube exploration of moderately-sized datasets (around 10 million records) in sub-second time.

Afterburner (and our tool running on top of it) is implemented as a JavaScript library and runs completely stand-alone inside a browser. Datasets are loaded from the local file system or from a remote server. Once loaded, data are immutable and stored in memory in column-oriented format. As discussed below, Afterburner exploits two JavaScript features: typed arrays for memory-efficient storage and asm.js for fast compiled queries.

3.1 Columnar Storage Using Typed Arrays

Array objects in JavaScript can store elements of any type and are not arrays in a traditional sense (compared to say, C) since consecutive elements may not be contiguous; furthermore, the array itself can dynamically grow and shrink. This flexibility limits the optimizations that the JavaScript engine can perform both during compilation and at runtime. In contrast, typed arrays in JavaScript are comprised of buffers, which simply represent untyped binary data, and views, which impose a read context on the buffer.

Typed arrays allow the developer to create multiple views over the same buffer. Afterburner takes advantage of this feature to pack relational data into a columnar layout. In our implementation, each column is laid out end-to-end in the underlying buffer, which can be traversed with a view of the corresponding type. The table itself is a group of pointers to the offsets of the beginning of the data in each column. Intermediate data for query execution are also stored using typed arrays.

3.2 Query Compilation into Asm.js

In conjunction with typed arrays, Afterburner takes advantage of asm.js, a strictly-typed subset of JavaScript that is designed to be easily optimizable by an execution engine. Any JavaScript function can request validation of a block of code as valid asm.js via a special prologue directive, `use asm`, which happens when the source code is loaded. Validated asm.js code (typically referred to as an asm.js module) is amenable to ahead-of-time (AOT) compilation, in contrast to just-in-time (JIT) compilation in vanilla JavaScript. Executable code generated by AOT compilers can be quite efficient, through the removal of runtime type checks (since everything is statically typed), operation on unboxed (i.e., primitive) types, and the removal of garbage collection.

Afterburner translates SQL queries into the string representation of an asm.js module (i.e., the physical query plan), calls `eval` on the code, which triggers AOT compilation and links the module to the calling JavaScript code, and finally executes the module (i.e., executes the query plan). The typed array storing all the tables is passed into the module as a parameter, and the query results are returned by the module.

3.3 Query Operators and Materialization

Supported SQL operators include selection/filters, aggregates, group by (using hashing) and joins (also using hashing). Notably, Afterburner avoids materialization of intermediate results as much as possible in order to fit inside a web browser memory. For example, we only store record identifiers in hash tables instead of copying the values in order to minimize the memory footprint of the hash-based operators such as joins and group bys.

For data cube exploration, we need to materialize fragments of the data cube. To reduce the memory footprint, we use dictionary encoding for dimension attribute values.

4 DEMONSTRATION SCENARIOS

In our demonstration, participants will create explanation tables to help guide their data cube exploration. We will prepare several real-world datasets for the demonstration, including airline upgrades (where the goal will be to understand what makes a passenger more likely to have their seat upgraded to first class) and U.S. census data (where the goal will be to understand what makes a person more likely to earn an annual income over $50,000). Most of our demonstration datasets are downloaded from the UCI archive.[1]

Figure 1 shows a screenshot of an explanation table with $k = 5$ patterns over the census dataset. The dimension attributes include workclass, education level, marital status, occupation, relationship to the head of the household, race, sex, and country. `COUNT(*)` refers to the number of rows in the dataset covered by a pattern and `AVG(p)` is the average value of the binary indicator attribute whose value is one if the income exceeds $50,000. The column labeled `distribution` visualizes the proportion of the (binary) measure attribute values that are one (in green) vs. zero (in red).

Clicking on the `explore` link at the end of each pattern allows users to drill into the rows captured by the pattern. For example, a user may want to learn more about the third pattern, which indicates

[1] http://archive.ics.uci.edu/ml/datasets/

Explanation Table

workclass	education	status	occupation	relationship	race	sex	country	COUNT(*)	AVG(p)	distribution	Explore
*	*	*	*	*	*	*	*	32561	0.24		explore
*	*	Never-married	*	*	*	*	*	10683	0.05		explore
*	Bachelors	Married-civ-spouse	*	*	*	*	*	2768	0.67		explore
*	*	Divorced	*	*	*	*	*	4443	0.10		explore
*	*	*	Other-service	*	*	*	*	3295	0.04		explore

Figure 1: A five-pattern explanation table over the U.S. census dataset.

Control Exploration

workclass	education	status	occupation	relationship	race	sex	country
ALL	Bachelors	Married-civ-spouse	*	*	*	*	*

Figure 2: Drilling into a pattern.

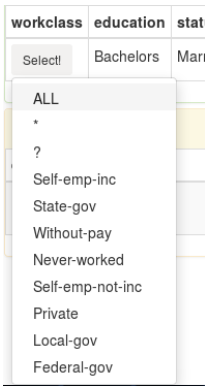


Figure 3: Manually drilling into a pattern.

that married people with a Bachelor's degree are more likely to earn a high salary. Figure 2 shows a screenshot of the corresponding exploration panel. The user can select one or more attributes to drill into. In the figure, `workclass=ALL` will compute separate aggregates for married people with a Bachelor's degree and for each workclass. Rather than selecting `ALL`, the user can also specify selected values of interest of additional attributes such as workclass. For example, Figure 3 shows a dropdown menu with all the values of workclass in the dataset.

Figure 4 shows the results of drilling into the third pattern, as specified in Figure 2. The user can now explore the distribution of the measure attribute for each workclass of married people with a Bachelor's degree.

Exploration

workclass	education	status	count(*)	avg(p)	distribution	explore
Self-emp-not-inc	Bachelors	Married-civ-spouse	265	0.51		explore
Private	Bachelors	Married-civ-spouse	1747	0.70		explore
Local-gov	Bachelors	Married-civ-spouse	220	0.62		explore
Federal-gov	Bachelors	Married-civ-spouse	105	0.72		explore
?	Bachelors	Married-civ-spouse	94	0.44		explore
Self-emp-inc	Bachelors	Married-civ-spouse	210	0.76		explore
State-gov	Bachelors	Married-civ-spouse	127	0.65		explore

Figure 4: A drill-down of the different patterns.

5 RELATED WORK

This paper is related to two bodies of work: data cube exploration and query plan compilation.

As we explained earlier, we use explanation tables for guided data cube exploration. The idea of information-theoretic data summarization initially appeared in Sarawagi et al. [13] and was then expressed in the form of explanation tables in subsequent work [3, 5]. We use the technique from El Gebaly et al. [3] in our demonstration. We note that there are other data cube exploration techniques such as smart drill-down [7], which can be added to future versions of our tool. Beyond data cubes, there is a wide variety of data exploration, data explanation, and outlier detection approaches (see, e.g., [1, 2, 11]), and it will be interesting to study whether they can be implemented efficiently in our in-browser framework.

Afterburner is based on the compiled query approach to query execution, similar to systems such as HIQUE [10], LegoBase [9, 14], Proteus [8], and HyPer [12]. Our query compilation techniques are relatively standard, with the exception of targeting JavaScript and using query plans that fit into a browser's limited memory.

6 CONCLUSIONS

In this paper, we motivated and described our tool for in-browser data cube exploration. We take advantage of modern browsers, which have become much more than dumb rendering endpoints, to provide a cross-platform, maintenance-free solution for exploring small to medium datasets. We believe that our approach is useful to "amateur data scientists" and non-expert users. In future work, we plan to enhance the effectiveness of our tool by including new data exploration techniques, and improve its efficiency by studying new optimizations within Afterburner itself.

REFERENCES

[1] Azza Abouzied, Joseph M. Hellerstein, and Avi Silberschatz. 2012. Playful Query Specification with DataPlay. *PVLDB* 5, 12 (2012), 1938–1941.

[2] Peter Bailis, Edward Gan, Kexin Rong, and Sahaana Suri. 2017. Demonstration: MacroBase, A Fast Data Analysis Engine. In *SIGMOD*. 1699–1702.

[3] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. 2014. Interpretable and Informative Explanations of Outcomes. *PVLDB* 8, 1 (2014), 61–72.

[4] Kareem El Gebaly and Jimmy Lin. 2017. In-Browser Interactive SQL Analytics with Afterburner. In *SIGMOD*. 1623–1626.

[5] Guoyao Feng, Lukasz Golab, and Divesh Srivastava. 2017. Scalable Informative Rule Mining. In *ICDE*. 437–448.

[6] Jim Gray, Adam Bosworth, Andrew Layman, and Hamid Pirahesh. 1996. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total. In *ICDE*. 152–159.

[7] Manas Joglekar, Hector Garcia-Molina, and Aditya G. Parameswaran. 2016. Interactive Data Exploration with Smart Drill-Down. In *ICDE*. 906–917.

[8] Manos Karpathiotakis, Ioannis Alagiannis, and Anastasia Ailamaki. 2016. Fast Queries Over Heterogeneous Data Through Engine Customization. *PVLDB* 9, 12 (2016), 972–983.

[9] Yannis Klonatos, Christoph Koch, Tiark Rompf, and Hassan Chafi. 2014. Building Efficient Query Engines in a High-level Language. *PVLDB* 7, 10 (2014), 853–864.

[10] Konstantinos Krikellas, Stratis Viglas, and Marcelo Cintra. 2010. Generating Code for Holistic Query Evaluation. In *ICDE*. 613–624.

[11] Arnab Nandi. 2013. Querying Without Keyboards. In *CIDR*.

[12] Thomas Neumann. 2011. Efficiently Compiling Efficient Query Plans for Modern Hardware. *PVLDB* 4, 9 (2011), 539–550.

[13] Sunita Sarawagi. 2000. User-Adaptive Exploration of Multidimensional Data. In *VLDB*. 307–316.

[14] Amir Shaikhha, Yannis Klonatos, Lionel Parreaux, Lewis Brown, Mohammad Dashti, and Christoph Koch. 2016. How to Architect a Query Compiler. In *SIGMOD*. 1907–1922.

ECOviz: Comparative Visualization of Time-Evolving Network Summaries

Lisa Jin
University of Michigan
lisajin@umich.edu

Danai Koutra
University of Michigan
dkoutra@umich.edu

ABSTRACT

How can we visualize, interact with, and 'learn' important structures of time-evolving networks? Given domain-specific attributes, such as node membership of functional brain regions, how can we use this domain knowledge to discover coherent structures and track their evolution over time? In this demo paper, we introduce ECOviz (for Evolving COmparative network visualization), a system that enables pairwise comparison of temporal graph summaries based on variations in data source and preprocessing parameters. Our system further allows the user to perform structural and temporal analysis of a graph through efficient querying and visualization of its summarizing subgraphs.

ECOviz performs the following tasks: (a) It generates a set of temporal structures for each graph of interest using a dynamic graph summarization algorithm offline; (b) It supports contrasting visual analysis of time-evolving network pairs by providing quantitative metrics on summary structure composition and temporal graph statistics; (c) It interactively visualizes the induced subgraph of each structure in a summary, at either a full time sequence or a time interval specified by the user.

In our demonstration, we invite the audience to use ECOviz to make comparisons between a variety of time-evolving functional human connectomes, and explore their salient temporal structures.

ACM Reference format:
Lisa Jin and Danai Koutra. 2017. ECOviz: Comparative Visualization of Time-Evolving Network Summaries. In *KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada, August 14th, 2017,* 8 pages.

1 INTRODUCTION

Given a set of nodes of interest, how can we improve the discovery and visualization of salient structures in a time-evolving network? The objective of summarizing such networks is to identify structures that are notable in their topology and/or recurrence over time. Showing changes over time, however, demands further knowledge of the graph's underlying structure, and perhaps calls for an application-driven approach. For visualization in particular, preserving the mental map across snapshots is desirable when following *groups* of nodes [3]. This is applicable when a user seeks to find community-level patterns within a dynamic graph.

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada

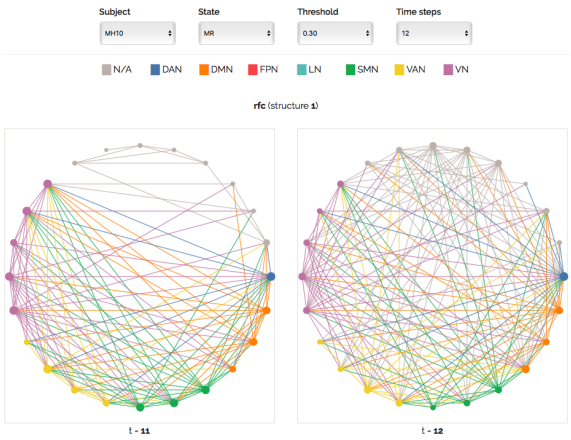


Figure 1: Visualization of temporal summary structure: ranged full clique (rfc). Resting-state sub-networks of interest are indicated by node color (e.g., orange corresponds to the default mode network 'DMN', green to the sensorimotor network 'SMN').

Tracking evolution of communities in dynamic networks, ranging from modules in protein-protein interaction networks [18] to groups in scientific co-authorship networks [4], is of high relevance for domain scientists. Especially in scientific fields such as connectomics, which explores the functional and structural connectivity of the brain, visualization is a vital tool for pattern discovery [20]. Domain scientists may lack graph drawing skills, but their expertise on the data at hand can be used to augment automatic graph analysis and layout algorithms. How can we pair the specificity of domain expertise with the objectivity of graph summarization output to depict the structure and evolution of dynamic graphs? Instead of communicating results outside of the problem context, we respond in the domain-specific 'language' of the user.

In this demo paper, we introduce ECOviz, a system that supports interactive, comparative analysis of time-evolving networks by focusing on domain-specific summaries of their most salient structures. For more holistic understanding, ECOviz also allows the user to 'zoom in' on one time-evolving network and interactively explore its discovered temporal patterns. Our system assimilates domain knowledge in the following ways:

- **Domain-specific Summarization**: To assist the discovery of coherent structures, we employ 'semi-supervision' that takes domain expertise into account early in the exploration process. This is achieved using static graph decomposition that is biased towards an *egocentric* view of high-interest nodes. Figure 1 shows one temporal pattern (full clique ranging time steps 11 and 12) in the summary of a 'mindful rest' functional network of a human subject.

- **Preprocessing-dependent Analysis**: While most real-world graphs are directly observed, many scientific domains (including neuroscience) infer graphs from measurements that are often in time series form. In response to the sheer volume of graph construction choices, we offer an interactive way to *evaluate* a comparison of preprocessing parameters. The contrasting data analysis interface includes data source selection provides complete flexibility in making inter- and intra-data comparisons.
- **Visualization of Communities**: To highlight communities of high-interest nodes (via domain-specific labels) and enable the extraction of richer insights, we provide quantitative meta-summaries of the structures and use colors to visually distinguish communities.

The paper is organized as follows. In Section 2, we introduce the application-specific data that motivated our system. Then in Section 3 we present our domain-specific graph summarization technique and in Section 4 we describe our system, ECOviz. Sections 5 and 6 give data analysis examples and our demonstration plan, respectively. Finally, Sections 7 and 8 contain related work and the conclusion.

2 CONNECTOMICS: DATA

While most real-world graphs are directly observed, functional brain networks are inferred from biological signals. Namely, blood oxygen level-dependent (BOLD) data from fMRI are a common source to computational models [9]. Fully connected, undirected graphs are typically constructed by computing the pairwise statistical dependence between all voxels (volume units of neurons). This step involves simulation over BOLD data to obtain per-voxel time series. Pearson's correlation coefficient – or another measure of association – is then computed pairwise between voxels. These values (in absolute terms) are filtered by a lower bound threshold, forming an unweighted graph.

In this demonstration, we use a dataset that consists of fMRI activity of 61 human subjects at both resting and mindful rest states. During the regular resting state (8 minutes), the thoughts of the subjects were allowed to wander about, while during the 'mindful rest' state the subjects were instructed to focus on their breath and actively not let their thoughts wander about.

Each fMRI session yields data using 100 ROI (region of interest) parcellation, each of which is accompanied by time series of length 240 timeticks (30 measurements per minute). Throughout the paper, we refer to ROIs as voxels or nodes. Out of the 100 voxels, 45 are part of resting-state networks of interest and labeled accordingly. The seven sub-networks of interest are: dorsal attention (DAN), default mode (DMN), fronto-parietal (FPN), language (LN), sensorimotor (SMN), ventral attention (VAN), and primary visual (VN) networks.

Time-evolving Graph Construction. We convert the time series per fMRI session to a time-evolving graph by extending the graph generation procedure described above. Specifically, instead of generating one connectome for the whole duration of the session (8 minutes), we split the time series into non-overlapping intervals of equal length and apply the generation process to each interval (i.e., each temporal snapshot is based on the statistical dependencies between time series during the corresponding interval). A uniform filtering threshold is applied to all the resulting networks for positive correlation values only. This leads to evolving snapshots of functional connectivity and allows us to track changes in thoughts (and their corresponding patterns) over time.

Two critical factors affect the construction of dynamic graphs: time interval granularity of the per-voxel time series and threshold value of the full association matrix. These choices can produce drastically different levels of sensitivity to noise for edge significance and aggregation [27]. As such, poorly constructed graphs can limit how well a summary captures true dynamics in the data. For instance, the full clique ranging time steps 11 and 12 in Figure 1 was found in a network formed with a threshold of 0.30 (correlation) and 12 time steps, yet its accuracy depends on how well the graph represents the subject's mindful rest state. We posit that prior knowledge of the biological signals, in the form of sub-network labels of voxels, can both indicate quality of graph construction and bolster pattern discovery in fMRI data.

3 PROPOSED METHOD: DOMAIN-SPECIFIC GRAPH SUMMARIZATION

Central to connectomics is finding novel patterns of activity between functional regions of the brain, with the goal of elucidating local and global organization. In contrast to the power law degree distribution found in many large-scale networks, the brain exhibits a small-world architecture, characterized by high local clustering and short global path lengths [9]. Superimposed on the structural tracts of the brain is a diverse, hierarchically organized functional network [21], whose typical inference was described in the introduction. We are particularly interested in mining the relatively unknown dynamics within and between specific modules (or sub-networks) of the functional network.

Current approaches in examining resting-state fMRI data include *model-dependent*, or focused on a single seed region of interest that is analyzed with respect to all other voxels, and *model-free* methods, or unsupervised techniques that include independent component analysis (ICA) [25]. While the former is simple and interpretable, it lacks the exploration of global brain patterns that the latter is capable of. To gain benefits of both methods, we use labels from resting-state networks, or functionally linked sub-networks that are highly active during rest, to inform our summarization algorithm.

We leverage TimeCrunch [22], a principled and parameter-free dynamic graph summarization algorithm. The algorithm (i) creates a set of subgraphs per static snapshot in the temporal graph; (ii) labels these subgraphs as structures based on the MDL principle (e.g., star, full clique, bipartite core); (iii) stitches static into temporal structures; and (iv) compiles a summary of top structures using again the MDL principle at the graph level. The resultant network summary consists of temporal patterns from the cross product of a *static vocabulary* that captures connectivity patterns (full clique, near clique, full bipartite core, near bipartite core, star, chain) and a *temporal vocabulary* that captures recurrence patterns (ranged, periodic, constant, flickering, oneshot). For instance, a graph may have a summary with several oneshot stars, a flickering bipartite core, and a periodic full clique.

In order to benefit from domain knowledge consisting of the nodes of interest (i.e., those belonging to specific sub-networks,

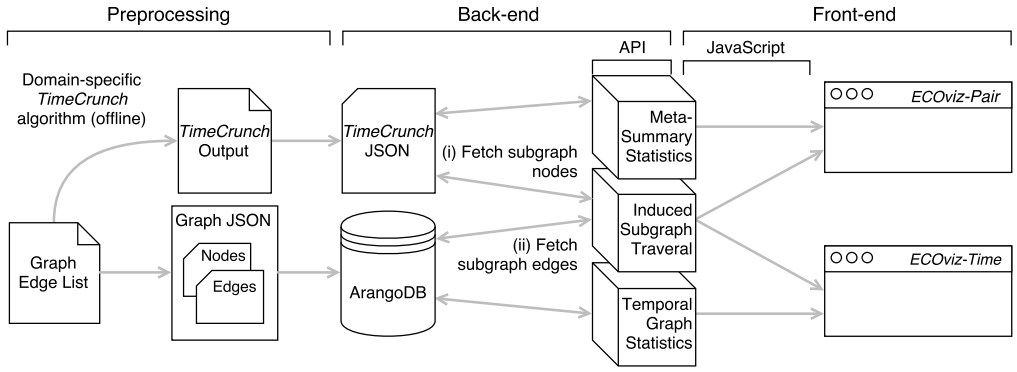


Figure 2: Full pipeline of graph summary visualization system. Major components include offline preprocessing, ArangoDB & Flask API back-end, and web interface (JavaScript) front-end.

such as the DMN in the brain), we propose a domain-specific subgraph extraction routine for the TimeCrunch [22] pipeline. Specifically, instead of using the original clustering routine of TimeCrunch, which is tailored towards real large-scale graphs with power-law degree distribution, we extract labeled nodes' *egonets*, or induced subgraphs of an *ego* node and its neighbors, as subgraphs for TimeCrunch and its static graph counterpart, VoG [15, 16]. We mainly employ egonets to simulate the model-dependent approach discussed previously, which uses seed ROIs for analysis of fMRI functional networks. Egonets also provide natural communities that partitioning algorithms targeting high-degree hub nodes – ill-suited for the small-worldness of brain networks – may overlook. Their use in analysis of heterogeneous social networks, which also have small-world properties, improved network abstraction [19].

4 SYSTEM OVERVIEW

In the following subsections, we discuss in detail the components of ECOviz. A pictorial overview of our system and its various components is given in Figure 2. Particular emphasis is placed on how resting-state network labels are utilized across the length of the entire pipeline.

4.1 Domain-specific Summarization

As described in Section 3, in place of the subgraph generation in TimeCrunch and VoG, we utilize an egocentric approach to partitioning the graph. This is achieved by using voxels of particular interest to neuroscientists as seed nodes. Specifically, the 'interesting' voxels are the ones that participate in well-known sub-networks, such as the default mode network (DMN) and other networks presented in Section 2. Irrespective of the labeled node's network of origin, we use its egonet as a subgraph input to TimeCrunch, resulting in a static total of 45 egonets per functional network. These labeled nodes are indicated in the 'labeled/total' ratio and 'entropy' columns of the summary tables (see Figures 3, 4). The former gives the number of labeled nodes per extracted egonet, and the latter is a measure of label diversity per egocentric community (e.g., a value of 1 means that the nodes are uniformly distributed among the sub-networks of interest).

For this demo, we extracted temporal summaries from 132 functional brain networks spanning: 11 human subjects, two rest states (resting and mindful rest state), and six combinations of preprocessing parameters (thresholds of $\{0.30, 0.45\}$ and time interval granularity of $\{12, 16, 24\}$) for the time-evolving graph creation.

4.2 Interactive Visualization

To support divergent modes of data analysis, the system provides two visualization views. ECOviz-Pair focuses on comparison between pairs of summaries differing in data source (i.e., subject and rest state) or preprocessing method (i.e., threshold value and time interval granularity). ECOviz-Time gives the user a more detailed narrative of how each structure evolves over time. While the views share a protocol for fetching structure connectivity, they differ in their interactivity and set of supporting features. Users interact with both views through selection of drop-down menus, each controlling a single parameter, at the top of the screen.

Preprocessing-dependent Analysis: A key feature of the system is to enable scientists to not only make inter-data comparisons, but also explore how tuning preprocessing parameters affects the summary structures found. As graph generation depends on these hyperparameters, we treat them as a set of settings that the user may toggle at will. Thus, the summarization results serve as implicit feedback about graph generation quality.

In ECOviz-Pair, we focus on the notion of summary diversity as an informal benchmark. To this end, three meta-summary charts are shown to the user: percentage of structures by structure type, node count by structure participation count, and top 10 node IDs by structure participation count (see Figure 3). The charts are displayed in a two-column format – users may either compare rest with mindful rest state of a single subject (Figure 5), or independently select parameters for each column (Figure 3).

As ECOviz-Time (Figure 4) offers a sequential view of how temporal structures evolve, we also display a chart that captures the sparsity of each functional network over time. This is intended to provide context to whether temporal changes in structure density are due to preexisting network structure. More concretely, a structure becoming denser over time could be due to preprocessing – the chosen time interval granularity may have produced networks with skewed temporal distributions of edges. For example, the

Figure 3: ECOviz-Pair contrasting data analysis view.

series of graph snapshots in Figure 4 appear to reach peak density at time step 11 (bottom-right cell), yet the network-wide sparsity chart above suggests that it is a global trend. This illustrates how

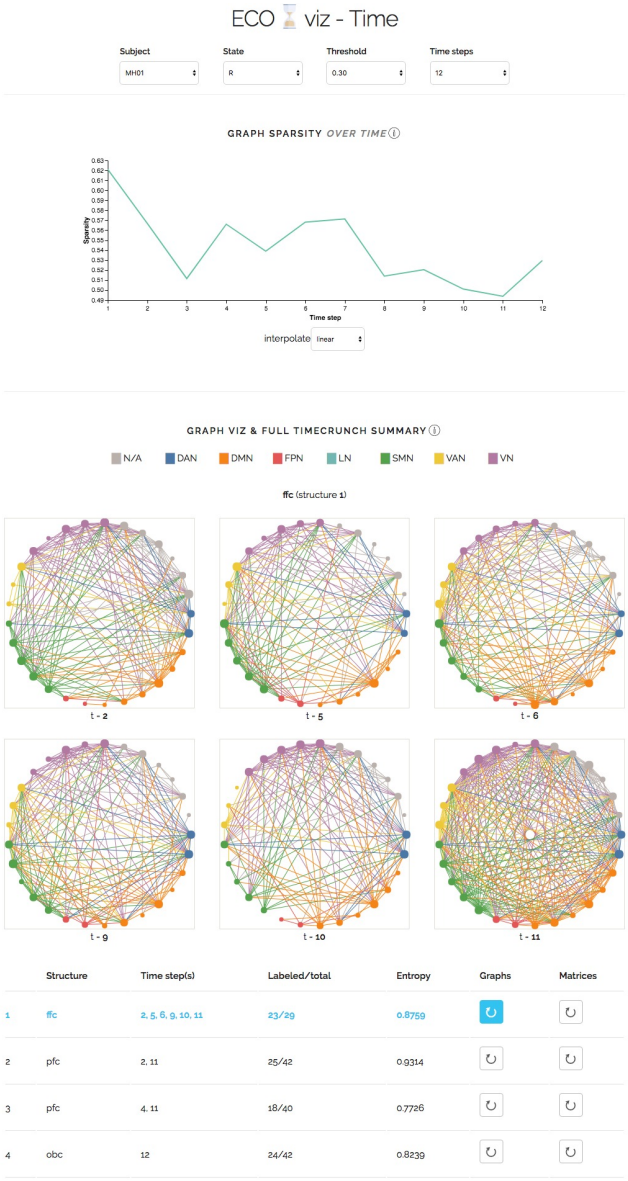


Figure 4: ECOviz-Time graph sequence of temporal summary structure: flickering full clique (ffc). Chart depicting graph sparsity over time is positioned above structure visualizations.

the temporal summary chart can highlight local, structure-specific trends in sparsity from those in the background.

Visualization of Communities: The main component of the system is a visualization of the summary structures, either at a particular time interval or a full time sequence. Since TimeCrunch mines for a predefined vocabulary of static structures, which includes cliques, bipartite cores, stars, and chains, we use this base representation of the labeled structure in the visualization. Doing so also allows the user to evaluate how well a structure's connectivity aligns with its label.

Most apparent in the visualization is the colored node representation of resting-state network labels. As the nodes within each

structure are ordered primarily by label and secondarily by node ID, ordering across time intervals is maintained. This enables the user to track a static map of the nodes across multiple time intervals, aiding in detecting edge evolution within the structure. We further apply these resting-state network labels to the adjacency matrix view of ECOviz-Time by grouping rows and columns by node label to highlight community dynamics.

4.3 System: End-to-End

Following offline execution of TimeCrunch on all functional networks, each of the summaries contains a list of structures, which describes the temporal structure type (e.g., full clique), node participation (i.e., of which nodes the structure contains), and time step participation of each structure. To depict the structure's connectivity, the system pairs node data from the summary with edge data from the original functional network. Providing real-time access to this data requires efficient storage and traversal of user-requested graphs. We chose *ArangoDB*, a multi-modal NoSQL database, as a solution to scalably fulfill these needs.

Once the system has stored network edge lists into ArangoDB and processed TimeCrunch output into JSON, the web server may begin receiving user queries. As structure visualization must support single and sequential time step requests (for the two views), a dedicated graph traversal API is utilized. The user chooses from a list of summary structures – fetched from the TimeCrunch JSON shown in Figure 2 – with their spatial and temporal properties. For each requested structure-time step pair, the graph traversal API fetches the participating node IDs from the TimeCrunch JSON (step 1 in back-end section of Figure 2). Next, the ArangoDB database is queried for the induced subgraph of these nodes (step 2 in back-end section of Figure 2). As the graph visualization JavaScript depends on this connectivity data, it makes either a single or multiple asynchronous requests (per visualization reload) to this API, depending on the front-end view.

Data flow of ECOviz-Pair and ECOviz-Time separates within JavaScript, with the links between the back-end APIs and front-end views diverging (front-end portion of Figure 2). The induced subgraph traversal API is shared among both views, while the meta-summary statistics and temporal graph statistics APIs are exclusive to ECOviz-Pair and ECOviz-Time, respectively.

5 DATA ANALYSIS: EXAMPLES

Here, we showcase the differing functions supported by the contrasting (ECOviz-Pair) and temporal (ECOviz-Time) data analysis views in terms of their functional network visualizations. The two views provide a more comprehensive glimpse at the data at hand, specific to the user's chosen purpose.

5.1 Contrasting Data Analysis

Within the ECOviz-Pair interface, there are two modes of data selection: a single set of drop-down menus controlling subject, threshold, and number of time steps, as well as a double set of menus each with an additional option for rest state (see Figure 3). The former automatically displays rest state on the left column and mindful rest on the right. The latter gives the user total control over the combinations of parameters to compare on either column. In the remainder of this subsection, we describe example use cases specific to each drop-down mode.

In the single set drop-down mode, the user is only concerned with making comparisons between resting and mindful rest states within the same subject. We demonstrate this in the pair of one-shot bipartite cores (obc) selected in Figure 5, with resting state on the left visualization and mindful rest on the right. In the resting state, there is an evident division between the SMN and other functional brain regions on either side of the bipartite core. There appears to be less segregation among nodes of the same label in the mindful rest state, which applies to both unlabeled nodes – which do not belong to sub-networks of interest – and labeled nodes in the DMN, LN, and VN. Interestingly, both rest states feature a relatively high-degree unlabeled node on the left side linking to nodes on the right. This observation could lead to further exploration of the unlabeled nodes by domain experts. As the TimeCrunch summarization algorithm is parameter-free, interpretation of trends in the summary structures depends on the user's application context.

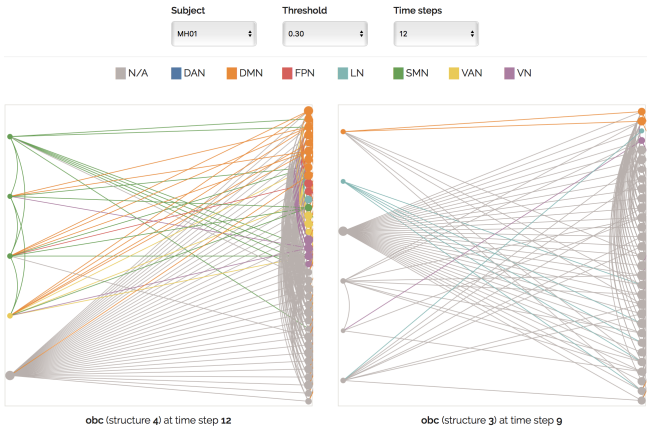


Figure 5: ECOviz-Pair temporal data analysis view of two one-time bipartite cores (obc). For the selected subject, the left structure corresponds to resting state and the right structure to mindful rest.

The double set drop-down mode, which is shown in Figure 3, offers full flexibility to the user in terms of variable(s) to compare, making it possible isolate effects of preprocessing method. The three meta-summary charts in ECOviz-Pair provide high-level insight on the structure and node composition of the TimeCrunch summaries. In Figure 3, we compare the effects of modifying temporal granularity within a single subject. On the left column, each voxel's time series is partitioned into 12 equal intervals, while the right uses 16. From the summary item proportion chart, coarser granularity results in a more temporally diverse summary – with the majority of the summary containing ranged, periodic, or flickering structures. The node count chart is more skewed right for coarser granularity, indicating more node overlap in structures. There is little variation in the top 10 node IDs, which suggests that temporal granularity has limited effect on the summary's most active nodes.

5.2 Temporal Data Analysis

As discussed in Section 4, ECOviz-Time focuses on temporal exploration of structures, showing the full sequence of time intervals that a structure appears in. Below, we detail two types of analyses that this view supports: *inter*- and *intra*-structure patterns.

The static layout of nodes across time intervals directs users' attention to trends in connectivity between structures. This is especially evident in Figure 1, where unlabeled nodes (colored in gray) shift from high intra-label connectivity to high inter-label connectivity between time steps 11 and 12. At time step 11, these unlabeled nodes are almost exclusively connected to each other, as the paucity of edges connecting them to nodes of other colors indicates. However, by time step 12, they have become heavily connected with nodes of dissimilar labels (including those within the DAN, DMN, SMN, VAN, and VN). This trend in connectivity may have been obscured in graph layouts that modify the layout of nodes between temporal structure snapshots.

Grouping nodes by label also facilitates more complex analysis of dynamics within resting-state network communities. The sequence of matrices in Figure 6 reveals how the adjacency matrix draws attention to intra-community patterns. We restrict our analysis on the four nodes within the DMN (colored in dark blue), which encompass nodes 26, 56, 75, and 91. At time steps 3 and 17, these nodes form stars centered around nodes 26 and 91, respectively. At time step 7 – between the two appearances of stars – the structure breaks into two sets of pairs (between nodes 91-26, 56-75). Finally, the DMN structure becomes a triangle at time step 19 (between nodes 75, 26, 91) and continues as a triangle at time step 20, though with a voxel replacement (between nodes *56*, 26, 91). Without the sequential matrix view, this fine of a granularity in pattern detection would likely be difficult to detect.

Figure 6: ECOviz-Time matrix sequence of temporal summary structure: periodic full clique (pfc). The DMN resting-state network reflects temporal patterns across the displayed time steps.

6 DEMONSTRATION PLAN

Prior to the demo, the system only requires an edge list of the dynamic or static network in question. Although this demo is specialized for functional brain network data and the preprocessing complexities that neuroscientists face, the system accepts any dynamic graph with a subset of nodes labeled by some criterion. Time-evolving networks must be partitioned into t edge lists, such that the i^{th} edge list contains edges present at time interval $i \in [1, t]$. Following data storage and TimeCrunch processing, the contrasting and temporal data analysis views are immediately available.

Contrasting Data Analysis: Regardless of single or double set drop-down mode, users interact with data source by modifying values of the desired drop-down menus. This automatically updates all supporting charts and the structure visualizations with the user's selections. In the case of double set drop-down mode, the system only updates the column referred to by the selected drop-down menu. To interact with the structure visualizations, users click on the button corresponding to the desired time step in the TimeCrunch summary table for the desired structure row. This updates the visualization with the chosen structure-time step pair.

Temporal Data Analysis: Users may visualize the full temporal sequence of a structure through two ways: graph visualization or adjacency matrix. To toggle each format, the user can click the button under the 'Graphs' and 'Matrices' column in the TimeCrunch summary table for the desired structure row (Figure 4). This refreshes the current visualization sequence with that of the structure selected by the user. In the adjacency matrix view, users may reorder the matrix by clicking, dragging, and dropping rows/columns – allowing for sorting criteria flexibility.

In both types of analyses, the user may hover over nodes in the structure visualizations to view node IDs. Proportion and entropy of resting-state region labels are also shown in the table underneath. We invite our audience to explore the spatial and temporal dynamics of their data through the interface.

7 RELATED WORK

Our work is related to visual graph analytics, and visualization techniques for temporal graphs.

Several graph visualization frameworks, including Apolo [10], OPAvion [1], and NetRay [14] focus on anomaly detection at the node level, while others [7, 23] visualize the patterns in the adjacency matrices. Apolo [10] is a graph tool that supports attention routing. The user picks a few seed nodes and Apolo interactively expands their vicinities to enable sense-making. OPAvion [1] is an anomaly detection system for large graphs that mines graph features on Hadoop, spots anomalies offline by leveraging anomaly detection techniques, and interactively visualizes the anomalous nodes via Apolo. Shneiderman proposes simply scaled density plots to visualize scatter plots in [23], [7] presents random and density sampling techniques for datasets with several thousands of points, while NetRay [14] focuses on informative visualizations of the spy (distribution and correlation plots of web-scale graphs). Unlike these works, the system in this paper visualizes domain-specific summaries of time-evolving networks and supports pairwise comparison of the extracted summary structures.

Limiting node movement between temporal snapshots, or preserving the mental map, has long been believed to benefit dynamic graph visualization [6]. Early methods targeting this constraint include *supergraph* creation that encodes node layouts in all time steps [11], and simulated annealing that minimizes the cost function of inter-timeslice node movement [17]. Since our method aims to provide responsive user interaction through fast graph drawing, these solutions do not provide the necessary speed.

Despite consensus that temporal transitions should be interpretable, choice of presentation mode (i.e., animation vs. small multiples) is still under debate. Small multiples, a timeline-based display, result in faster response times among participants of dynamic graph analysis tasks [2, 12]. Qualitative responses also indicate that animation between frames leads to higher cognitive load when tracking multiple, simultaneous *community* transitions. However, accuracy in tasks that involved following *specific* nodes and edges improved in the animation case. Another experiment found that participants detected patterns across a wider window of time steps using small multiples as compared to animation [8]. Results show that the best approach depends on the user's task: *global* topological and temporal trends are easier to detect using small multiples; *local* ones are better suited for animation.

In giving users the ability to detect community dynamics, a major challenge lies in the display of both the node and community topology of dynamic graphs. Vehlow et. al [26] developed a method with node-link diagrams that overlay ribbons linking communities across adjacent time steps. This elegantly avoids the issue of edge overdraw by only displaying node-link diagrams at the junction of time steps. The animated radial layout proposed by Yee et. al [28] relies on polar projection of nodes in a radial layout to reduce low-level edge crossings, and animation between focal nodes to show high-level trends. These approaches offer principled ways to navigate the trade-off between showing detail and abstraction.

Apart from community and node-level dynamics, ECOviz must also show context from a known set of static structures. We choose radial and spine drawings [5] to reflect the respective forms of cliques and bipartite cores. To convey community dynamics within these structures, we also stratify nodes by domain-specific label, a user-defined semantic substrate [24]. These layout restrictions make previously discussed approaches of [26] and [28] unsuitable for ECOviz. Since our work uses small multiples for its more global community-level task, the edge crossings present in clique structures could benefit from a static layout such as hierarchical edge bundling [13].

8 CONCLUSION

In this paper, we leverage domain-specific insight in partially labeled data to produce interpretable summaries of dynamic graphs. Specifically, we propose a summary generation process that uses an *egocentric* view of labeled nodes to direct TimeCrunch towards exploring existing functional communities in the connectome. We also introduce a visualization-based system, ECOviz, to allow users to interact with the generated summaries and compare the results of different data sources. Via ECOviz-Pair we offer contrasting analysis between dynamic graphs' data source (e.g., resting vs. mindful rest state) and preprocessing methods for ease of *evaluation*, as well as more detailed exploration of a network's temporal patterns via ECOviz-Time. To better *express* structure connectivity over time, we maintain a static node layout according to the TimeCrunch encoded structure and node labels, producing a clearer visualization of dynamics within and between nodes of the same community.

Future steps include quantifying summary quality to automate the selection of preprocessing parameters for dynamic graph construction, assisting the user in detecting time interval granularity (instead of keeping intervals of equal length), and allowing the user to customize the summarization routine by allowing for introduction of new static graph vocabulary patterns (beyond cliques, bipartite cores, etc.). Though the task of exploring dynamic graphs for structures is largely unsupervised, the previous modifications would adapt to the user's data, creating a domain-specific graph summary that more effectively communicates findings to the user.

9 ACKNOWLEDGEMENTS

The authors thank Chandra Sripada, M.D., for sharing the fMRI data that inspired this system. This material is based upon work supported by the University of Michigan.

REFERENCES

[1] L. Akoglu, D. H. Chau, U. Kang, D. Koutra, and C. Faloutsos. OPAvion: Mining and visualization in large graphs. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 717–720. ACM, 2012.

[2] D. Archambault, H. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, 2011.

[3] D. Archambault and H. C. Purchase. The mental map and memorability in dynamic graphs. In *Pacific Visualization Symposium (PacificVis), 2012 IEEE*, pages 89–96. IEEE, 2012.

[4] A.-L. Barabási, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications*, 311(3):590–614, 2002.

[5] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph drawing: Algorithms for the visualization of graphs*. Prentice Hall PTR, 1998.

[6] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. The state of the art in visualizing dynamic graphs. *EuroVis STAR*, 2, 2014.

[7] E. Bertini and G. Santucci. By chance is not enough: Preserving relative density through nonuniform sampling. In *Proceedings of the Eighth International Conference on Information Visualisation*, pages 622–629. IEEE, 2004.

[8] I. Boyandin, E. Bertini, and D. Lalanne. A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples. In *Computer Graphics Forum*, volume 31, pages 1005–1014. Wiley Online Library, 2012.

[9] E. Bullmore and O. Sporns. Complex brain networks: Graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009.

[10] D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: Interactive large graph sensemaking by combining machine learning and visualization. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 739–742. ACM, 2011.

[11] S. Diehl, C. Görg, and A. Kerren. Preserving the mental map using foresighted layout. In *VisSym*, pages 175–184, 2001.

[12] M. Farrugia and A. Quigley. Effective temporal graph layout: A comparative study of animation versus static display methods. *Information Visualization*, 10(1):47–64, 2011.

[13] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[14] U. Kang, J. Y. Lee, D. Koutra, and C. Faloutsos. Net-Ray: Visualizing and mining billion-scale graphs. In *PAKDD (1)*, pages 348–361, 2014.

[15] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos. VoG: Summarizing and understanding large graphs. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 91–99. SIAM, 2014.

[16] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos. Summarizing and understanding large graphs. In *Statistical Analysis and Data Mining*. John Wiley & Sons, Inc., 2015.

[17] Y.-Y. Lee, C.-C. Lin, and H.-C. Yen. Mental map preserving graph drawing using simulated annealing. In *Proceedings of the 2006 Asia-Pacific Symposium*

on Information Visualisation, volume 60, pages 179–188. Australian Computer Society, Inc., 2006.

[18] A. C. Lewis, N. S. Jones, M. A. Porter, and C. M. Deane. The function of communities in protein interaction networks at multiple scales. *BMC Systems Biology*, 4(1):100, 2010.

[19] C.-T. Li and S.-D. Lin. Egocentric information abstraction for heterogeneous social networks. In *International Conference on Advances in Social Network Analysis and Mining*. IEEE, 2009.

[20] D. S. Margulies, J. Böttger, A. Watanabe, and K. J. Gorgolewski. Visualizing the human connectome. *NeuroImage*, 80:445–461, 2013.

[21] D. Meunier, R. Lambiotte, A. Fornito, K. D. Ersche, and E. T. Bullmore. Hierarchical modularity in human brain functional networks. *Frontiers in Neuroinformatics*, 3, 2009.

[22] N. Shah, D. Koutra, T. Zou, B. Gallagher, and C. Faloutsos. TimeCrunch: Interpretable dynamic graph summarization. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1055–1064, 2015.

[23] B. Shneiderman. Extreme visualization: Squeezing a billion records into a million pixels. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 3–12, 2008.

[24] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006.

[25] M. P. Van Den Heuvel and H. E. H. Pol. Exploring the brain network: A review on resting-state fMRI functional connectivity. *European Neuropsychopharmacology*, 20(8):519–534, 2010.

[26] C. Vehlow, F. Beck, P. Auwärter, and D. Weiskopf. Visualizing the evolution of communities in dynamic graphs. In *Computer Graphics Forum*, volume 34, pages 277–288. Wiley Online Library, 2015.

[27] H. E. Wang, C. G. Bénar, P. P. Quilichini, K. J. Friston, V. K. Jirsa, and C. Bernard. A systematic framework for functional connectivity measures. *Frontiers in Neuroscience*, 8:405, 2014.

[28] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *IEEE Symposium on Information Visualization*, pages 43–50. IEEE, 2001.

Clipped Projections for More Informative Visualizations [A Work-in-Progress Report]

Bo Kang, Junning Deng, Jefrey Lijffijt, Tijl De Bie
Department of Electronics and Information Systems, IDLab
Ghent University — Imec
firstname.lastname@ugent.be

ABSTRACT

Imagine data that contains a dense core with points scattered around away from the core. An example would be data with outliers. A figure with a full view of a 2-dimensional projection of data then typically shows the points in the core all close to each other. Due to the fact that the visualization medium as well as our eyes have finite resolution, it may then not be possible to discern the location of the points in the core. In that case it may be more interesting to show a zoomed-in visualization that allows one to explore the structure of the core, while providing only limited information about points that are not part of the core. A trade-off emerges between showing small and large-scale structure, parametrized by the size of a bounding box. The quantification of this trade-off using Information Theory, and the concurrent optimization of the size of a bounding box and finding informative linear projections are the topics of this paper.

CCS CONCEPTS

•Mathematics of computing → Exploratory data analysis; Dimensionality reduction; Information theory;

KEYWORDS

Exploratory Data Mining, Dimensionality Reduction, Information Theory, Subjective Interestingness

ACM Reference format:
Bo Kang, Junning Deng, Jefrey Lijffijt, Tijl De Bie. 2017. Clipped Projections for More Informative Visualizations
[A Work-in-Progress Report]. In *Proceedings of KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada, August 14th, 2017 (IDEA @ KDD'17),* 8 pages.
DOI:

1 INTRODUCTION

Assume a data analyst wishes to glean insights into a 2-dimensional data set by visualizing it. For real-valued data, the most straightforward technique for doing this would be by means of a scatter plot. However, when the data consists of a dense core, with additionally a number of points scattered some distance around it, the dense core tends to show up as a blob of partially occluding points; see Figure 1(a) for an example.

IDEA @ KDD'17, Halifax, Nova Scotia, Canada

DOI:

The result of showing the far-away points is thus that the points in the core become less discernible from each other (whether due to limitations in the plot resolution or in human perception). The amount of information provided about them is effectively reduced by having to zoom out, while the resolution remains constant.

Clipped scatter plots. In this paper we propose the notion of a *clipped scatter plot*. Informally speaking, it can be obtained by overlaying a bounding box on the scatter plot (red box in Figure 1a), and clipping all points outside of this bounding box to the nearest point on the boundary (as indicated by the blue dashed lines in Figure 1a). After doing this, one can zoom in to ensure the bounding box fills the plotting area. The clipped points are then shown with a different marker to distinguish them from points that appear near the boundary, but are within the bounding box area (Figure 1b).

What a user learns from a clipped scatter plot is this: For an unclipped point, the user knows its location up to the resolution (of the displayed plot or human perception, whichever is worse). For a point clipped along a certain dimension, the user only learns that it is further away from the origin than the size of the bounding box along that dimension. Informally, the user learns that such points are 'far away' (outside of the bounding box) in some direction, but precisely how far is not revealed.

By varying the size of the bounding box and corresponding zoom level, clipped scatter plots thus allow one to trade-off detail about the points with small norms, with detail about the points with large norms. In this paper, we discuss how to formalize and optimize this trade-off in a rigorous manner, relying on Information Theory. As an illustration, the bounding box used in Figure 1 is optimal with respect to this measure.

Clipped projections. Data is usually high-dimensional, and in order to visualize it in a clipped scatter plot, its dimensionality needs to be reduced to 2, for example by means of a projection. We will call a clipped scatter plot of a projection of the data a *clipped projection*. To most efficiently inform the data analyst about the data, one can then search for the most informative clipped projection. This amounts to a simultaneous optimization problem over all possible 2-dimensional projections[1] and all possible bounding box sizes (for both dimensions).

We note that clipped projections are distinct from projections of a data set after outlier removal, for several reasons. First, a point that is outlying along one dimension may not be so along another one, such that it may be clipped in one clipped projection but not in another. It may also be clipped along just one dimension in the clipped projection of the data. Second, determining which points are outliers and which are not is often a hard call (Figure 1 is misleading in this respect). Our approach does not require one to make

[1] In fact, our work trivially extends to d-dimensional projections for arbitrary d.

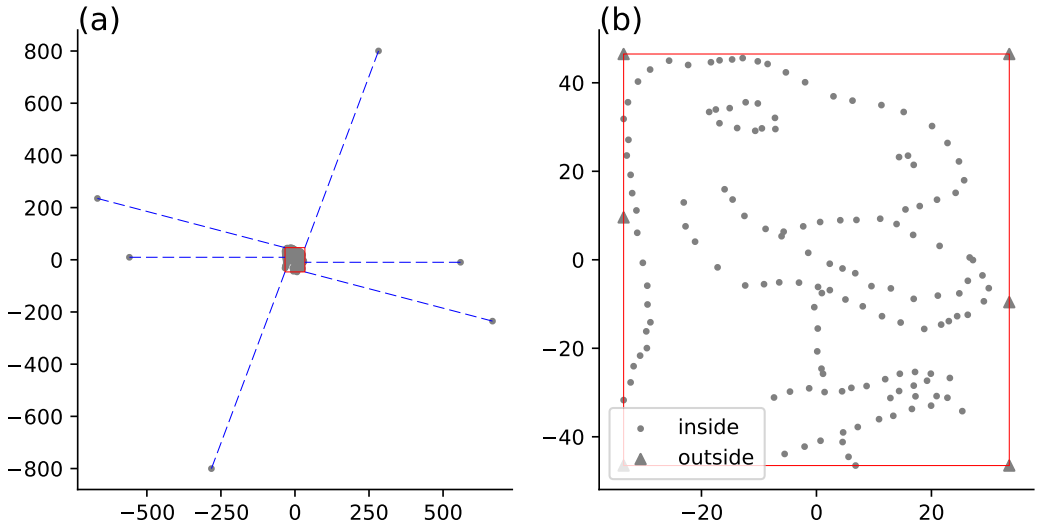


Figure 1: A scatter plot of a 2-dimensional data set (a) is not necessarily effective in revealing information about the data. Indeed, the points with large norm necessitate zooming out to such an extent that the detail in the core set of points is lost to the eye. By clipping points far away from the center to the boundaries of the red bounding box (blue dashed lines show where these points are clipped to) and zooming in as much as possible, a *clipped scatter plot* (b) is obtained. Here, we can discern the dinosaur hidden in the data. (The data consists of six artificial outliers plus Alberto Cairo's figure: http://www.thefunctionalart.com/2016/08/download-datasaurus-never-trust-summary.html.)

that call—our focus is merely on conveying as much information as possible about the entire data set through an intuitive visualization. Third, we never actually remove any points from the visualization: we merely convey less information about those outside the bounding box.

Contributions. In this work-in-progress paper, we report on early results on the following aspects.

- We introduce the notion of a *clipped scatter plot* and *clipped projection* (Section 2.2).
- We quantify the amount of information a clipped projection conveys about the data (Section 2.3). This quantification is parameterized both by the projection and the size of the bounding box.
- We introduce an algorithm that aims to optimize the information content, searching for the most informative clipped projection of a given data set (Section 3).
- We include some experiments that empirically analyse the ability of the algorithm to find the most informative clipped projection, the scalability of the algorithm, and the usefulness of the approach (Section 4).

2 CLIPPED PROJECTIONS AND THEIR INFORMATION CONTENT

In this section, we provide the formal definition of clipped projections and then discuss how to quantify their informativeness. First, we have to introduce some notation.

2.1 Notation

We use upper case bold face letters to denote matrices, lower case bold face letters for vectors, and normal lower case letters for scalars. We denote a d-dimensional real-valued dataset as $\hat{\mathbf{X}} \triangleq (\hat{\mathbf{x}}'_1, \hat{\mathbf{x}}'_2, \ldots, \hat{\mathbf{x}}'_n)' \in \mathbb{R}^{n \times d}$, and the corresponding random variable as $\mathbf{X}$. We will refer to $\mathbb{R}^{n \times d}$, the space the data is known to belong to, as the *data space*. Dimensionality reduction methods search weight vectors $\mathbf{w} \in \mathbb{R}^d$ of unit norm (i.e. $\mathbf{w}'\mathbf{w} = 1$) onto which the data is projected by computing $\hat{\mathbf{X}}\mathbf{w}$. If k vectors are sought, they will be stored as columns of a matrix $\mathbf{W} \in \mathbb{R}^{d \times k}$ where $\mathbf{W}'\mathbf{W} = \mathbf{I}$. We will denote the projections of a data set $\hat{\mathbf{X}}$ onto the column vectors of $\mathbf{W}$ as $\hat{\mathbf{\Pi}}_{\mathbf{W}} \in \mathbb{R}^{n \times k}$, or formally: $\hat{\mathbf{\Pi}}_{\mathbf{W}} \triangleq \hat{\mathbf{X}}\mathbf{W}$, and analogously for the random variable counterpart $\mathbf{\Pi}_{\mathbf{W}} \triangleq \mathbf{X}\mathbf{W}$. We will write $\mathbf{I}$ to denote the identity matrix of appropriate dimensions, and $\mathbf{1}_{n \times d}$ (or $\mathbf{1}$ for short if the dimensions are clear from the context) to denote a n-by-d matrix with all elements $\mathbf{1}_{ij} = 1$. We define $\mathbf{A}_{n \times m} \in [\mathbf{B}_{n \times m}, \mathbf{C}_{n \times m}]$ as $a_{i,j} \in [b_{i,j}, c_{i,j}]$ for every $c_{i,j} \geq b_{i,j}$, $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

2.2 Clipped projections

We define a *bounding box* to be a centered k dimensional hyper-rectangular region with range $(-\mathbf{c}, \mathbf{c})$ where $\mathbf{c} \in \mathbb{R}^k_+$. Given a bounding box defined by $\mathbf{c}$ and a projection $\mathbf{z}_i = \mathbf{W}'\mathbf{x}_i$ of a data point $\mathbf{x}_i$, its j-th coordinate either:

- falls outside of $(-\mathbf{c}_j, \mathbf{c}_j)$. Then position of the point is specified within range $\mathbf{z}_{ij} \in [\mathbf{c}_j, \infty)$ (or $\mathbf{z}_{ij} \in (-\infty, -\mathbf{c}_j]$).
- falls in $(-\mathbf{c}_j, \mathbf{c}_j)$. The position of the point is specified only up to a *pixel* of size $f \cdot 2\mathbf{c}_j$, i.e., $\mathbf{z}_{ij} \in [\mathbf{z}_{ij} - f\mathbf{c}_j, \mathbf{z}_{ij} + f\mathbf{c}_j]$, where f is the resolution parameter.

In order to concisely define a projection with respect to the bounding box, we need to introduce a few concepts. Firstly, we define a mapping function that corresponds to the clipping procedure: $t(\mathbf{z}_{ij}, \mathbf{c}_j) = \max(-\mathbf{c}_j, \min(\mathbf{c}_j, \mathbf{z}_{ij}))$. Now, we can express the

lower and upper boundaries of the location of $\mathbf{z}_{ij}$, as conveyed by the clipped scatter plot:

$$\mathbf{l}_j(\mathbf{c}_j, t(\mathbf{z}_{ij}, \mathbf{c}_j)) = \begin{cases} -\infty & : t(\mathbf{z}_{ij}, \mathbf{c}_j) = -\mathbf{c}_j, \\ \mathbf{z}_{ij} - f\mathbf{c}_j & : -\mathbf{c}_j < t(\mathbf{z}_{ij}, \mathbf{c}_j) < \mathbf{c}_j, \\ \mathbf{c}_j & : t(\mathbf{z}_{ij}, \mathbf{c}_j) = \mathbf{c}_j. \end{cases}$$

$$\mathbf{u}_j(\mathbf{c}_j, t(\mathbf{z}_{ij}, \mathbf{c}_j)) = \begin{cases} -\mathbf{c}_j & : t(\mathbf{z}_{ij}, \mathbf{c}_j) = -\mathbf{c}_j, \\ \mathbf{z}_{ij} + f\mathbf{c}_j & : -\mathbf{c}_j < t(\mathbf{z}_{ij}, \mathbf{c}_j) < \mathbf{c}_j, \\ +\infty & : t(\mathbf{z}_{ij}, \mathbf{c}_j) = \mathbf{c}_j. \end{cases}$$

Collectively, we define matrix $\mathbf{L}(\hat{\Pi}_\mathbf{W}, \mathbf{c})$ as a $n \times k$ matrix where each entry $\mathbf{L}(\hat{\Pi}_\mathbf{W}, \mathbf{c})_{ij}$ is the lower boundary of j-th coordinate of the projection of i-th data point. That is,

$$\mathbf{L}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)_{ij} = \mathbf{l}_j(c_j, t(\hat{\Pi}_\mathbf{W}^{(ij)}, \mathbf{c}_j)). \tag{1}$$

Similarly, $\mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)$ is the $n \times k$ matrix where each entry

$$\mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)_{ij} = \mathbf{u}_j(c_j, t(\hat{\Pi}_\mathbf{W}^{(ij)}, \mathbf{c}_j)). \tag{2}$$

Finally, we can define the syntax of a *clipped projection* as:

$$\hat{\mathbf{X}}\mathbf{W} \in \left[\mathbf{L}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right), \mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)\right]. \tag{3}$$

2.3 Information content of clipped projections

Prior belief model. Our aim is to quantify the information content of a clipped projection. Just like statistics are computed in comparison with a null model, in order to compute the information content of information, we need to specify a background model. Ideally, such a background model would reflect the actual prior knowledge of the user, such that the information content is an appropriate measure for the amount of information the visualization provides to the specific user [2, 3].

We adopt the same approach: we encode these prior beliefs in a probability density $p_\mathbf{X}$ over the data space $\mathbb{R}^{n\times d}$. The probability it assigns to any measurable subset of $\mathbb{R}^{n\times d}$ corresponds to the probability that the data $\hat{\mathbf{X}}$ would belong to that subset under the prior belief. The density function $p_\mathbf{X}$ can typically not be specified directly and instead we infer it from a given set of prior beliefs, as the Maximum Entropy distribution subject to those beliefs. As such, the notion of interestingness here is *subjective*, as the ranking of patterns depends on the belief state of the user.

In this paper, we assume the user has prior beliefs about the scale of a dataset.[2] The user might believe that the average scale of the data points, quantified by their squared norms, is some constant denoted as $\sigma^2 d$ and have no other knowledge. This can be encoded as a constraint on the second moment of the distribution $p_\mathbf{X}$:

$$\mathbb{E}_{\mathbf{X}\sim p_\mathbf{X}}\left[\mathrm{Tr}(\mathbf{X}'\mathbf{X})\right] = \sigma^2 \cdot nd. \tag{4}$$

The MaxEnt distribution subject to this constraint is well known and equal to a product distribution of multivariate Normal distributions $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, i.e.,

$$p_\mathbf{X}(\mathbf{X}) = \frac{1}{\sqrt{(2\pi\sigma^2)^{nd}}} \exp(-\frac{1}{2\sigma^2}\mathrm{Tr}[\mathbf{X}'\mathbf{X}]). \tag{5}$$

[2]There may be many other prior beliefs a user may have. Different prior belief types may result in background distributions of different types (See e.g., [4]). As the goal of this paper is to demonstrate the idea of the clipped projections, we leave the investigation of different prior beliefs as future work.

Given a projection matrix, the marginal distribution $p_{\Pi_\mathbf{W}}$ for projection $\mathbf{\Pi}_\mathbf{W} = \mathbf{X}\mathbf{W}$ then reads:

$$p_{\Pi_\mathbf{W}}(\mathbf{X}\mathbf{W}) = \frac{1}{\sqrt{(2\pi\sigma^2)^{nk}}} \exp(-\frac{1}{2\sigma^2}\mathrm{Tr}[\mathbf{W}'\mathbf{X}'\mathbf{X}\mathbf{W}]). \tag{6}$$

Probability of a clipped projection. If the projection $\mathbf{z}_i$ of a point $\mathbf{x}_i$ falls on the inside of the bounding box in j-th dimension, for small f its probability is well approximated by:

$$\begin{aligned}\mathbf{Pr}_{\Pi_\mathbf{W}}(\mathbf{z}_{ij} \in [\hat{\mathbf{z}}_{ij} - f\mathbf{c}_j, \hat{\mathbf{z}}_{ij} + f\mathbf{c}_j]) &= \int_{\hat{\mathbf{z}}_{ij}-f\mathbf{c}_j}^{\hat{\mathbf{z}}_{ij}+f\mathbf{c}_j} p_{\Pi_\mathbf{W}}(\mathbf{z}_{ij})d\mathbf{z}_{ij} \\ &\approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\hat{\mathbf{z}}_{ij}^2}{2\sigma^2}) \cdot 2f\mathbf{c}_j.\end{aligned} \tag{7}$$

A point $\mathbf{z}$ that falls outside the bounding box on the j-th dimension has probability

$$\begin{aligned}\mathbf{Pr}_{\Pi_\mathbf{W}}(\mathbf{z}_{ij} \in [\mathbf{c}_j, +\infty)) &= \mathbf{Pr}_{\Pi_\mathbf{W}}(\mathbf{z}_{ij} \in (-\infty, -\mathbf{c}_j]) \\ &= \int_{c_j}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{t^2}{2\sigma^2})dt.\end{aligned} \tag{8}$$

The first equality follows the symmetricity of the bounding box.

Now, the probability of a clipped projection can be written as,

$$\begin{aligned}&\mathbf{Pr}(\hat{\Pi}_\mathbf{W} \in \left[\mathbf{L}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right), \mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)\right]) \\ &= \int_{\Pi_\mathbf{W} \in \left[\mathbf{L}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right), \mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)\right]} p_{\Pi_\mathbf{W}}(\Pi_\mathbf{W})d\Pi_\mathbf{W} \\ &= \prod_{i=1,2,\ldots,n} \left\{ \int_{\mathbf{l}_1(c_1, t(\mathbf{x}_i\mathbf{W}_1, c_1))}^{\mathbf{u}_1(c_1, t(\mathbf{x}_i\mathbf{W}_1, c_1))} \cdots \int_{\mathbf{l}_k(c_k, t(\mathbf{x}_i\mathbf{W}_k, c_k))}^{\mathbf{u}_k(c_k, t(\mathbf{x}_i\mathbf{W}_k, c_k))} p_{\Pi_\mathbf{W}}(\mathbf{z}_i) \right. \\ &\left. d\mathbf{z}_1 d\mathbf{z}_2 \ldots d\mathbf{z}_k \right\}.\end{aligned} \tag{9}$$

The information content. Relying on the background distribution (Eq. 9), we can now quantify the information content (IC) of a clipped projection as the negative log probability of the projection under the distribution. Denote the index set of the projection of points on j-th dimension fall into $(-\mathbf{c}_j, \mathbf{c}_j)$ as $\mathbf{I}_j$, then the number of points falls outside of the bounding box is $n - |\mathbf{I}_j|$. Formally, we have the information content:

$$\begin{aligned}\mathrm{IC}(\mathbf{W}, \hat{\Pi}_\mathbf{W}, \mathbf{c}) &= -\log \Pr\left(\hat{\Pi}_\mathbf{W} \in \left[\mathbf{L}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right), \mathbf{U}\left(\hat{\Pi}_\mathbf{W}, \mathbf{c}\right)\right]\right) \\ &= -\log\left[\prod_{j=1}^{k}\left(\prod_{i\in\mathbf{I}_j} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\mathbf{z}_{ij}^2}{2\sigma^2}\right) \cdot 2f\mathbf{c}_j \right.\right. \\ &\left.\left. \cdot \prod_{i\notin\mathbf{I}_j} \int_{\mathbf{c}_j}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{t^2}{2\sigma^2})dt\right)\right]\end{aligned} \tag{10}$$

Now our goal of finding the most informative clipped projection can be formalized as an optimization problem:

$$\begin{aligned}\underset{\mathbf{W},\mathbf{c}}{\text{argmax}}\ & \mathrm{IC}(\mathbf{W}, \hat{\Pi}_\mathbf{W}, \mathbf{c}) \\ \text{s.t.}\ & \mathbf{W}'\mathbf{W} = \mathbf{I} \\ & \mathbf{c} > \mathbf{0}.\end{aligned} \tag{11}$$

In general, the solution of problem (11) corresponds to clipped projections that include as much information as possible complementary to the prior beliefs. Notice that given the prior belief stated above, if we ignore clipping—i.e., if the optimal clipped projection includes all points inside the bounding box—, the optimal solution is equivalent to PCA [3]. However, with clipping also the optimal $\mathbf{W}$ is typically different. In the next section, we analyze problem (11) and propose an approximation scheme to approach it.

3 FINDING THE MOST INFORMATIVE CLIPPED PROJECTION

Solving the optimization problem (11) requires to evaluate the objective function efficiently. However, the integral function (tail probability of normal distribution) in Equation (10) does not have a closed form of elementary functions hence can only be computed approximately. Moreover, note that for different projection matrix $\mathbf{W}$, the positions of points in the projection are different, hence the optimal $\mathbf{c}$ may change. This means the optimal bounding box (half size $\mathbf{c}$) and the number of points falling into the bounding box (i.e., $|\mathbf{I}|$) both depend on the projection matrix $\mathbf{W}$. Such dependency makes the optimization problem difficult to solve.

In this section, we first approximate the tail probability of a normal distribution by an upper bound and obtain an objective function consisting only elementary functions. We then propose an efficient optimization strategy that relies on automatic differentiation and gradient manifold optimization.

3.1 Bounding the tail probabilities

To obtain a closed form representation of (Eq. 10) with elementary functions we approximate the tail probability of a normal distribution as follows[3]:

$$\begin{aligned}\int_{\mathbf{c}_j}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-t^2/(2\sigma^2)} dt &\le \int_{\mathbf{c}_j}^{+\infty} \frac{t}{\mathbf{c}_j} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-t^2/(2\sigma^2)} dt \\ &= \frac{\sigma e^{-\mathbf{c}_j^2/(2\sigma^2)}}{\mathbf{c}_j\sqrt{2\pi}}.\end{aligned} \tag{12}$$

The first inequality follows the fact that $\frac{t}{\mathbf{c}_j} \ge 1$.

Now the objective function in (Eq. 10) can be re-written as:

$$\begin{aligned}&\mathrm{IC}(\mathbf{W}, \hat{\mathbf{I}}_{\mathbf{W}}, \mathbf{c}) \\ &\approx -\log\left[\prod_{j=1}^{k}\left(\prod_{i\in\mathbf{I}_j} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\mathbf{z}_{ij}^2)/(2\sigma^2)} \cdot 2f\mathbf{c}_j \cdot \prod_{i\notin\mathbf{I}_j} \frac{\sigma}{\mathbf{c}_j} \frac{e^{-\mathbf{c}_j^2/(2\sigma^2)}}{\sqrt{2\pi}}\right)\right] \\ &= \sum_{j=1}^{k}\left[\sum_{i\in\mathbf{I}_j} \frac{\mathbf{z}_{ij}^2}{2\sigma^2} + (n-|\mathbf{I}_j|)\frac{\mathbf{c}_j^2}{2\sigma^2} + (n-2|\mathbf{I}_j|)\log(\mathbf{c}_j)\right. \\ &\quad \left. |\mathbf{I}_j|\frac{1}{2}\log(2\pi\sigma^2) - |\mathbf{I}_j|\log(2f) + (n-|\mathbf{I}_j|)\log(\frac{\sqrt{2\pi}}{\sigma})\right].\end{aligned} \tag{13}$$

Notice the parameter $\mathbf{c}$ and $\mathbf{I}$ both depend on $\mathbf{W}$. That is, for every $\mathbf{W}$ we need to search for an $\mathbf{c}$ that maximizes the IC. Once $\mathbf{c}$ is computed, then the point sets within bounding box $\mathbf{I}$ with respect to each dimension are determined.

[3] A more detailed discussion can be found at https://mikespivey.wordpress.com/2011/10/21/normaltails/

3.2 Optimization strategy

The dependency of parameters $\mathbf{c}$ and $\mathbf{I}$ on $\mathbf{W}$ as well as the constraint $\mathbf{W}'\mathbf{W} = \mathbf{I}$ make objective (Eq. 13) difficult to be optimized simultaneously over all three parameters. Nevertheless, we propose a gradient method to perform the simultaneous optimization. Observe the orthonormality constraint posed on $\mathbf{W}$ leads to problem (13) being a *Stiefel* manifold optimization problem[4]. This can be addressed fairly efficiently with a standard toolbox. We use the *pyManopt* toolbox [5] to obtain an approximate solution.

In order to apply gradient based solver in *pyManopt*, we need to further compute the gradient of problem (13) with respect to variable $\mathbf{W}$. By encoding the objective function using *TensorFlow*, pyManopt can use TensorFlow's to calculate the gradient automatically.

The remaining question is: how to encode the step of searching optimal $\mathbf{c}$ (hence $\mathbf{I}$) into an objective function which then can be efficiently evaluated in a single step? The answer relies on the observation that for a specific $\mathbf{W}$ we only need to evaluate for each dimension $O(n)$ number of c (hence $O(kn)$ in total) to find the optimal $\mathbf{c}$. Without losing generality, we formally state the observation for j-th dimension as:

Proposition 1. *The optimal $\mathbf{c}_j^*$ that maximizes objective function (Eq. 13) coincides with the j-th absolute coordinate value of some projected point $\mathbf{z}_i$, namely, $\mathbf{c}_j^* = |\mathbf{z}_{ij}|$ for some $i = 1, \ldots, n$.*

Proof. To prove the proposition, it is equivalent to show that the objective function between the two neighboring coordinates (i.e., $\mathbf{c}_j \in [|\mathbf{z}_{m,j}|, |\mathbf{z}_{m+1,j}|]$) tends to be either monotonically increasing or convex.

The monotonic increase case can be easily identified by computing the first derivative of $\mathrm{IC}(\mathbf{c})$ with respect to $\mathbf{c}_j$, which is

$$\begin{aligned}\frac{d}{d\mathbf{c}_j}\mathrm{IC}(\mathbf{c}) &= \frac{n-|\mathbf{I}_j|}{\sigma^2}\mathbf{c}_j + \frac{n-2|\mathbf{I}_j|}{\mathbf{c}_j} \\ &= \frac{(n-|\mathbf{I}_j|)\mathbf{c}_j^2 + (n-2|\mathbf{I}_j|)\sigma^2}{\mathbf{c}_j\sigma^2}.\end{aligned} \tag{14}$$

When $n - 2|\mathbf{I}_j| > 0$, since $(n - |\mathbf{I}_j|) > 0$, it is straightforward that $\frac{d}{d\mathbf{c}_j}\mathrm{IC}(\mathbf{c}) \ge 0$. This implies that $\mathrm{IC}(\mathbf{c})$ monotonically increases as $\mathbf{c}_j$ increases, and the local maximum occurs at the right boundary, i.e., $|\mathbf{z}_{m+1,j}|$.

The other case is $n - 2|\mathbf{I}_j| \le 0$. To show this gives a convex piecewise $\mathrm{IC}(\mathbf{c})$. Let us look at the second derivative of $\mathrm{IC}(\mathbf{c})$ w.r.t c

$$\begin{aligned}\frac{d^2}{d\mathbf{c}_j^2}\mathrm{IC}(\mathbf{c}) &= \frac{n-|\mathbf{I}_j|}{\sigma^2} + \frac{2|\mathbf{I}_j|-n}{\mathbf{c}_j^2} \\ &= \frac{(n-|\mathbf{I}_j|)\mathbf{c}_j^2 + (2|\mathbf{I}_j|-n)\sigma^2}{\mathbf{c}_j^2\sigma^2}\end{aligned} \tag{15}$$

Since $2|\mathbf{I}_j| - n \ge 0$, we can easily notice $\frac{d^2}{d\mathbf{c}_j^2}\mathrm{IC}(\mathbf{c})$ is always positive, which essentially implies convexity. This means the largest function value is obtained on the boundary of interval $[|\mathbf{z}_{m,j}|, |\mathbf{z}_{m+1,j}|]$.

Thus, in call cases, the local maximum of $\mathrm{IC}(\mathbf{c})$ lies either at the left boundary or the right one. This observation allows us to search for an optimal c in the set $\{|\mathbf{z}_1|, |\mathbf{z}_2|, ..|\mathbf{z}_n|\}$. □

[4] A Stiefel manifold $\mathbf{V}_k(\mathbb{R}^n)$ is the set of ordered k-tuples of othornomal vetors in $\mathbb{R}^n$.

The proposition allows us to restrict the search space from $\mathbb{R}$ to n points. A naive strategy of searching optimal $\mathbf{c}$ would be to enumerate all points (set $\mathbf{c}_j = |\mathbf{z}_{ij}|$ for $i = 1, \ldots, n$) and find $|\mathbf{z}_{ij}|$ that gives the best objective value. For each $|\mathbf{z}_{ij}|$, evaluating the point set that fall in a bounding box (i.e., $\mathbf{I}_j$) requires time $O(n)$. Hence, the naive search strategy has complexity $O(n^2)$.

By a more careful thinking, the search can be improved to $O(n \log(n))$. The idea is as follows:

- sort $|\mathbf{z}_{ij}|$, $i = 1, \ldots, n$ in ascending order, $O(n \log(n))$. In TensorFlow, sorting can be encoded as a node in computational graph using function **tf.nn.top_k**[5].
- search the $\mathbf{z}_{ij}$ using the new order, $O(n)$. For each $\mathbf{c}_j = |\mathbf{z}_{ij}|$, we compare the objective values obtained by either containing $\mathbf{z}_i$ in the bounding box (i.e., $\mathbf{z}_i \in \mathbf{I}_j$) or without (i.e., $\mathbf{z}_i \notin \mathbf{I}_j$), and keep the larger value of two, $O(1)$. As the points are sorted, $|\mathbf{I}_j|$ is simply the number of the evaluated points, $O(1)$.
- to efficiently evaluate the first term (summation) in objective function (Eq.13), we accumulate the sum along the search of new $|\mathbf{z}_{ij}|$. This step costs $O(n)$. In TensorFlow, this can be encoded using function **tf.cumsum**[6].

Based on the above discussion, we can now evaluate objective function (13) by summing over the objective value over k dimensions. For each dimension j, we evaluate the summand (in rectangular brackets of Equation. 13) on a vector of $\mathbf{c}_j$s ($\{|\mathbf{z}_{1j}|, |\mathbf{z}_{2j}|, ..|\mathbf{z}_{nj}|\}$) and find the $\mathbf{c}_j^*$ that maximizes the summand. Thus, the evaluation of the objective function costs $O(n \log(n) + kn)$.

4 EMPIRICAL EVALUATION

In this section, we present two case studies which demonstrate how clipped projections may help users to explore data. Note that the purpose of our experiments is not to investigate superiority of clipped projections over existing methods for dimensionality reduction. Instead, we aim to investigate whether and to which extent the clipped projections usefully depend on the prior beliefs, in highlighting information that is complementary to them. Because the optimal solution for the specific prior belief assumed above without clipping is equivalent to PCA [3], we indeed compare the results from the method with projections corresponding to the principal components of the data.

4.1 UCI image segmentation dataset

The UCI Image segmentation dataset[7] consists of 210 data points. Each data point corresponds to an small 3×3 region (9 pixels) and was drawn randomly from a database of 7 outdoor images. The images were hand-segmented to create a classification for every pixel. The data points are described by 19 image features (e.g., centroid-row-position, hue-mean, intensity-mean). Each data points is classified as one of the following 7 classes: brickface, sky, foliage, cement, window, path, grass. As preprocessing, we centered the data.

We computed a visualization using $f = 0.01$, meaning that we expect to be able to effectively discern 100 points along one axis. We set the variance of the background distribution is to be the average variance of the data, namely $\sigma^2 = \mathrm{Tr}(\mathbf{X}'\mathbf{X})/nd$. To find an informative projection, we tried 100 random starts and used the best scoring result.

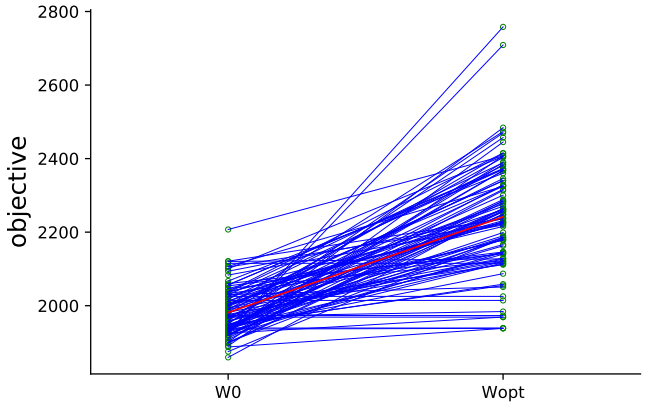


Figure 2: Objective values obtained from 100 random starts on the UCI Segmentation data. Blue lines connect the objective value (green circle) obtained at the initial step ($\mathbf{W}_0$) and the final step ($\mathbf{W}_{\mathbf{opt}}$) of each random start. The red line shows the average initial and final score.

We found that each random start took 0.21 seconds on average. In order to understand something about the difficulty of the optimization problem, and to find whether the gradient descent manages to find a good result, we plotted the objective scores of the initial random w with optimized c and the corresponding final objective scores in Figure 2. The red line shows how the objective value improves on average. We find that the quality indeed varies.

The best clipped scatter plot is shown in Figure. 3c. As a comparison, we consider the scatter plot of the projection against first (x-axis) and second (y-axis) principal component of the full data (3a). The principal components are dominated by a single point that has statistics much unlike the rest of the data. In contrast, our method indeed presents a quite different view. The projection is somewhat different (Figure. 3b) and the information content is greatly increased by clipping several points (3d). We can see the clipped scatter plot shows variation in the center of the data. It also gives the information about direction of the clipped points (points corresponding to the triangular markers on the edges).

Note that in Figure. 3c, the bounding box does not tightly fit the scattered points on the right side. This is due to the constraint that the bounding box is centered, i.e., the distances from origin to the bounding box boundaries are the same in both directions of each dimension. We plan to remove this assumption in the future and have a more flexible bounding box with the location of its center being optimized together with the box size.

4.2 UCI shuttle dataset

The UCI Shuttle dataset[8] consists of 14500 data points and 9 integer features. Each data point belongs to one of the 7 classes: 'Rad Flow', 'Fpv Close', 'Fpv Open', 'High', 'Bypass', 'Bpv Close', 'Bpv Open'. Similar to the case described in the previous section, we set

[5] see TensorFlow API https://www.tensorflow.org/api_docs/python/tf/nn/top_k
[6] see TensorFlow API https://www.tensorflow.org/api_docs/python/tf/cumsum
[7] http://archive.ics.uci.edu/ml/datasets/image+segmentation, 'segmentation.data'
[8] https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle), 'shuttle.tst'

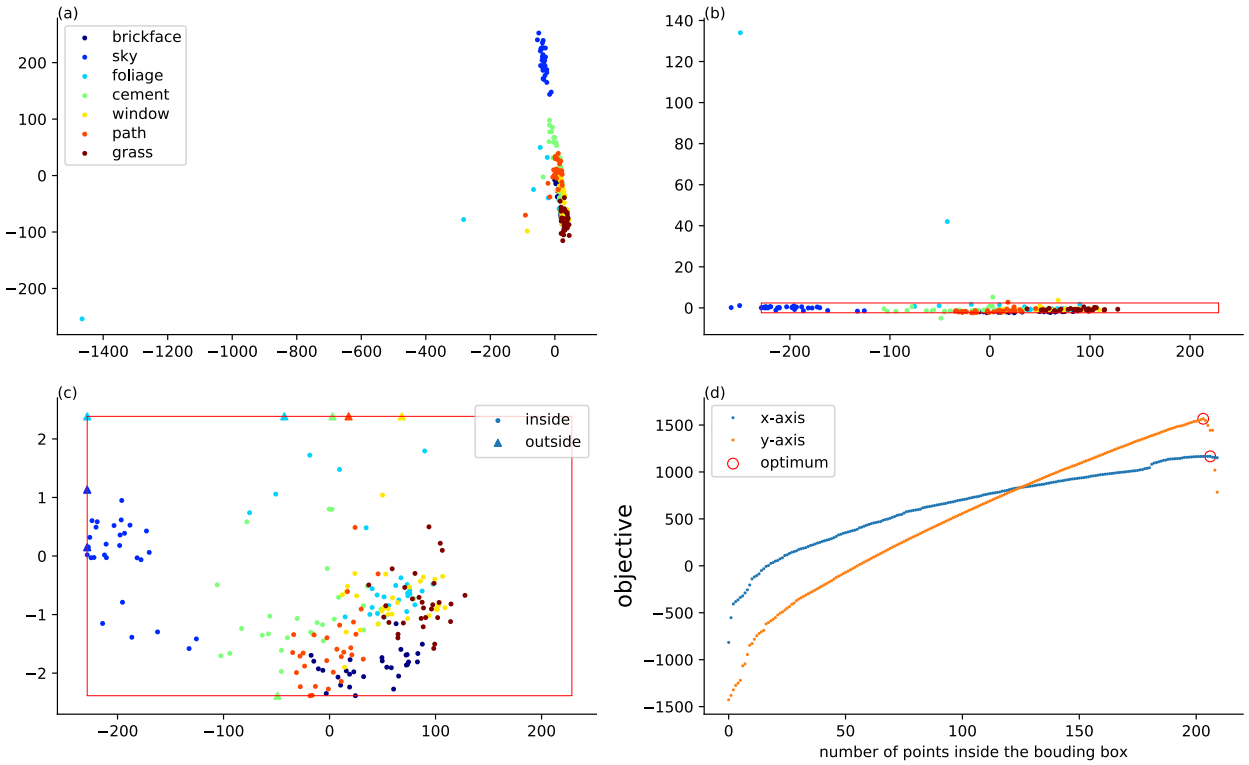


Figure 3: Results on the UCI Segmentation data. (a) The full data projected onto PCA first and second components. (b) The full data projected after optimization but without clipping. The seven class labels are indicated with colors and the red box gives the bounding box. (c) The end-result of our method. Round dots here correspond to data points that are fully inside the box, while triangles correspond to points (partially) outside the bounding box. (d) The objective values obtained for different bounding box sizes c_x, c_y along the directions of W. The PCA projection scores 2050.15 for the two dimensions combined.

$f = 0.01$. The variance of the background distribution is also set to be the average variance of the data, namely $\sigma^2 = \mathrm{Tr}(\mathbf{X}'\mathbf{X})/nd$. The dataset is centered. Same as the previous experiment, we tried 100 random starts. Each random start take on average 0.51 seconds. This may illustrate that the optimization strategy scales well, since the size of shuttle data is ten times larger as the segment data, yet the average time per random start only doubled.

Figure 4 contains the same plots as Figure 3 but for the Shuttle data. The PCA result (4a) shows the variance structure dominated by a set of points with large norms. The bounding box found after optimization is so small in this case, that it is not even visible on the non-clipped scatter plot (4b). The clipped scatter plot (4c) shows that the majority of the data points form a layered structure on a small scale. The layered structure may partially be due to the discreteness of the data. 4d shows the objective values for various $\mathbf{c}$ along the final two projection direction.

For both dimensions, the objective increased very fast initially. This is because the projection most of the points lies close to the origin. When the size of bounding box increase the objective function also increase linearly. The objective function starts to drop rapidly at the end, when the points with large magnitude are included.

4.3 Runtime

Table 1 summarizes the runtime of our method in all experiments presented in this paper. In all cases, we used the solver offered by pyManopt to perform gradient descent (with automatic differentiation provided by TensorFlow) over the Stiefel manifold. We tried ten random starts in all three cases and picked the projection that gives the best objective. Note in the first row of the table, our optimization strategy scales gracefully when the data size increases from Synthetic dataset (148×2) to UCI Segment (210×9) and then UCI Shuttle (14500×9). Although evaluating the objective function involves optimizing the bounding box size, the costs (second row) remain almost constant for increasing data size; the constant overheads from pyManopt and TensorFlow dominate this step.

5 CONCLUSION

A scatter plot of a projection is arguably the most basic way of conveying complete information about a high-dimensional numerical data set. If suitable projections can be found, it promises to empower human data analysts by allowing them to use the remarkable pattern recognition capabilities of human perception: clusters, (local) correlations, outliers, etc. are readily noticed without effort.

Yet, often the scale of a scatter plot is too strongly influenced by a possibly small set of points that are farther than usual from

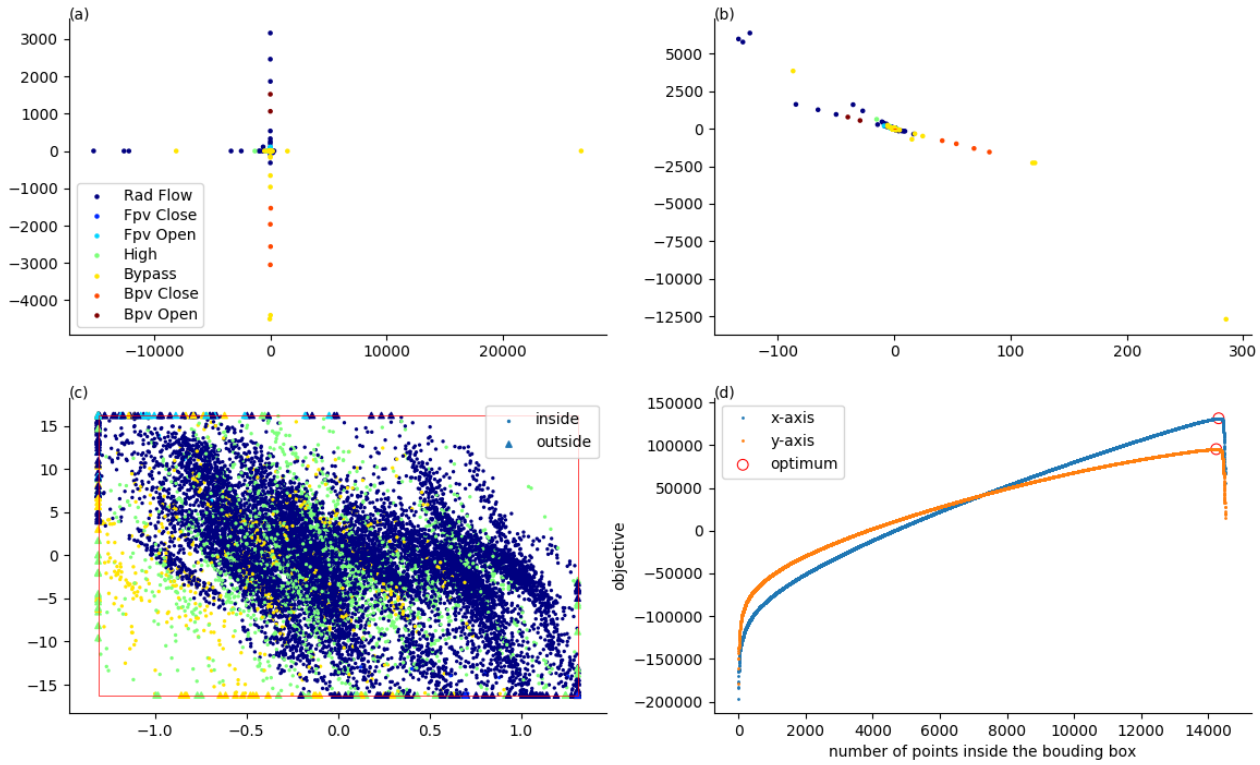


Figure 4: Equivalent to Figure 3, for the UCI Shuttle data. (a) The full data projected onto PCA first and second components. (b) The full data projected after optimization without clipping. The seven class labels are indicated with colors and the red box gives the bounding box. (c) The end-result of our method. Round dots here correspond to data points that are fully inside the bounding box, while triangles correspond to points (partially) outside the bounding box. (d) The objective values obtained for different bounding box sizes c_x, c_y along the directions of W. The PCA projection scores only 69366.9 for the two dimensions combined, against 226630.0 for our method.

	synthetic	UCI Segment	UCI Shuttle
Optimization	0.9717	1.5268	8.2443
Evaluation	0.1307	0.1316	0.1344

Table 1: Runtime (in seconds) of our method for all experiments (§4.3). Each measurement of optimization (first row) is an average over ten runs, where each run consists of ten random start of the gradient based solver from pyManopt and TensorFlow. We also measured the cost of evaluating the objective function (second row). Each measurement is an average over ten evaluations. We used a machine with Intel Quad Core 2.7 GHz CPU and 16 GB RAM.

the centre of the data. As a result, the amount of detail that can be shown for the points closer to the centre is reduced, which is problematic if such points are numerous and the variation among them important. As a result, the overall information conveyed by a scatter plot can be disappointing.

To address this issue, we proposed the notion of a *clipped projection*, which clips the farthest points in a data projection to a bounding box, and subsequently zooms in to let the bounding box fill the plotting area. We then quantified the amount of information a clipped projection conveys about the data, and proposed an algorithm for maximizing this information content over all possible projections and bounding box sizes.

The information content of a clipped projection is formalized by relying on the FORSIED[9] framework [1]. This framework aims to formalize the information content of data mining patterns in a subjective manner: considering the data analyst's prior beliefs about the data. In the current work-in-progress report, we assumed that the user has no prior idea about the data other than its overall scale (which can be easily computed). Our ongoing work, which also builds and improves on previous applications of the FORSIED framework to dimensionality reduction [3], focuses on deploying these principles for other prior beliefs as well, as well as to visualizations of high-dimensional data other than clipped projections.

Acknowledgements. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement no. 615517, from the FWO (project no. G091017N, G0F9816N), and from the European Union's Horizon 2020 research and innovation programme and the FWO under the Marie Sklodowska-Curie Grant Agreement no. 665501.

[9] 'Formalizing Subjective Interestingness in Exploratory Data mining'.

REFERENCES

[1] T. De Bie. 2011. An information-theoretic framework for data mining. In *Proc. of KDD*. 564–572.

[2] T. De Bie. 2013. Subjective interestingness in exploratory data mining. In *Proc. of IDA*. 19–31.

[3] Tijl De Bie, Jefrey Lijffijt, Raúl Santos-Rodríguez, and Bo Kang. 2016. Informative Data Projections: A Framework and Two Examples. In *Proc. of ESANN*. 635–640.

[4] Bo Kang, Jefrey Lijffijt, Raúl Santos-Rodríguez, and Tijl De Bie. 2016. Subjectively Interesting Component Analysis: Data Projections That Contrast with Prior Expectations. In *Proc. of KDD*. 1615–1624.

[5] James Townsend, Niklas Koep, and Sebastian Weichwald. 2016. Pymanopt: A Python Toolbox for Optimization on Manifolds using Automatic Differentiation. *Journal of Machine Learning Research* 17, 137 (2016), 1–5. http://jmlr.org/papers/v17/16-177.html

Towards an Interactive Learn-to-Rank System for Economic Competitiveness Understanding

Caitlin Kuhlman
Worcester Polytechnic Institute
cakuhlman@wpi.edu

Elke Rundensteiner
Worcester Polytechnic Institute
rundenst@wpi.edu

ABSTRACT

Ranking models are useful tools often employed to aid in decision making. In fields such as economics, the development of indicators to rank economies or regions are typically dictated by *expert opinion*. With the increased availability of high fidelity open data, better tools for developing and understanding rankings can provide valuable insight into social and economic questions. This paper presents a preliminary foray into the development of such tools. We introduce a vision for leveraging state-of-the-art algorithms from the Information Retrieval field to design interactive learn-to-rank tools. Incorporated into data analytics systems via plug-and-play components, such tools hold the potential to better evaluate the comparative merits of different regions and to interactively assess the impact of different features on the final rankings. The *MyRanker* paradigm is applied in the context of MATTERS, a public system for evaluating the economic competitiveness of US states. A preliminary analysis and discussion of the system highlight its promise for ranking analysis.

KEYWORDS

Learning algorithms, Interactive Visualizations, Rankings, Machine Learning

ACM Reference format:
Caitlin Kuhlman and Elke Rundensteiner. 2017. Towards an Interactive Learn-to-Rank System for Economic Competitiveness Understanding. In *Proceedings of KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada, August 14th, 2017*, 9 pages.
DOI:

1 INTRODUCTION

Rankings are a fundamental tool used in many applications to help people understand the relative merit of objects or choices. They are commonly employed to simplify decision making when the number of factors impacting choice is large. In economics, a variety of rankings or indicators are used to gauge the relative performance of countries and regions [5, 17, 18]. These indicators are formed by combining various statistics such as tax codes, GDP, population, and so on with the aim to measure a relative ranking among objects according to economic principles. Another example of ranking for decision making is the use of college rankings [8, 20], which are designed to help potential students choose which school to attend by accounting for factors such as student demographics, location, and salary outcomes for graduates. In fact aggregated statistics are used widely for everything from valuing the stock market via indices such as the S&P 500 and NASDAQ to ranking restaurants and other businesses via social media sites such as Yelp.

Machine learning techniques to automatically rank objects have been extensively researched in the context of Information Retrieval [13]. Indeed, efficient web search is facilitated by learning-to-rank algorithms which evaluate the relevance of online documents to a given user query. This type of user-driven ranking has been on the front lines of the information revolution, broadly providing high fidelity results over data resources otherwise too large to browse.

With advances in data science and the proliferation of high quality open data come opportunities to leverage these machine learning methods to answer social and economic questions. Interest in data science and knowledge discovery techniques as applied to social sciences and economics is growing [11, 18]. The development of robust tools is needed to enable non-expert users to better understand the explanatory power of machine learning models for socioeconomic outcomes. Such tools can ultimately greatly increase the utility of open data for social good.

In this work we explore the use of machine learning to aid in the construction and understanding of ranking models. Powered by learning-to-rank machine learning [13], we introduce a new paradigm for interactive exploration to aid in the understanding of existing rankings as well as facilitate the automatic construction of user-driven rankings. Components are incorporated into a plug-and-play framework. We demonstrate how the framework can be applied to the problem of measuring and analyzing economic competitiveness through the development of a prototype data analytics system. Interactive learn-to-rank tools extend the capabilities of MATTERS [1], an online platform designed to evaluate the relative competitive advantages of US states.

1.1 Motivation: Rankings and Their Pitfalls

One of the most common applications of data to social science and policy is in the field of economics, where measurements such as rankings serve a critical role in evaluating the economic health of regions and shaping policy. For example, the so-called "Misery Index" [2] is a historical measure of economic performance developed by economist Arthur Okun. This simple index is computed as the sum of the seasonally adjusted unemployment rate to the annual inflation rate. Many such indicators have been designed to measure past economic performance or predict future economic conditions.

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada

DOI:

[1] http://matters.mhtc.org/
[2] http://www.miseryindex.us/

Regional rankings are often employed to quantify relative economic competitiveness. For instance, the World Economic Forum publishes an annual Global Competitiveness Report [17] containing a ranking of countries around the world computed from numerous metrics and survey data. In the US, other indices compare the competitiveness of different states, such as the Milken Institute's State Technology and Science Index [5] or CNBC's Top States for Business [3] ranking.

The design of such indices depends heavily on "expert opinion", which drives the selection of the metrics and weightings used in their construction. While extremely valuable, the dissemination of expert knowledge through such rankings is somewhat limited. One, it may be that the actual formula used to compute the ranking is not made public. Or, though published, the model is unlikely to be accessed by the average consumer. By only considering the final ordering given by the ranking, consumers are limited in their ability to gain further insight into the implications of the ranking. Another consideration in the design of such ranking models is confirmation bias. Even with the best of intentions, evaluations originating in expert opinion may succumb to this common phenomena [15].

Furthermore, latent factors incidentally measured by a ranking are not always made explicit. The data used in the design of the ranking may be serving as a proxy for undesirable metrics. For instance, when considering the economic competitiveness of different regions, evaluating the quality of the available talent pool based on race or gender would be highly undesirable. For this reason, a ranking model should be evaluated in the context of all available data, and compared with rankings based on demographic information to ascertain whether it is reflecting inherent bias in the underlying datasets.

At times, the designers of ranking systems may struggle to avoid these pitfalls, and to determine the most useful and fair data upon which to base rankings. The incorporation of learning-to-rank algorithms into highly usable interfaces is crucial to help users better understand existing rankings, and to aid them in the creation of ranking models which reflect their intuition and value system. In this work we thus propose new interactive tools to explore the construction of such indices.

1.2 MyRanker: Our Proposed Interactive Learn-to-Rank Paradigm

In this work, we investigate the question: *Are there data-driven approaches to the construction of rankings that would allow designers to gain more insight into the concepts they attempt to model?* To address this question, we propose the design of an interactive exploratory paradigm for both the construction of rankings, as well as better understanding of existing ranking models, here referred to as MyRanker. Our easy-to-use rank construction tools in MyRanker allow stakeholders to drive the ranking process. They can either manually specify criteria for their preferred rankings via a visual interface, or leverage rank learning algorithms from the machine learning field [13] to automatically derive rankings based on partial information based on their domain knowledge or priorities. Intuitive visual interfaces allow users to design rankings which reflect their intuition or meet their objective simply by indicating partial preferences over objects in the dataset.

Further, we study the question: *Can we more closely couple the underlying data and the resulting ranking, so that users better understand the impact of data on the relative ranking of objects?* For this, visual rank exploration tools in MyRanker are offered to interactively explore, compare, and analyze these newly constructed rankings. Multiple visual displays are closely interlinked – visualizing both the relative ordering and the detailed description of the objects being ranked. Direct display of any adjustment of criteria on the resulting ranking can bring insights into the relative impact of particular data on a particular ranking – with promise of a high value return.

This seamless integration between rank construction tools and rank exploration tools into one single easy-to-use analytics system empowers users to gain insights into the differences between ranking models. Incorporated into a plug-and-play framework, our MyRanker paradigm enables users to better understand the interplay of the data and their rankings through integrated data visualizations and interactions.

1.3 Contributions

The contributions of this work including the following:

(1) This paper introduces the notion of an interactive paradigm for learn-to-rank tools supporting the process of exploring and understanding rankings. This considers both the ease of the specification of rankings via visual support as well as the display of ranking results.
(2) We present a plug-and-play framework, called MyRanker, to support stakeholders to interact with and thus understand ranking models. MyRanker integrates rank learning components into a data analytics system.
(3) We describe a demonstration of our interactive learn-to-rank tool, MyRanker, applied for economic competitiveness evaluation as part of the Massachusetts Technology, Talent, and Economic Reporting System (MATTERS). Integrated components for user interaction and rank learning are seamlessly incorporated into this analytics platform.

2 LEARNING-TO-RANK BACKGROUND

In the Information Retrieval (IR) field, a number of methodologies for learning-to-rank have been developed. In [13], Lui et al. categorize 3 different supervised learning approaches: pointwise, pairwise, and listwise. The pointwise approach reduces to a regression analysis, where a model is trained on instances which each have either a numeric or ordinal score assigned to them. A ranking of unseen data is determined based on the regression values given by the model. Listwise approaches learn based on entire sets of ordered objects. That is, the training set consists of multiple ordered lists with corresponding rankings, and the model assigns an ordering over an entire previously unseen list [4, 19].

In the pairwise approach, training data is composed of pairs of objects. Labels are assigned to each pair which indicate a preference between them. For instance, given a pair (a, b) it is assigned label 1 is a is preferred to b, label -1 if b is preferred to a, and 0 if the two instances are equally preferred. Datasets which consist of individual

[3] http://www.cnbc.com/americas-top-states-for-business/

objects and labels can be transformed to this pairwise format by forming every possible ordered pair and comparing their labels. In [7], the pairwise approach is shown to reduce the learning-to-rank problem to a binary classification problem. Given this, any classification model can be employed to learn a ranking. Proposed classification models, to just name a few, include SVM [7], neural networks [3], and regularized least-squares [16].

The techniques developed in IR are intended to present only the most relevant results ranked based on user queries to a search engine. This task has a few distinguishing features from ranking in other contexts. For one, the amount of data in this problem is large. Thousands of documents may be returned for a single query, and prohibitively many features could be extracted from each. Models can be trained on a huge corpus of text. For data at this scale, certain methods may have an advantage over others. For instance, neural networks can leverage and in fact improve when applied to huge training sets [3]. For many other problems, such as social or economic evaluations, the data is likely to not be so large. While much public data are available, often for analytics tasks it is cleaned and preprocessed and only a small relevant set of data is used to determine the final outcome. In such cases models which require large amounts of training data will not perform well.

Another consideration is that in the search context, often the task is to find only the top results, not necessarily a complete ordering of all objects. For a thousand documents returned for a query, a user will likely only be interested in a handful of the most relevant results. Therefore a number of evaluation metrics have been developed which favor correct results at the top of a list and are forgiving of mis-ranked data toward the end of the list [9]. Such measures are not appropriate in other contexts, where the position of all objects in the ranking is of keen interest.

3 USER-DRIVEN RANKING: THE MYRANKER FRAMEWORK

Since the design of rankings is usually intended to capture some quality of the objects being ranked that cannot be directly measured, it is by its nature a difficult task for users to perform. Our MyRanker interactive learn-to-rank framework thus alleviates the need for the user to assign an explicit value as rank to each object in the dataset. Instead, by asking users to assign preferences between a select subset of objects (possibly those objects that they are personally familiar with), the system learns from their mental model of the problem. This empowers the users to construct rankings automatically using only partial information.

MyRanker solicits hints from the user in the form of a preference assignment between pairs of objects. This approach lets users tap into their expertise or intuition about the concept they are trying to capture in their ranking – yet without the need to manually construct an entire ordering. We employ the pairwise learning-to-rank approach to facilitate user-guided ranking analysis. Users assign priorities to pairs of objects via sample pairs, e.g., (object1 > object2). The system then learns a pairwise model which determines both a continuous value for each object and a resultant ranking.

The interactive learn-to-rank system is realized as a plug-and-play framework for ranking analytics, i,e., modules such as the learning algorithm as well as displays can be easily switched out.

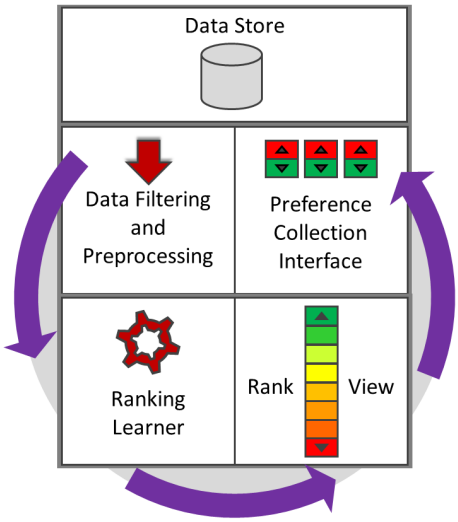


Figure 1: MyRanker Framework

We target applications where the data may be relatively small and is available from a data warehouse. Figure 1 sketches the system architecture of the MyRanker framework. It consists of a data repository, data filtering and preprocessing module, preference collection module, rank learning model, as well as a visual analytics interface. The visual analytics interface presents results, both rankings as well as their underlying data, to users through a number of interlinked displays. Upon visual inspection, users can manually update their preference and metric selections or individual weightings of metrics to refine the learned model in an iterative fashion.

4 MYRANKER APPLIED TO ECONOMIC COMPETITIVENESS ANALYSIS

The Massachusetts Technology, Talent, and Economic Reporting System (MATTERS) is an online public tool developed at Worcester Polytechnic Institute under the guidance of the Massachusetts High Technology Council[4] – in partnership with numerous stakeholders and domain experts. The goal of MATTERS [14] is to better understand and measure the economic competitiveness of US states using open data. To achieve this goal, MATTERS consolidates a rich collection of publicly available socioeconomic datasets. By making over 50 datasets available in one place for the first time, the system empowers decision makers from government officials to company executives to evaluate the economic conditions in their state in contrast to other states.

Developed with experts from high technology industry, research organizations, and higher education institutions, MATTERS provides descriptive analytics for the data in the system. In addition, automated web extraction tools and administrative easy-to-use data curation tools have been developed by WPI [6]. These data curation tools have now been made available to external partners, namely, teams of students at Brandeis University for continued data curation into the MATTERS warehouse twice a year. MATTERS

[4] http://mhtc.org

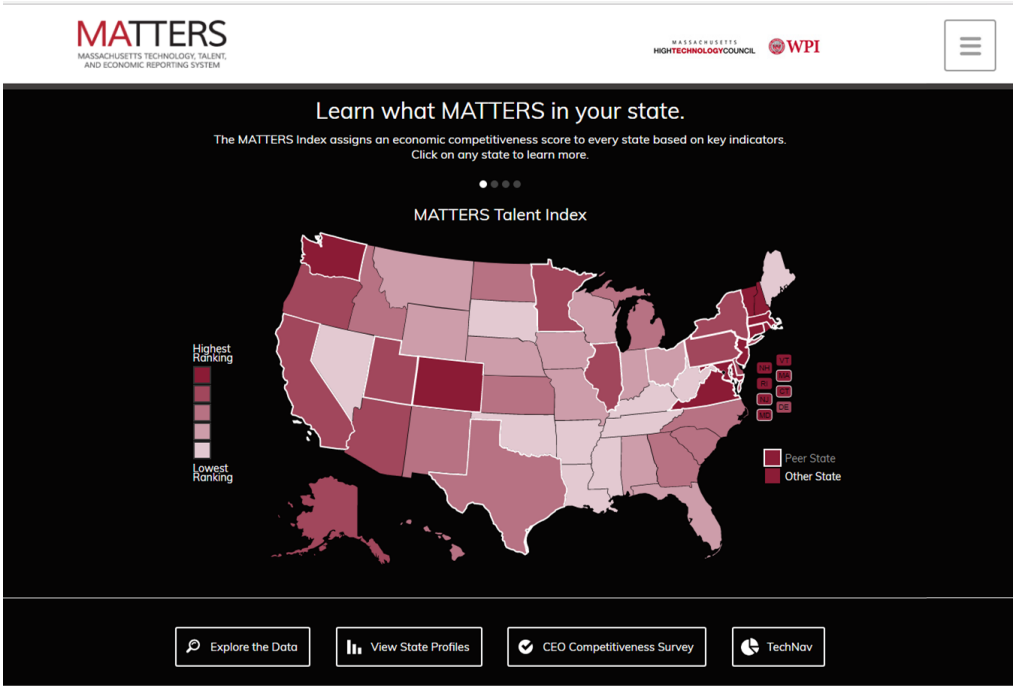


Figure 2: The Massachusetts Technology, Talent, and Economic Reporting System (http://matters.mhtc.org)

also provides a public-facing API to facilitate data reuse by other researchers. [5]

MATTERS is not designed to be prescriptive in scope, rather this open data repository includes a suite of tools which allow for user-driven data exploration [1]. Therefore, the MATTERS system provides the perfect test-bed for new ranking analysis features. The MyRanker framework applied in this context provides additional exploratory tools to aid in the understanding of existing rankings, as well as to facilitate the automatic construction of new user-driven ranking models for states. Seamless integration of MyRanker into MATTERS capitalizes on the customized visualizations in the MATTERS system, which are designed to provide insights specific to the spatio-temporal nature of the data. This allows for comparison of rankings to discover correlations, or observe changes in rankings of certain regional areas or over time.

4.1 Rank Specification Tools

Economic indices represent cumulative data over different related datasets. Beyond carefully crafted indices based on expert knowledge, the MATTERS rank specification tools now also empower users to combine data and perform complex analysis themselves in an interactive fashion. The data in the system is comprised of existing rankings (including 4 MATTERS Indices), as well as a collection of other (raw) data related to measuring economic competitiveness. The data is organized under 4 categories: Talent, Cost of Doing Business, Tax Climate and Quality of Life. Easy to use interfaces allow for the selection of individual metrics of interest to the user. Rank specification tools create new rankings based on these selections, via manual weightings of individual metrics (Figure 3) or pair preferences (Figure 4).

Custom rankings are stored as numeric formulas retrievable through user accounts. This way, the rankings are kept up-to-date as new data becomes available. The system regenerates data according to the user-defined rules each time the index is requested. For use in custom rankings, the following strategies are used to clean and standardize the data in order to produce meaningful results:

Missing Values. The MATTERS system contains data for multiple years, and the availability of each dataset may vary. A state ranking is computed for each year that at least one selected metric is available. If a value for some metric is missing for a given year, the closest previous value is used. If there is no previous value, the closest possible value is used.

Normalization. The datasets in the system vary greatly. Some, such as tax rates, are percentages which vary only by a few tenths of a point, while others are numbers in the millions representing state populations, or GDP. The data must be normalized so that large values do not dominate. Therefore the data in the system is standardized by setting the mean of each metric to 0 and the standard deviation to 1.

Inverted Trends. Typically when looking at trends, high values are considered better than low values. However, for some data the opposite is true, as in a low unemployment rate being preferred to a high rate. For the manual construction of metrics in the Rank Builder tool, negative coefficients are used for data with inverted trends.

[5] http://matters.mhtc.org/api

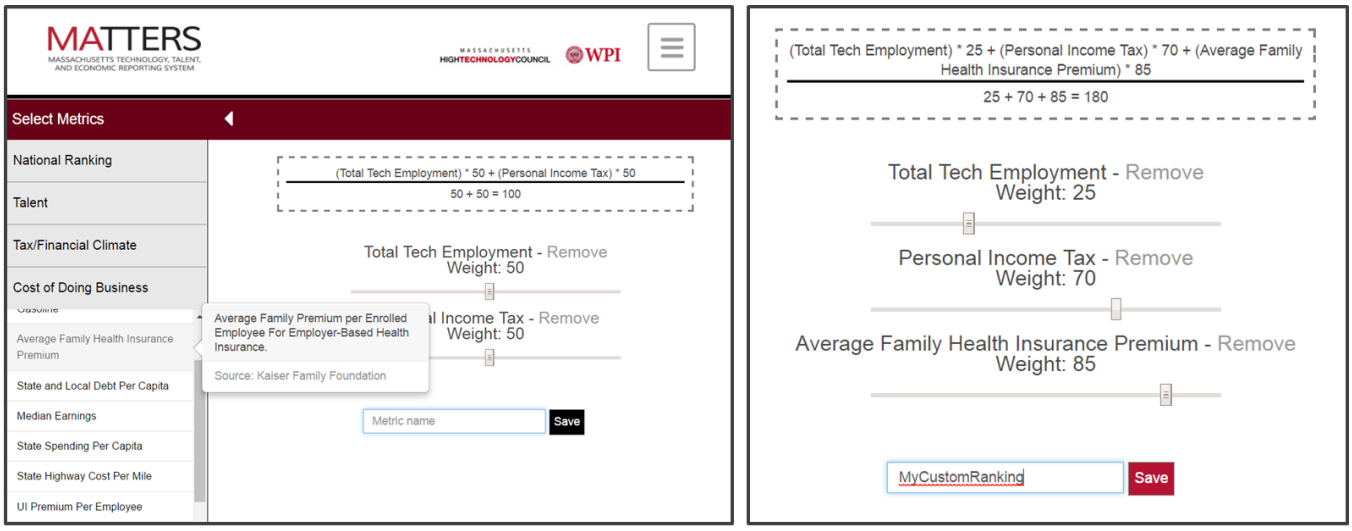


Figure 3: MATTERS manual Rank Builder tool.

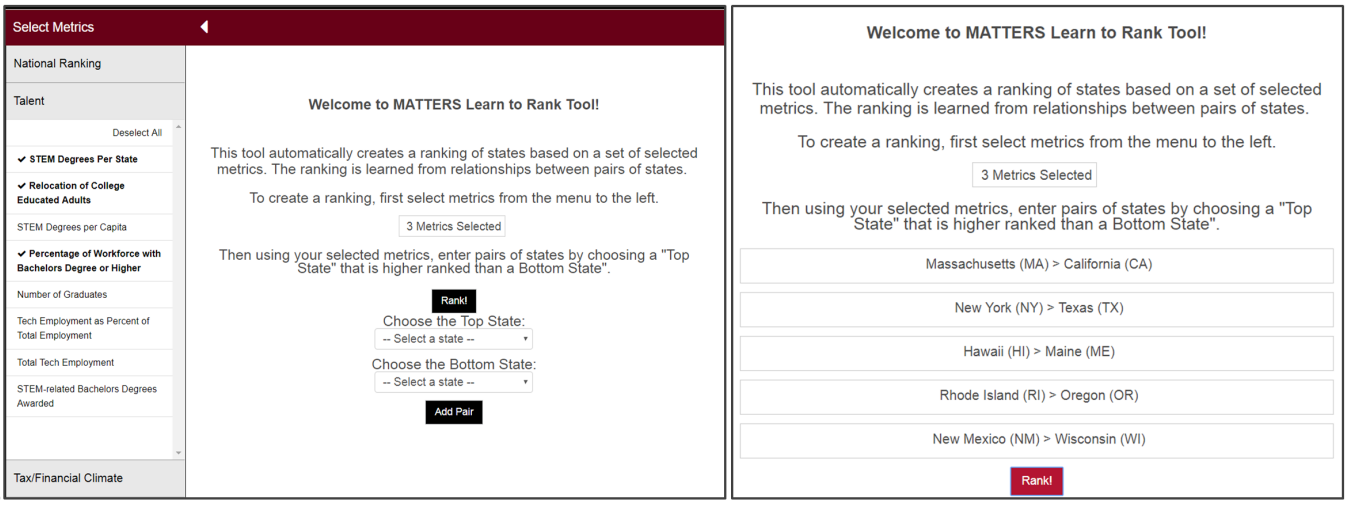


Figure 4: MATTERS pairwise Learn-to-Rank tool.

4.1.1 Rank Builder Tool. This novel interactive metric-creation tool was developed as an instrument for users to visually define their own indices. Users determine which datasets are of most interest and specify their relative importance to compose a compound index. Figure 3 shows the interface for the Rank Builder tool. Users can select metrics from the menu on the left, which lists datasets organized by category in collapsible menus. A description and source link are displayed when the user hovers over each dataset in the menu. Selected metrics are each displayed along with a slider tool. The screen on the right of Figure 3 shows how users can specify a weighting for each metric to indicate its impact on the desired state ranking. The formula for the resulting weighted average is shown at the top of the screen.

4.1.2 Learn-to-Rank Tool. This feature implements the pairwise rank learning component of the MyRanker framework. Figure 4 shows the interface to the MATTERS Learn-to-Rank tool. Metrics to be used in the automatic construction of a ranking are selected from the left-hand menu. Then, via the dropdown menus shown in the center screen, users can enter pairs of states. Preference is indicated by selecting a "Top State" and a "Bottom State". Once the user has entered a series of state pairs (as shown on the right of Figure 4), the automatic learn-to-rank engine is run using the "Rank!" button. A global ordering of all states is automatically learned based on this partial input from the user.

Upon naming and saving it, users can view their resulting ranking model in the rank builder view described above. This allows them to examine the weightings learned for each underlying metric and the resulting overall formula for the ranking. Users can then further customize the model by adjusting the weights manually through the rank builder interface if desired.

5

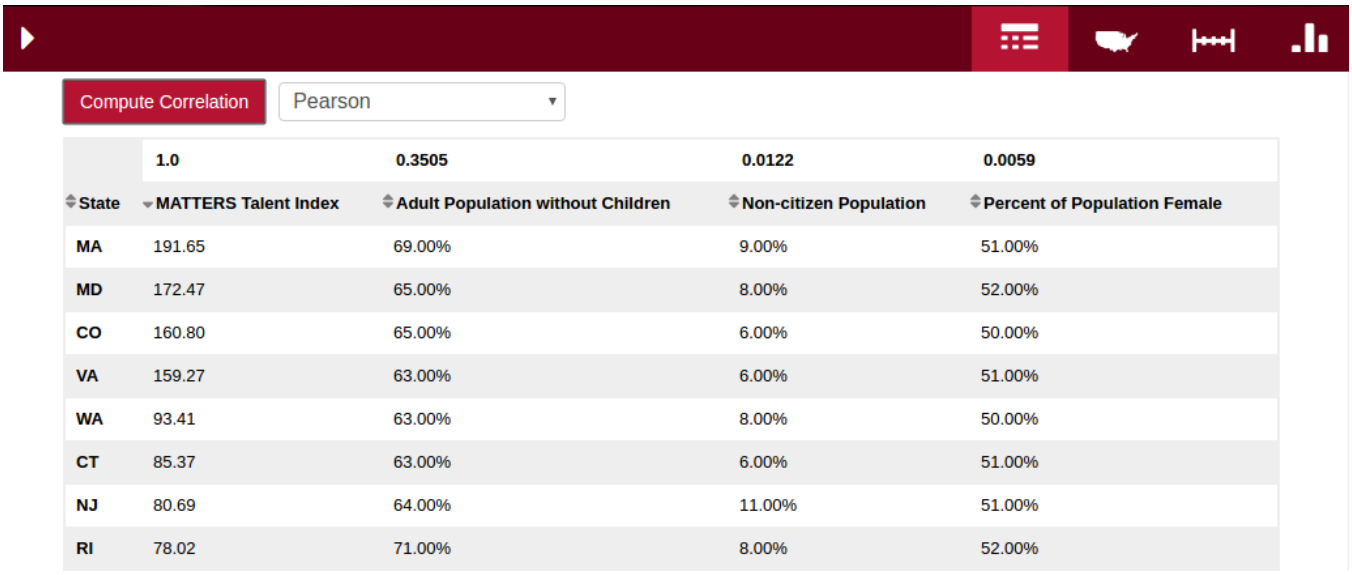


Figure 5: The MATTERS Talent ranking is compared with demographic data in the MATTERS table view.

4.2 Rank Views

Multiple visual displays in the MATTERS system provide further opportunity for evaluation and understanding of custom rankings. Displays offer descriptive analytics not only for all newly defined user rankings, but the entirety of the data in the MATTERS collection. This enables easy comparison between created rankings as well as between rankings and other data in the system. Customized views provide insight into the spatio-temporal nature of rankings created from data in the MATTERS system. Comparisons can be made across states and over time using the MATTERS Data Explorer feature. Figures 5 and 6 show some of the views available in the MATTERS system.

4.2.1 Table View. Custom rankings can be viewed in a table alongside other data in the MATTERS system. Data can be displayed in a number of configurations, varying the number of states, metrics, or years to be displayed. To further aid in understanding the relationships between data and rankings, correlation analysis is provided in the table view. A measure of the correlation between the first metric in the table with each of the other selected metrics is computed with the click of a button. Users can select a number of useful correlation measures including the Pearson Correlation Coefficient[12] and the Kendall Tau Coefficient [10] (Fig 5).

4.2.2 Timeline View. Time series analysis is provided for all MATTERS data including custom rankings in the timeline view. Users can select a set of states to view in the chart and compare their relative performance and changing trends over time (Fig 6a).

4.2.3 Map View. A choropleth map view allows users to compare rank values across all states at once. This view paints a picture of the distribution of rankings throughout the country at a glance. A sequential color scheme [2] allows users to easily discern and evaluate the relative performance of states and their geographic neighbors. A comparison of maps highlights the regional differences in rankings (Fig.6b).

4.2.4 State Profile View. Finally, MATTERS provides individual State Profile views. These displays show the values for each underlying metric contributing to the rank of an individual state. The colors red, yellow, and green indicate the performance of each metric as compared to the rest of the states, and trend arrows show whether the state has been improving or declining over time. This view can help users gain insight into the impact of each metric on the composite score for an individual state. Comparisons between profile views expose the differences in individual metrics which impact the relative performance of states (Fig. 6c).

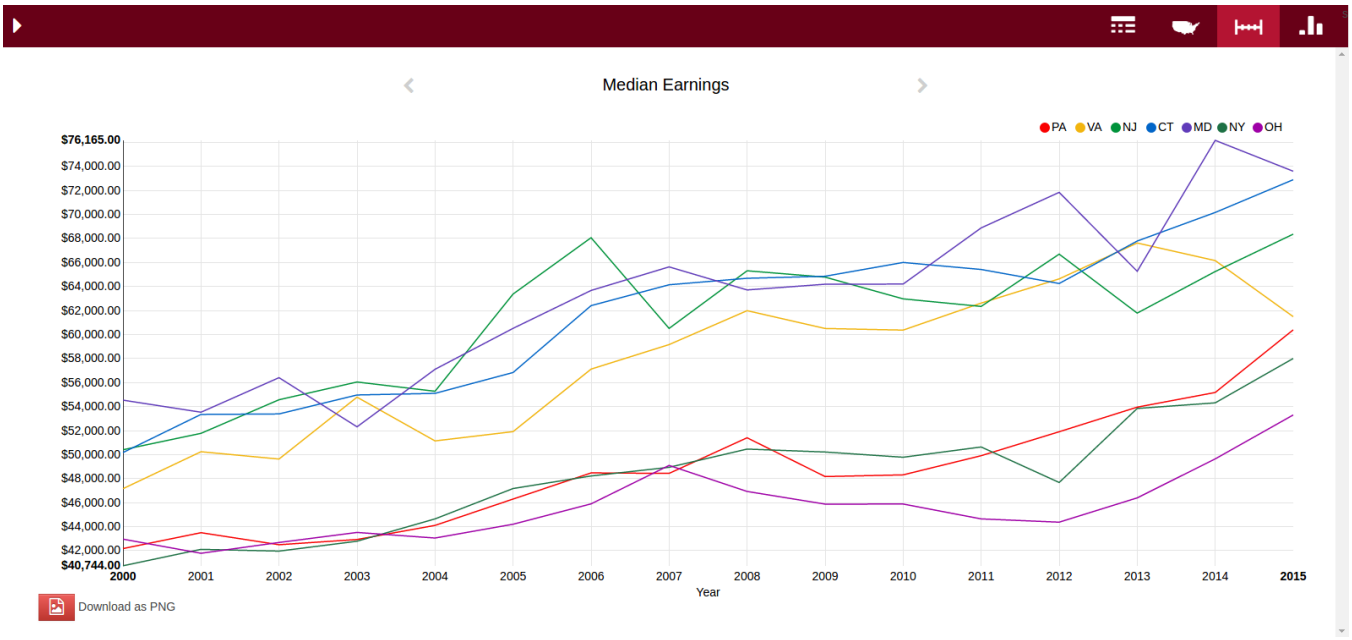
5 PRELIMINARY EVALUATION

The design of tools for interactive ranking exploration poses a number of challenges and open research questions. The MyRanker paradigm introduces questions regarding whether we can successfully learn rankings based on partial user input, and how to evaluate the "goodness" of such a ranking. Will the rankings we learn achieve the goal of the end user (economic or otherwise motivated)? Or might they contain bias and possibly put certain players into an unfair disadvantage. To begin to answer these questions we evaluate the utility of the MyRanker framework applied to the MATTERS system.

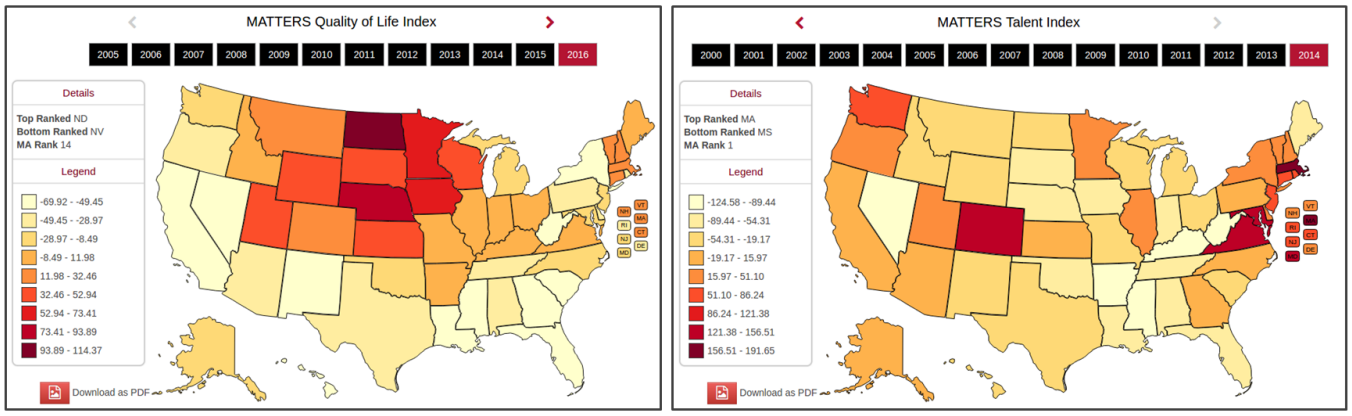
5.1 MyRanker Learning Module Evaluation

The MyRanker Framework provides a plug-and-play strategy wherein any pairwise learning-to-rank method can be easily incorporated into the ranking learner module. In the MATTERS interactive learn-to-rank tool we employ the method of regularized least squares given in [16] called RankRLS. The authors have provided a public software package for learning-to-rank. [6]

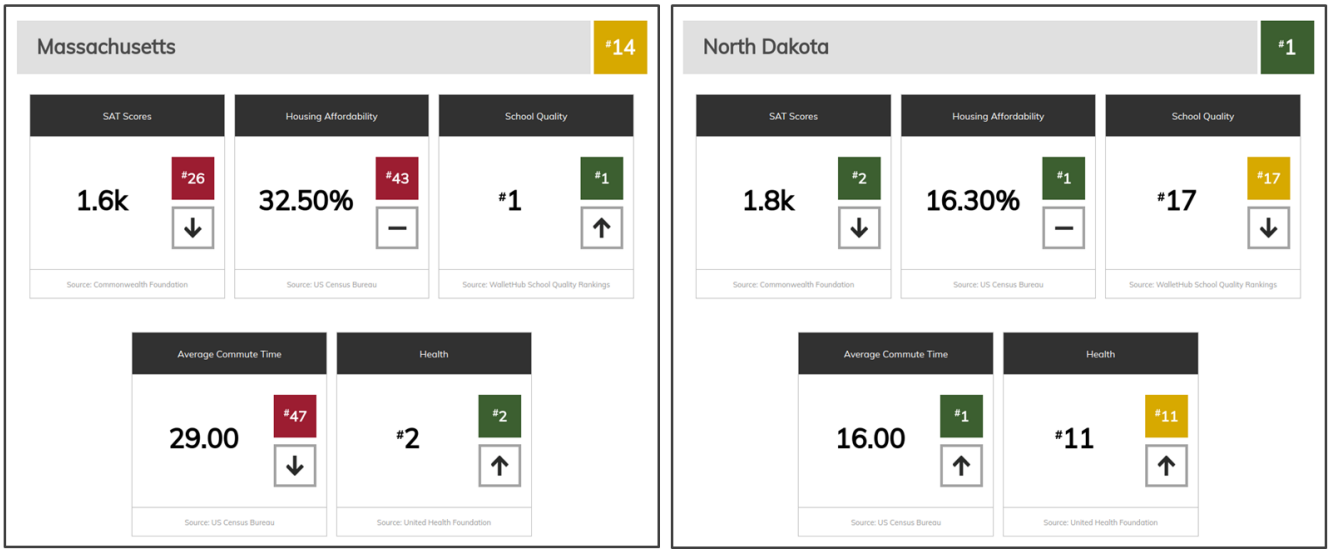
[6] http://staff.cs.utu.fi/~aatapa/software/RLScore/index.html



(a) The MATTERS timeline view shows how earnings data changes over time for a set of states.



(b) MATTERS rankings are compared as choropleth maps in the MATTERS map view.



(c) The metric values which make up a custom ranking are compared using the MATTERS State Profile.

Figure 6: Ranking views in MATTERS.

Number of States	Number of Pairs	cindex	tau
2	2	0.50	0.00
3	6	0.64	0.28
4	12	0.68	0.37
6	30	0.80	0.60
8	56	0.83	0.66
12	132	0.86	0.73
16	240	0.88	0.77
24	552	0.92	0.84

Table 1: The impact of the number of training pairs of states used to predict the MATTERS Cost index using RankRLS.

To evaluate the performance of this pairwise model on the data in the MATTERS warehouse, we would like to determine first how well RankRLS can learn an existing ranking of states, and second how much information is necessary to collect from users for a high quality ranking. To perform a preliminary assessment of the performance of the learn-to-rank tool we take advantage of rankings in our system which have been designed by experts from the Massachusetts High Tech Council. Four indices to measure the economic competitiveness of states have been constructed using the manual MATTERS Rank Builder interface – one for each of the data categories in the MATTERS system: Talent, Tax Climate, Cost of Doing Business and Quality of Life. Each metric consists of a weighted combination of metrics in the MATTERS system. We choose one, the 2014 MATTERS Cost of Doing Business Index, and evaluate how well RankRLS can learn this ranking.

Evaluation Metrics. Two ranking correlation measures are used to evaluate the RankRLS model. The first measure denoted "cindex" is a simple measure of pair concordance between the predicted ranking and the true ranking. A pair (a, b) is concordant if the predicted rank of a is greater that the predicted rank of b and this corresponds to a true rank of $a > b$. The cindex score is computed as the fraction of concordant pairs out of all pairs. This measure yields a number between 0 and 1, with 0.5 indicating random performance. We also employ the well-know statistical Kendall Tau [10] correlation coefficient. This measures the ordinal association between two rankings. A tau score of 1 indicates a perfect match, -1 indicates that one ranking is the reverse of the other, and 0 that the two rankings are independent.

The Cost of Doing Business ranking is computed using the metrics *Retail Price of Electricity, Median Earnings, Average Family Health Insurance Premium, and UI Premium per Employee.* Each metric is weighted evenly. Training on all possible pairs of states and the data from the 4 underlying metrics in the ranking yields a cindex of 0.95 and Kendall Tau score of 0.91. This was determined using 5-fold cross validation repeated over 10 randomized trials and taking the average score.

To evaluate the impact of the number of training pairs on the ranking we randomly selected subsets of states of varying sizes and formed all pairs to train on. Then the model was tested on the rest of the data in a cross-validation manner. The average of 10 randomized trials were taken. Table 1 shows the impact of the number of pairs on the accuracy of the learned rankings. As expected, performance increases with the amount of training data.

Number of Metrics	cindex	tau
0	0.95	0.91
2	0.94	0.88
4	0.93	0.87
8	0.92	0.84
12	0.90	0.80
16	0.88	0.77
18	0.87	0.75

Table 2: The impact of the number of metrics used to predict the MATTERS Cost index using RankRLS.

To evaluate the impact of noise from other underlying metrics, Table 2 shows the performance of the RankRLS model when trained on collections of metrics of increasing size. For each trial we include the 4 underlying metrics which make up the Cost of Doing Business ranking, and then randomly select additional datasets in the MATTERS system to train on as well. In the experiment given in the first row of Table 2 we train only on the 4 metrics, then in the next row 2 additional metrics are added, and so on. 10 randomized trials are averaged for each experiment. We can see that the pairwise learning model is impacted by noise from other data. This suggests the MyRanker framework could benefit from the addition of a feature selection or regularization step.

5.2 Use Case in Ranking for Talent Understanding

In addition to constructing rankings, the goal of an interactive ranking paradigm is to help users better interpret and understand ranking models. In the MATTERS system, the interplay of rank specification tools and data visualizations facilitates this understanding. Ranking models can be inspected using the rank builder interface (Section 4.1.1), and rankings can be compared with underlying raw data using the many MATTERS views (Sec 4.2).

Here we give an example of how MATTERS can be used to perform this type of analysis, by considering the MATTERS Talent Index. It is easy to look a state like California with its hub of innovation in Silicon Valley and observe that science and technology can drive prosperity with astounding impact. Policy makers and business leaders in other states may wonder how to foster similar drivers of economic growth in their own states. When the Massachusetts High Tech Council was developing the MATTERS system, they identified the ability to attract and maintain a highly skilled talent pool as one key to success in this area.

MATTERS provides a custom Talent Index designed by domain experts based on 4 metrics: *STEM Degrees Per Capita, Relocation of College Educated Adults, Bachelor's Degree Holders in Workforce, and Tech Employment as Percent of Total Employment.* The views and comparison tools in MATTERS can easily provide insight into this ranking, ensuring that users do not simply have to accept the index at face value. They may be interested in evaluating a number of concerns. Perhaps this ranking could contain implicit negative bias, measuring not just the underlying metrics, but also a trend based on race, gender or another undesirable measure. Or users might wonder how much additional insight this ranking really provides compared to other measures based on different metrics.

Figure 5 shows the MATTERS Talent Index displayed in the table view along with three demographic datasets. A highly undesirable ranking might reflect such information. However, using the correlation button, the Pearson Correlation Coefficient for the Talent Index compared to each demographic dataset is displayed in the top row of the table. We learn that none of these datasets are highly correlated with the Talent Index.

In another view we can observe how the Talent Index ranking compares to the other MATTERS Indices. It could be that the states with the highest quality of life attract the most talent, and therefore the Talent ranking might not be a unique measure. However, when we compare the choropleth maps shown in Figure 6b we can observe that the Talent ranking clearly has a different distribution across states from that of the MATTERS Quality of Life Index. Therefor the Talent Index is indeed measuring different phenomena. A deep dive into the factors contributing to individual state scores could provide additional insight using the State Profile view (Figure 6c).

With these strategies, the ranking model can be evaluated in the context of all available data. Compared with demographic information and other rankings, users can ascertain whether it is providing a meaningful measure of states.

6 DISCUSSION AND FUTURE WORK

Traditional approaches to the design of economic indices are based in theory and start with expert opinion. New approaches to data-driven analysis may provide previously unseen insights by leveraging open data and state-of-the-art machine learning techniques. In this work we introduce a vision for interactive learn-to-rank tools to facilitate the creation, exploration, and understanding of rankings. Our plug-and-play MyRanker framework provides interlinked components to achieve this goal in a data analytics system. We demonstrate the power of this paradigm for economic competitiveness evaluation as part of the MATTERS dashboard.

The brief evaluation presented here indicates potential for the construction of high quality rankings using partial knowledge specified by a user. Additional evaluation of pairwise learning based on user preferences is required to understand the trade-offs between user experience and accuracy. Further, highly usable interfaces are required to bring the power of the ranking algorithm to a non-expert audience. User studies are required to evaluate the utility of the learn-to-rank tools proposed in this paper. A simple to understand graphical display and intuitive interactions must capture the intent of the user, and facilitate their understanding and trust in the learned model. While alternate views are feasible, their relative utility must be formally studied for a given application context and user group targeted. Continued study and subsequent refinement of this new class of interactive learn-to-rank tools is planned.

Finally, to aid in the interpretation of learned rankings, a number of useful extensions to the MyRanker paradigm are being explored. We have shown that comparisons between underlying metrics and rankings provide useful insights regarding the explanatory power of ranking models. Tools to automatically learn those relationships would greatly aid in this type of analysis. Regularization and feature reduction techniques could provide an integral piece of the data analysis pipeline to further this understanding. Reducing the original input feature space to select only data which have the greatest impact on the position of individual objects can increase both user understanding and thus acceptance of constructed rankings.

7 ACKNOWLEDGEMENTS

The authors would like to thank the Massachusetts High Tech Council for their guidance and collaboration on the MATTERS project and partial funding support. We also appreciate the contributions of students who have worked developing the MATTERS system, especially Dmytro Bogatov and Jillian Hennessy for their work on the Rank Builder Tool. Student contributers are listed on the project website at http://davis.wpi.edu/dsrg/PROJECTS/MATTERS. We also acknowledge partial funding support from the Student Research Participation Program at NSRDEC administered by ORISE through an interagency agreement with the U.S. DOE.

REFERENCES

[1] Dmytro Bogatov and Jillian R Hennessy (advisor: Elke Rundensteiner; sponsor: Massachusetts High Technology Council). Data matters: Customizing economic indices to measure state competitiveness. 2016. URL https://web.wpi.edu/Pubs/E-project/Available/E-project-042616-011522/.

[2] Cynthia A Brewer, Mark Harrower, B Sheesley, A Woodruff, and D Heyman. Colorbrewer 2.0. Accessed June 2017. URL http://www.ColorBrewer.org/.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96. ACM, 2005.

[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, pages 129–136. ACM, 2007.

[5] Ross DeVol, Joe Lee, and Minoli Ratnatunga. 2016 State Technology and Science Index: Sustaining americafis innovation economy. 2016.

[6] Alex Fortier, Westley Russell, Kevin Mee, and Long Nguyen Duc (advisor: Elke Rundensteiner; sponsor: Massachusetts High Technology Council). Advancing matters. 2015. URL https://web.wpi.edu/Pubs/E-project/Available/E-project-042915-094249/.

[7] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Support vector learning for ordinal regression. 1999.

[8] White House. College scorecard. 2013. URL https://collegescorecard.ed.gov/data/.

[9] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[10] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2): 81–93, 1938.

[11] Caitlin Kuhlman, Karthikeyan Natesan Ramamurthy, Prasanna Sattigeri, Aurelie C Lozano, Lei Cao, Chandra Reddy, Kush R Varshney, and Aleksandra Mojsilovic. How to foster innovation: a data-driven approach to measuring economic competitiveness. *IBM Journal of Research and Development*, in press.

[12] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[13] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[14] Rodica Neamtu, Ramoza Ahsan, Jeff Stokes, Armend Hoxha, Jialiang Bao, Stefan Gvozdenovic, Ted Meyer, Nilesh Patel, Raghu Rangan, Yumou Wang, Dongyun Zhang, and Elke A. Rundensteiner. Massachusetts Economy and Technology Index System. In *Proceedings of the International Workshop on Data Science for Macro-Modeling*, DSMM'14, pages 11:1–11:2. ACM, 2014.

[15] Raymond S Nickerson. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of general psychology*, 2(2):175, 1998.

[16] Tapio Pahikkala, Evgeni Tsivtsivadze, Antti Airola, Jorma Boberg, and Tapio Salakoski. Learning to rank with pairwise regularized least-squares. In *SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, volume 80, pages 27–33. Citeseer, 2007.

[17] Klaus Schwab, Xavier Sala-i Martin, et al. The global competitiveness report 2016-2017. World Economic Forum, 2017.

[18] Hal R Varian. Big data: New tricks for econometrics. *The Journal of Economic Perspectives*, 28(2):3–27, 2014.

[19] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1192–1199. ACM, 2008.

[20] Li Zhou. Obama's new college scorecard flips the focus of rankings. *The Atlantic*, 2015. URL https://www.theatlantic.com/education/archive/2015/09/obamas-new-college-scorecard-flips-the-focus-of-rankings/405379/.

Interactive Unsupervised Clustering with Clustervision

Work-in-Progress

Bum Chul Kwon
IBM T.J. Watson Research Center
bumchul.kwon@us.ibm.com

Ben Eysenbach
Massachusetts Institute of Technology
bce@mit.edu

Janu Verma
IBM T.J. Watson Research Center
jverma@us.ibm.com

Kenney Ng
IBM T.J. Watson Research Center
kenney.ng@us.ibm.com

Adam Perer
IBM T.J. Watson Research Center
adam.perer@us.ibm.com

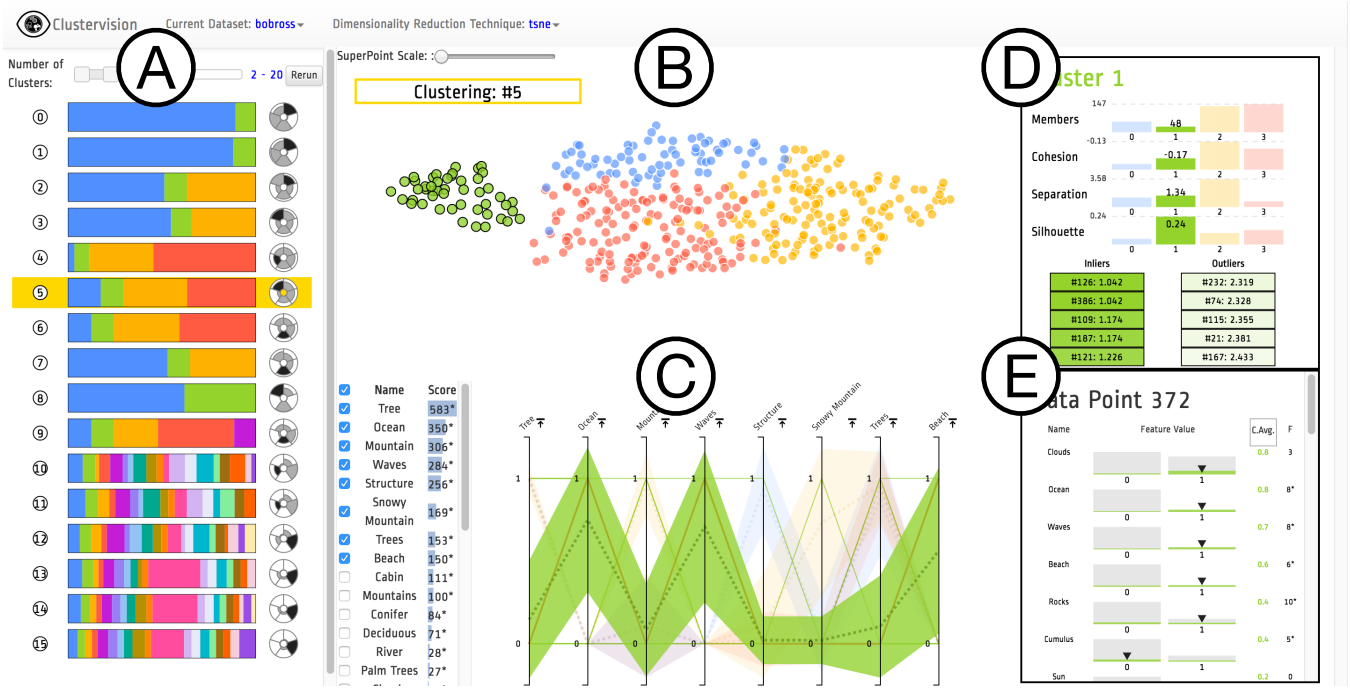


Figure 1: An overview of *Clustervision* on a dataset describing 403 paintings by the "Joy of Painting" artist Bob Ross. (A) *Ranked List of Clustering Results* shows 16 different clustering results that are sorted by the aggregated quality measures; (B) *Projection* shows a selected clustering result (highlighted in yellow in (A)) on a projection of data points colored corresponding to corresponding clusters; (C) *Parallel Trends* show the trends of feature values of data points within corresponding clusters in areas across parallel coordinates. Cluster 1 (Green Color) is highlighted; (D) *Cluster Detail* shows quality measures of a selected individual cluster (Cluster 1); (E) *Data Point* shows the feature value distribution of the selected cluster as well as the selected data point (Data Point 372 within Cluster 1).

KEYWORDS

Unsupervised Clustering, Visual Analytics, Quality Metrics, Interactive Visual Clustering

ACM Reference format:
Bum Chul Kwon, Ben Eysenbach, Janu Verma, Kenney Ng, and Adam Perer. 2017. Interactive Unsupervised Clustering with Clustervision. In *Proceedings of IDEA Workshop Submission, Halifax, NS, Canada, 2017 (IDEA'17),* 4 pages.
https://doi.org/10.475/123_4

IDEA'17, 2017, Halifax, NS, Canada

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

1 INTRODUCTION

Clustering, the process of grouping together similar items into distinct partitions, is a common type of unsupervised machine learning that can be useful for summarizing and aggregating complex multi-dimensional data. However, data can be clustered in many ways, and

there exist a large number of algorithms designed to reveal different patterns. While having access to a wide variety of algorithms is helpful, in practice, it is quite difficult for data scientists to choose and parameterize algorithms to get the clustering results relevant for their dataset and analytical tasks. To alleviate this problem, we built a clustering analysis system, *Clustervision*, that helps ensure data scientists find the right clustering among the large amount of techniques and parameters available. Our system clusters data using a variety of clustering techniques and parameters and then recommends good clustering results utilizing a variety of quality scoring metrics. In addition, users can guide the system to produce more relevant results by providing task-relevant constraints on the data. Our visualization interface allows users to find high quality clustering results, explore the clusters using a variety of coordinated visualization techniques, and select the cluster result that best suits their task.

2 CLUSTERVISION

In order to support interactive exploration of clustering results, we propose *Clustervision*. In this paper, we demonstrate the system using a small but illustrative dataset of 403 paintings produced on the PBS show "The Joy of Painting". Over the course of the 403 episodes, a variety of diverse landscapes were painted, which were manually coded by FiveThirtyEight[1] using 67 features (e.g. trees, water, mountains, and weather elements).

After a dataset is loaded into the tool, *Clustervision* computes and evaluates all possible combinations of clustering techniques and parameters. In this configuration, *Clustervision* will use three clustering techniques (k-means, Spectral Clustering, and Agglomerative Clustering) and 19 parameter configurations (k=2-20), resulting in 58 clustering results. The system can also optionally include more clustering techniques and parameters. Each of the clustering results are then analyzed using, by default, 5 quality metrics (Calinski-Harabaz[8], Silhouette[11], Davies-Bouldin[5], S_{Dbw}[7], and Gap Statistic[13]). As each of these quality metrics aim to compute quality using different properties of the clusters (e.g. variance, within-cluster distance, between-cluster distance, density), we chose not to rely on a single metric but instead a variety of diverse metrics. By default, the top 3 highest ranking results from each metric are presented to the user, resulting in 15 results top results for the user to consider. In order to ensure the results aren't too similar, an item will only be considered as a top result if its at least 5% different from another top result.

Figure 1(a) shows an example of the top 15 clustering results. Each row features a clustering summary glyph, where each colored stripe represents a color whose width is proportional to the number of data points in that cluster. Each cluster has a unique color that is consistently used across all views in the UI. On the right is a quality summary glyph that shows values of each of the five quality metrics. Similar to a pie chart, the glyph is a circular region divided into five equal slices for each of the metrics.

In order to understand if a particular clustering result is relevant to the analytical task, users often need to see their data points in context of the cluster groupings. The *Projection* view encodes data points as circular elements in a two dimensional space, resembling a scatterplot, as shown in Figure 1(b). However, instead of plotting

[1] https://fivethirtyeight.com/features/a-statistical-analysis-of-the-work-of-bob-ross/

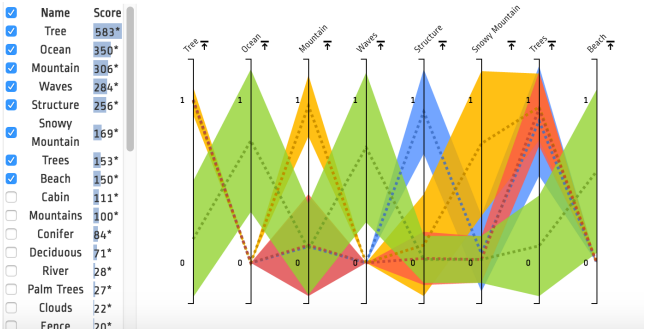


Figure 2: The *Parallel Trends* view is similar to parallel coordinates, but in order to simplify the complexity of many lines, the view focuses on showing the trends of each cluster. *Parallel Trends* has vertical axes that represents each feature of the data points. However, instead of drawing a line crossing the axes for each data point as in parallel coordinates, *Parallel Trends* draws an area path per cluster. The intervals cross each axes, where the vertical ends represent standard deviation or 95% confidence intervals for the corresponding features.

the data on only two dimensions of the data, *Clustervision* uses dimensionality reduction techniques (e.g. t-SNE [10]) to synthesize all of the dimensions. The main use of the *Projection* view is to have a consistent and stable representation, as the positions of the data points remain stable across all clustering results. Although the position of the data points gives clues to the distance and separation between clusters, users get more evidence about the underlying properties of the clusters from the other views. The *Projection* view serves as one way to explore both individual data points and clusters. Most importantly, it allows users to use other views to get more details about the selected data points and clusters.

In order to help summarize the clustering results, the *Ranked Features* and *Parallel Trends* views are coordinated with the projection view and shows information about the features of the selected clustering result. One of the challenges associated with unsupervised clustering is that even after clusters are defined by a technique, it is difficult to summarize why the cluster groupings were made. In an attempt to retrieve the features responsible for the separation, we utilize univariate statistics to compute whether there is a statistically significant relationship between each feature and each cluster. We consider this a classification task, where each cluster is a class, and compute the analysis of variance (ANOVA) for the entire dataset. The result scores based on the ANOVA F-Value allow us to rank each feature in order of importance. These important features are displayed as a ranked list in the *Ranked Features* view, where each feature name is augmented with a numeric importance score and a corresponding bar chart, as shown in Figure 1(c).

The *Parallel Trends* is similar to parallel coordinates, but in order to simplify the complexity of many lines, initially the view only shows the trends of each cluster. As in parallel coordinates, *Parallel Trends* has vertical axes that represents each feature of the data points. However, instead of drawing a line crossing the axes for each data point as in parallel coordinates, *Parallel Trends* draws an area path per cluster. The intervals cross each axes, where the vertical

ends represent standard deviation or 95% confidence intervals for the corresponding features. Then, a dotted line is drawn on top of the area path per cluster to show the mean values for each cluster for the corresponding data feature. To see details of a cluster, users can click on an area path to show individual lines that represent corresponding data points within the cluster as shown in Figure 1(c). This implementation also allows users to sort axes, switch axes, and filter on specific feature values on each axis, which are interaction techniques common to parallel coordinates.

For example, in the selected Bob Ross clustering shown in Figure 2, the top features most responsible for the cluster grouping are the presence of trees, mountains, and oceans in paintings. This ranked list in conjunction with the *Parallel Trends* views help show how these features correlate with the clusters. The Green cluster has uniquely high values in Ocean, Waves, and Beach, giving a clear indication that this cluster represents the ocean-oriented paintings of Ross. This cluster is demonstrably different from the Yellow cluster (which has high values of tree, mountain, snowy mountains, and trees), the Blue cluster (with Structures), and the Red cluster (with tree and trees). While only the top 8 features are shown, other features can be added by selecting them.

The *Cluster Detail* view appears when users select a particular cluster from the *Projection* or *Parallel Trends* views. This view is designed to present a summary of the clusters using statistics and prototypes. For the selected cluster, the number of data points that are members of the cluster is shown as a labeled bar that is the same color of the cluster. This number is put in context with all of the other cluster sizes by showing translucent bars representing each cluster to form a bar chart. Similar bar charts are shown for statistics summarizing the cluster, such as cohesion, separation, and silhouette scores, as shown at the top of Figure 1(d). In addition to these statistical summaries, the *Cluster Detail* view also shows members of the cluster that typical or atypical for the cluster based on the distance metric, as inliers and outliers.

The *Data Point* view appears when users select or mouseover a data point in the *Projection* or *Parallel Trends* views. The *Data Point* provides details about the actual values of a data points features. However, this view also puts them in the context of other other data points by presenting the distribution of values alongside each value. For binary variables and categorical feature values with less than five levels, we show histogram rather than density plot and provide triangle marks to show the selected data point as seen in Figure 1(e).

Users can sort features by their name, value, cluster average value, and importance. The importance calculation is similar to the technique used in the *Ranked Features* view. However, here the technique considers assigns the selected cluster as a first class, and all other clusters as a second class. By computing an ANOVA using these cluster-centric classes, it is possible to determine which features are responsible for why the selected cluster is different from all other clusters. This option presents the most important features at the top of this view, making it easy to compare between data points and clusters by mouse-overing regions of the interest in the *Projection* view.

Users can also interactively request new results by setting up constraints with respect to specific data points. Users can select multiple data points and tell the system that they need to be either in the same cluster or in separate clusters. Then, the system filters clustering results based on the requirements set.

3 RELATED WORK

There have been many previous visualization systems that attempt to employ clustering to support high dimensional data analysis. Hierarchical Clustering Explorer [12] allows users to investigate an overview of a clustering result and to compare details of clusters by using coordinated displays. VISTA [4] enables users to visually view clusters of a clustering result on 2D projection and apply internal quality metric scores. Dicon [3] visualizes multidimensional clusters' quality as well as attribute-based information through icon-based visualization. Unlike *Clustervision*, these systems do not support comparison between multiple clustering results.

Some systems allow users to provide feedback on clustering results so that the next run applies the inputs. desJardins et al. [6] proposed a technique to iteratively run and visualize constrained clustering with constraints made by users. iVisClustering allows users to adjust cluster hierarchy and to re-label individual data items (i.e., documents) into another cluster [9]. Cluster Sculptor also allows users to update cluster labels on a 2D projection [2]. Boudjeloud-Assala *et al.* proposes an interactive visual clustering system that allows users to define seeds and limits of clusters for steering the clustering process [1]. While these systems help steer the user toward better clustering results, the user must define how to make the clustering better rather than receiving recommendations from the system, unlike *Clustervision*.

4 CONCLUSION

In this paper, we described the features of *Clustervision*, a work-in-progress which we believe is a promising interface to help data scientists find meaningful clusterings of their data. By integrating clustering techniques and quality metrics with coordinated visualizations, the system allows users to interactively explore and analyze clustering results at various levels. Although we demonstrated this system on a small dataset in this paper, we are currently deploying this system with a team of data scientists to find meaningful clusters of patients that share complex diseases, which they plan to publish in an upcoming medical journal.

REFERENCES

[1] Lydia Boudjeloud-Assala, Philippe Pinheiro, Alexandre BlanschÃl', Thomas Tamisier, and BenoÃőt Otjacques. 2016. Interactive and iterative visual clustering. *Information Visualization* 15, 3 (July 2016), 181–197.

[2] P. Bruneau, P. Pinheiro, B. Broeksema, and B. Otjacques. 2015. Cluster Sculptor, an interactive visual clustering system. *Neurocomputing* 150, Part B (Feb. 2015), 627–644.

[3] N. Cao, D. Gotz, J. Sun, and H. Qu. 2011. DICON: Interactive Visual Analysis of Multidimensional Clusters. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (Dec. 2011), 2581–2590.

[4] Keke Chen and Ling Liu. 2004. VISTA: Validating and Refining Clusters Via Visualization. *Information Visualization* 3, 4 (Dec. 2004), 257–270.

[5] David L. Davies and Donald W. Bouldin. 1979. A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Mach. Intell.* 1, 2 (Feb. 1979), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

[6] Marie desJardins, James MacGlashan, and Julia Ferraioli. 2007. Interactive Visual Clustering. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI '07)*. ACM, New York, NY, USA, 361–364.

[7] Maria Halkidi and Michalis Vazirgiannis. 2001. Clustering Validity Assessment: Finding the Optimal Partitioning of a Data Set. In *Proceedings of the 2001 IEEE*

International Conference on Data Mining (ICDM '01). IEEE Computer Society, Washington, DC, USA, 187–194. http://dl.acm.org/citation.cfm?id=645496.657864

[8] Marcin Kozak. 2012. âĂIJA Dendrite Method for Cluster AnalysisâĂİ by CaliÂĎski and Harabasz: A Classical Work that is Far Too Often Incorrectly Cited. *Communications in Statistics - Theory and Methods* 41, 12 (2012), 2279–2280. https://doi.org/10.1080/03610926.2011.560741 arXiv:http://dx.doi.org/10.1080/03610926.2011.560741

[9] Hanseung Lee, Jaeyeon Kihm, Jaegul Choo, John Stasko, and Haesun Park. 2012. iVisClustering: An Interactive Visual Document Clustering via Topic Modeling. *Computer Graphics Forum* 31, 3pt3 (June 2012), 1155–1164.

[10] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[11] Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53 – 65. https://doi.org/10.1016/0377-0427(87)90125-7

[12] Jinwook Seo and B. Shneiderman. 2002. *Interactively exploring hierarchical clustering results*.

[13] Robert Tibshirani, Guenther Walther, and Trevor Hastie. 2000. Estimating the number of clusters in a dataset via the Gap statistic. 63 (2000), 411–423.

Learning Strategies in Game-theoretic Data Interaction*

Ben McCamish
Oregon State University
mccamisb@oregonstate.edu

Arash Termehchy
Oregon State University
termehca@oregonstate.edu

Behrouz Touri
University of Colorado Boulder
touri@colorado.edu

Liang Huang
Oregon State University
liang.huang@oregonstate.edu

ABSTRACT

As most database users cannot precisely express their information needs in the form of database queries, it is challenging for database query interfaces to understand and satisfy their intents. Database systems usually improve their understanding of users' intents by collecting their feedback on the answers to the users' imprecise and ill-specified queries. Users may also learn to express their queries precisely during their interactions with the database system. In this paper, we report our progress on developing a formal framework for representing and understanding information needs in database querying and exploration. Our framework considers querying as a collaboration between the user and the database system to establish a *mutual language* for representing information needs. We formalize this collaboration as a signaling game between two potentially rational agents: the user and the database system. We empirically analyze the users' learning mechanisms using a real-world query workload. Given the users' learning mechanisms, we extend and evaluate some reinforcement learning mechanisms for the database system to establish effectively a mutual language between with adapting users. We believe that this framework naturally models the long-term interaction of users and database systems.

KEYWORDS

Usable query interfaces, Interactive query interfaces, Game theory, Rational agents, Reinforcement Learning, Intents and Queries

ACM Reference format:
Ben McCamish, Arash Termehchy, Behrouz Touri, and Liang Huang. 2017. Learning Strategies in Game-theoretic Data Interaction. In *Proceedings of , Halifax, Nova Scotia, Canada, August 14th, 2017 (KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17)),* 9 pages.
DOI:

1 INTRODUCTION

Because most users do not know database query languages, such as SQL, the structure, and/or the content of their databases, they cannot precisely express their queries [10, 11, 20, 21]. Hence, it is challenging for database query interfaces to understand and satisfy users' information needs, i.e., intents. Developing usable query interfaces that can effectively answer imprecise and ill-specified queries has attracted a great deal of attention in the last decade [6, 10, 11, 18, 19, 23]. Ideally, we would like the user and query interface to establish a *mutual understanding* where the query interface understands how the user expresses her intents and/or the user learns to formulate her queries precisely.

Researchers have proposed several techniques in which a database system may improve its understanding of the true information need behind a query [10, 11, 18, 19]. These methods generally assume that the way a user expresses her intents remains generally intact over her course of interaction with the database. However, users may leverage their experience from previous interactions with the database to express their future intents more precisely. For example, the more a user interacts with a relational database, the more familiar she may become with the important and relevant attributes and relations in the database, and therefore, the more precisely she may express her queries over the database. Moreover, current methods mainly improve the mutual understanding of a user and a database for a single information need. Nevertheless, many users explore a database to find answers for various information needs potentially over a long period of time. For example, a biologist may query a reference database that contains information about certain genes and proteins for several years. Thus, a natural and realistic model for database interaction should consider the long-term adaptation for both users and database systems during their interactions.

To address the aforementioned shortcomings, we have recently proposed a novel framework that models database querying as a collaborative game between *two active and potentially rational agents:* the user and query interface [26]. The common goal of the players is to reach a mutual understanding on expressing intents in the form of queries. The players may reach this goal through communication: the user informs the database system of her intents by submitting queries, the database system returns some results for the queries, and user provides some feedback on how much the returned results match her intents, e.g., by clicking on some desired answer(s). The user may also modify her query to better reflect her intent after exploring the returned answers. Both players receive some reward based on the degree by which the returned answers satisfy the intents behind queries. We believe that this framework naturally models the long-term data interaction between humans and database systems.

In this paper, we provide an overview of our proposed framework. Also, using a real-world query workload, we investigate how users learn to map their intents to queries. We analyze various reinforcement learning strategies for users and whether users frequently explore various alternatives of expressing a certain intent, or preserve relatively successful strategies. Our analysis indicate that while users show some exploration behavior, they mainly reuse successful

*A portion of this work was been previously published in HILDA titled *A Game-theoretic Approach to Data Interaction: A Progress Report*

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada
2017. .
DOI:

methods of expressing intents. Furthermore, we extend two reinforcement learning strategies, namely UCB-1 [2] and Roth and Erv [29] algorithms for the database system learning. UCB-1 is a popular choice for on-line and reinforcement learning in information retrieval systems as these systems model the interaction between the user and the information system as a Multi Armed Bandit problem [28, 33]. Our empirical results show that Roth and Erv algorithm often outperforms UCB-1 where users learn to modify their strategies.

2 SIGNALING GAME FRAMEWORK

Next, we present an overview of the components of our model from [26].

2.1 Intent

An *intent* e represents an information need sought after by a user. We assume that each intent is a query in a fixed query language, e.g., SQL. The set of possible intents is infinite. However, in practice a user has only a finite number of information needs in a finite period of time. Hence, we assume the number of intents for a particular user is finite. We index each intent over a database instance by $1 \leq i \leq m$.

2.2 Query

Because a user may not be able to precisely formulate her intent e, she may submit query $q \neq e$ to the database instead. Of course, the user still expects that the DBMS returns the answers of intent e for query q. Queries may be formulated in the same language used the represent intents. For example, one may submit an ill-specified SQL query, e.g., do *not* use the right joins, to express her intent which is also a SQL query. But, it may be sometimes hard for users to express their queries using formal query languages [20]. For instance, some users may prefer to use languages that are easier to use, e.g., keyword or natural language queries, to express their intents. Our model does not require the language that describes intents and the language used to specify queries to be the same. Hence, the intent of a query over a relational database may be precisely formulated by a SQL query, but users may use keyword queries to express that intent. A user in practice submits a finite number of queries in a finite time period. Hence, we assume that the set of all queries submitted by a user is finite. We index each query over a database instance by $1 \leq j \leq n$. Table 1 shows a fragment of a database with relation $Grade$ that contains information about students and their grades. A user may want to find the grade for student Sarah Smith, which can be represented as (Keyword) query 'Sarah Smith CS'. But, since she does not know the content of the database, she may submit the under specified query 'Smith'.

2.3 Result

Given a query q over a database instance I, the database system returns a set of tuples in I as the response to q. Because the database system knows that the input query may not precisely specify the user's intent, it considers various methods to find answers that satisfy the information need behind the query [11]. It often uses a scoring function that scores all candidate tuples according to their degree of relevance to the input query and return the ones with higher scores, i.e., most relevant tuples [11].

2.4 Strategies

The user strategy indicates the likelihood by which the user submits query q_j given that her intent is e_i. Hence, a user strategy, U, is a $m \times n$ row-stochastic matrix from the set of intents to queries. Similarly, the database system strategy shows the result returned by the database system in the response of the input query q_j. In other words, the database strategy is an $n \times o$ row-stochastic matrix from queries to the set of possible results. We note that our model does not require the database system to materialize and maintain its strategy as an $n \times o$ matrix. A database system may implement its strategy using a function over some finite set of queries and tuples [11, 25]. Each pair (U, D) is called a *strategy profile*. Consider again the university database shown in Table 1. Tables 1(a) and 1(b) show a user's intents and the queries they submit to the database system to express these intents, respectively. Table (c) illustrates a strategy profile for these sets of intents and queries.

2.5 Stochastic Strategies

Normally, database systems adapt strategies with only 0/1 entries [11]. For example, given the input query q_j, they may return a set of tuples whose scores according to a fixed and deterministic scoring function is above some given threshold. Hence, their query answering algorithms are usually deterministic and do not involve any randomization. Nevertheless, it has been shown that this approach does not allow the database system to collect feedback from the users on sufficiently diverse set of tuples because users can provide feedback only on tuples that have a relatively high score according the scoring function. Since the users' feedback will remain biased toward those tuples, the database system will gain only a limited insight about the intents behind the query. This is particularly important in long-term interactions because the database system has more opportunities to communicate and learn about users' preferences. Hence, researchers propose adapting a more probabilistic strategy in which the database system with some probability may deviate from its scoring function and present other tuples to the user to collect their feedback. Of course, if the database system shows too many non-relevant tuples to the user, the user may give up using the system. Thus, it is necessary to have a trade-off between showing the tuples which the database system deems relevant to the input query and the ones that it is not sure to be relevant but interested to see users' feedback for them to balance the usability of the system in the short-term and improve its effectiveness in the long run. Empirical studies over large document collections show that it is possible to find such a trade-off and significantly improve the effectiveness of answering ill-specified queries [32]. We follow these results and assume that the database may adapt a probabilistic strategy.

2.6 Reward

After the user submits a query to the database system and is presented by a set ot tuples, she will provide some feedback on the returned results. This feedback may be implicit, e.g., click-through information or the amount of time spent on reading the information of a tuple, or explicit by marking some tuples as relevant and others as non-relevant. Obviously, the goal of both the user and the database system is to see as many relevant answers as possible in the returned results. Hence, we assume that both the user and the database system

receive some reward according to the effectiveness of the returned results after each interaction. We use standard effectiveness metric *NDCG* to measure the reward for the user and database system given a returned set of tuples [25]. The value of NDCG is between 0-1 and roughly speaking it is higher for the results with more relevant answers. Our framework can be extended for other standard effectiveness metrics, such as precision at k.

2.7 Signaling Game

We model the long-term interaction of the user and the database system as a repeated game with identical interests played between the user and the database system. At each round of the game, the user wants to receive information about a randomly selected intent e_i. She picks query q_j with probability U_{ij} according to her strategy to convey this intent to the database system. The database system receives the query and returns a result l_ℓ to the user with probability $D_{j\ell}$. The user provides some implicit or explicit feedback on l_ℓ and both players receive reward of $r(e_i, l_\ell)$ at the end of this interaction. Each player may modify its strategy according to the reward it receives at the end of each round. For example, the database system may reduce the probability of returning the results without positive feedback for the same query.

$$u(U, D) = \sum_{i=1}^{m} \pi_i \sum_{j=1}^{n} U_{ij} \sum_{\ell=1}^{o} D_{j\ell}\, r(e_i, l_\ell). \tag{1}$$

where π is the prior probability of choosing an intent by the user and r is the NDCG score. Neither of the players knows the other player's strategy. The players communicate only by sending queries, results, and feedback on the results. In this paper, we focus on an important question: how do users learn and update their strategies.

First_Name	Last_Name	Dept	Grade
Sarah	Smith	CS	A
John	Smith	EE	B
Hayden	Smith	ME	C
Kerry	Smith	CE	D

Table 1: A database instance of relation Grade

(a) Intents

Intent#	Intent
e_1	John Smith in EE
e_2	Kerry Smith in CE
e_3	Sarah Smith in CS

(b) Queries

Query#	Query
q_1	'Kerry Smith'
q_2	'Smith'

(c) A strategy profile

	q_1	q_2
e_1	0	1
e_2	1	0
e_3	0	1

	l_1	l_2	l_3
q_1	0	1	0
q_2	0.5	0	0.5

Table 2: Intents, queries, and a strategy over the DB in Table 1.

3 OPEN PROBLEMS

Traditionally, usable query interfaces, e.g., keyword query interfaces, aim at improving users' satisfaction by optimizing some effectiveness metrics, e.g., $p@k$, for their input queries [11]. In our game-theoretic formalization, however, the goal of the DBMS should be to guide the interaction to a desired and stable state, i.e., equilibrium, in which, roughly speaking, both players do not have any motivation to change their strategies and they both get the maximum possible reward. There are three important questions regarding this game.

- What are the desired and undesired equilibria of the game? It is important to identify the non-optimal equilibria of the game as the interaction may stabilize in these equilibria.
- What are the reasonable assumptions on the behavior and the degree of rationality of the user? (Section 4)
- Given the answers to the previous two questions, what strategy adaptation mechanism(s) should the DBMS use to guide and converge the interaction to a desired equilibrium fast? At the first glance, it may seem that if the DBMS adapts a reasonable learning mechanism, the user's adaptation can only help further the DBMS to reach an optimal state as both players have identical interest. Nevertheless, in some collaborative two-player games in which both players adapt their strategies to improve their payoff, the learning may not converge to any (desired) equilibrium and cycle among several unstable states [30, 35]. More importantly, the DBMS should use an adaptation strategy that keeps users engaged [17]. In other words, the adaptation mechanism may not significantly reduce the payoff of the user for too many subsequent sessions. (Section 5)

We present some preliminary results on the last two aforementioned questions in the following sections. In this work we do not address the first one.

4 HOW DO USERS ADAPT?

Listed below are the equations for the reinforcement learning algorithms that we used. This section also contains details on the empirical analysis that we performed.

4.1 Bush and Mosteller's

Bush and Mosteller's model increases the probability that a user will choose a given query when searching for a specific intent by an amount proportional based on the reward of using that query and the current probability of using this query for the intent in the strategy [7]. It also decreases the probabilities of queries not used in a successful interaction. This method updates the probabilities of using queries for the intent e_i after an interaction using the following formulas.

$$U_{ij}(t+1) = \begin{cases} U_{ij}(t) + \alpha^{BM} \cdot (1 - U_{ij}(t)) & q_j = q(t) \wedge r \geq 0 \\ U_{ij}(t) - \beta^{BM} \cdot U_{ij}(t) & q_j = q(t) \wedge r < 0 \end{cases} \tag{2}$$

$$U_{ij}(t+1) = \begin{cases} U_{ij}(t) - \alpha^{BM} \cdot U_{ij}(t) & q_j \neq q(t) \wedge r \geq 0 \\ U_{ij}(t) + \beta^{BM} \cdot (1 - U_{ij}(t) & q_j \neq q(t) \wedge r < 0 \end{cases} \tag{3}$$

In the aforementioned formulas, $\alpha^{BM} \in [0, 1]$ and $\beta^{BM} \in [0, 1]$ are parameters of the model, $q(t)$ denotes the query picked by the

user at time t, and r is the reward of the interaction. If query q_j is equal to the query chosen by the user to represent intent e_i, then Equation 2 is used. For all other queries q_j for the intent e_i at time t, Equation 3 is used. The probabilities of using queries for intents other than e_i remains unchanged. Since the value of NDCG is always greater than zero, i.e., $r > 0$, the parameter β^{BM} is never used nor trained.

4.2 Cross's Model

Cross's model modifies the user's strategy similar to Bush and Mosteller's model [14]. However, it uses the amount of the received reward to update the user strategy. There are two parameters in this model, α^C and β^C that influence the rate of reinforcement

$$U_{ij}(t+1) = \begin{cases} U_{ij}(t) + R(r) \cdot (1 - U_{ij}(t)) & q_j = q(t) \\ U_{ij}(t) - R(r) \cdot U_{ij}(t) & q_j \neq q(t) \end{cases} \quad (4)$$

$$R(r) = \alpha^C \cdot r + \beta^C \quad (5)$$

In the above formulas, $\alpha^C \in [0, 1]$ and $\beta^C \in [0, 1]$ are the parameters used compute the adjusted reward $R(r)$ based on the value of actual reward r. The parameter β^C is a static increment of the adjusted reward. Similar to Bush and Mosteller's model, the aforementioned formulas are used to update the probabilities of using queries for the intent e_i in the current interaction. Other entries in the user's strategy are remained unchanged.

4.3 Roth and Erev's Model

Roth and Erev's model reinforces the probabilities directly from the reward value r that is received when the user enters query $q(t)$ [29]. Its most important difference with other models is that it explicitly accumulates all the rewards gained by using a query to express an intent. The primary differences of this model and the previous two models are that 1) it does not have any parameter to train and 2) there is not an explicit penalization of queries that are not used. $S_{ij}(t)$ in matrix $S(t)$ maintains the accumulated reward of using query q_j to express intent e_i over the course of the user and database system interactions up to round (time) t.

$$S_{ij}(t+1) = \begin{cases} S_{ij}(t) + r & q_j = q(t) \\ S_{ij}(t) & q_j \neq q(t) \end{cases} \quad (6)$$

$$U_{ij}(t+1) = \frac{S_{ij}(t+1)}{\sum_{j'}^{n} S_{ij'}(t+1)} \quad (7)$$

Roth and Erev's model increases the probability of using a query to express an intent based on the accumulated rewards of using that query over the long-term interaction of the user and data management system. It does not explicitly penalize other queries. Of course, because the user strategy U is row-stochastic, each query not used in a successful interaction, i.e., an interaction with $r > 0$, will be implicitly penalized as when the probability of a query increases, all others will decrease to keep U row-stochastic.

4.4 Roth and Erev's Modified Model

Roth and Erev's modified model is similar to the original Roth and Erev's model, but it has an additional parameter that determines to what extent the user takes in to account the outcomes of her past interactions with the system [15]. It is reasonable to assume that the user may forget the results of her much earlier interactions with the system. User's memory is imperfect which means that over time the strategy may change merely due to the forgetful nature of the user. This is accounted for by the *forget* parameter $\sigma \in [0, 1]$. Matrix $S(t)$ has the same role it has for the Roth and Erev's model.

$$S_{ij}(t+1) = (1 - \sigma) \cdot S_{ij}(t) + E(j, R(r)) \quad (8)$$

$$E(j, R(r)) = \begin{cases} R(r) \cdot (1 - \epsilon) & q_j = q(t) \\ R(r) \cdot (\epsilon) & q_j \neq q(t) \end{cases} \quad (9)$$

$$R(r) = r - r_{min} \quad (10)$$

$$U_{ij}(t+1) = \frac{S_{ij}(t+1)}{\sum_{j'}^{n} S_{ij'}(t+1)} \quad (11)$$

In the aforementioned formulas, $\epsilon \in [0, 1]$ is a parameter that weights the reward that the user receives, n is the maximum number of possible queries for a given intent e_i, and r_{min} is the minimum expected reward that the user wants to receive. The intuition behind this parameter is that the user often assumes some minimum amount of reward is guaranteed when she queries the database. The model uses this minimum amount to discount the received reward. We set r_{min} to 0 in our analysis, representing that there is no expected reward in an interaction. Therefor the model uses the total received reward to reinforce a strategy.

4.5 Empirical Results Methods

We detail how the empirical work is setup and the parameters used in this section.

4.5.1 Query Workload. We have used a subsample of a Yahoo! query log for our empirical study [34]. The Yahoo! query log consists of queries submitted to a Yahoo! search engine over a period of time in July 2010. Each record in the query log consists of a time stamp, user cookie, query submitted, the 10 results displayed to the user, and the positions of the user clicks. All the record logs are anonymized such that each time stamp, query, and returned result are saved as a unique identifier. Accompanying the query log is a set of *relevance judgment scores* for each query and result pair. The relevance judgment scores determine user satisfaction with that result. The score has the possible values of 0,1,2,3,4, with 0 meaning not relevant at all and 4 meaning the most relevant result. For our analysis we sorted the query log by the time stamp attribute to simulate the time line of the users interaction with the Yahoo! search engine. We determine the intent behind each query by using the relevance judgment scores for the results for each query. We consider the intent behind each query to be the set of results, i.e., URLs, with non-zero relevance scores.

4.5.2 Reinforcement Learning Methods. We have used and adapted six different reinforcement learning methods to model users' strategy in interaction with data systems. These models mainly vary based on 1) the degree by which the user considers past interactions when computing future strategies, 2) how they update the user strategy, and 3) the rate by which they update the user strategy. Some models assume that the user leverages outcomes of her past interactions when she updates her current strategy. Other models allow

the user to forget older interactions. These methods also differ in how they use the value of reward to update the user's strategy. Some reinforce a behavior, e.g., using a certain query to convey an intent, after a successful attempt by a fixed value independent of the amount of reward. Others use the value of received reward to reinforce a behavior. Finally, a model may have some discounting factors to control the rate by which a behavior is reinforced. In this study, we have used the value of NDCG for the returned results of a query as the reward in each interaction. Since NDCG models different levels of relevance, it provides a more exact estimate of the true reward in an interaction than other metrics that measure the quality of a ranked list, such as precision at k.

The six models we have adapted to model users' strategy in interaction with database systems are Bush and Mosteller's model [7], Cross's Model [14], Roth and Erev's Model [29], Roth and Erev's Modified Model [15], Win-Stay/Lose-Randomize [4], and Latest Reward. The last method simply using the most recent reward as the new probability for that query intent.

4.5.3 Comparing the Methods. Next, we compare the aforementioned models in terms of their use of past interaction and their update rules. Bush and Mosteller's, Cross's, and both Roth and Erev's models use information from the past to compute the future strategies. The Roth and Erev's models use the information about the past interactions more than others. Win-Stay/Lose-Randomize and Latest-Reward models do not rely as much as the first four methods on the outcomes of the previous interactions. Cross's, both Roth and Erev's, and the Latest-Reward models use the value of the reward that is received after entering a query to update the strategy. Bush and Mosteller's and Win-Stay/Lose-Randomize models change their strategies based on a fixed amount independent of the reward.

4.5.4 Training and Testing. Some models, e.g., Cross's model, have some parameters that need to be trained. We have used 5,000 records in the query workload and found the optimal values for those parameters using a grid search and the sum of squared errors. Each strategy has been initialized with an uniform distribution of probabilities, so that all queries are likely to be used for a given intent at the initial strategy. Once we had the parameters estimated for each model, we let each model to run over 300,000 and 500,000 records that follow the initial 5,000 records in the query log to compute a user strategy. We have evaluated the accuracy of the trained user strategies in predicting the future strategies of the users using the interaction records for 2,000 unique intents in the query log that follow the 300,000 records used in training. For each intent, we have found its first log record that immediately follows the records used in training and compared the predication of the strategy with the query actually used in this log record to express the intent. To compare the prediction accuracies of the strategies, we calculated the mean squared distance between what a given strategy predicted and what the user actually did.

4.6 Empirical Results

Tables 3 and 4 shows the results from the tests that we performed as well as the estimated parameters. A lower mean squared distance implies that the model more accurately represents the users' learning method. Roth and Erev's and Roth and Erev's modified models both perform the best out of all the tested models. Because both Roth and Erev models update the users strategies using the information of the previous strategies and interactions, users use their previous strategies and the outcomes of their previous interactions with the system when they pick a query to express their current intent. This result also indicates that the value of received reward should be considered when reinforcing a strategy. From our analysis it appears that users show a substantially intelligent behavior when adopting and modifying their strategies.

Method	Mean Squared Distance	Standard Deviation	Parameters
Bush and Mosteller	0.01252	0.0785	$\alpha^{BM} = 0.14$
Cross	0.01261	0.07875	$\alpha^{C} = 0.06$ $\beta^{C} = 0.11$
Roth and Erev	0.00993	0.05949	
Roth and Erev modified	0.00994	0.05954	$\sigma = 0$ $\epsilon = 0.18$
Win-Stay/Lose-Randomize	0.01747	0.06451	$\pi = 0.01$
Latest-Reward	0.12384	0.17118	

Table 3: The accuracies of learning algorithms - 300,000 queries

Method	Mean Squared Distance	Standard Deviation	Parameters
Bush and Mosteller	0.0112	0.07161	$\alpha^{BM} = 0.14$
Cross	0.01131	0.07207	$\alpha^{C} = 0.06$ $\beta^{C} = 0.11$
Roth and Erev	0.00993	0.07326	
Roth and Erev modified	0.00994	0.0733	$\sigma = 0$ $\epsilon = 0.18$
Win-Stay/Lose-Randomize	0.01752	0.06388	$\pi = 0.01$
Latest-Reward	0.15167	0.19614	

Table 4: The accuracies of learning algorithms - 500,000 queries

Bush and Mosteller's, Cross's, and Win-Stay/Lose-Randomize models perform worse than either of Roth and Erev's models. Bush and Mosteller's model has a relatively low value of α. Therefore, the rate of reinforcement is quite slow as the lower α is, the less a successful strategy is reinforced. With an α of 0 for example, there would be no reinforcement at all. Bush and Mosteller's model also does not consider the reward when reinforcing and therefor cannot reinforce queries that get effective results more than others that receive a smaller reward. Cross's model suffers from the same lack of reinforcement rate as Bush and Mosteller's but has an additional downfall. If the reward is extremely low, almost zero, the query will still be reinforced as β is a constant value independent of the reward. This means that queries with higher reward will be reinforced more,

but also means that queries with an extremely low reward will still be reinforced when they probably should be left alone.

Win-Stay/Lose-Randomize does not provide an accurate predication because it does not consider the entire history of strategies that the user has used. It also does not explore the space of possible queries to improve the effectiveness of the interaction. Hence, it seems that the users keep exploring possible queries to express an intent more effectively, although they may already know a query that conveys the intent quite successfully. Also, by only considering the previous reward, Win-Stay/Lose-Randomize cannot make robust adjustments and instead makes fixed changes in the model that are quite drastic. Finally, Latest-Reward performs the worst when compared to all models by an order of magnitude. This is because not only does this method have not memory like Win-Stay/Lose-Randomize, but it also reinforcing the strategy too drastically.

They leverage all most of their past interactions and their outcomes, i.e., have an effective long-term memory. This is specially interesting as the results of previous lab studies have indicated that mostly only proficient subjects rely on the accumulated rewards of the past interactions and use Roth and Erev's model for learning. Those studies show that non-proficient users tend to use models that do not leverage the information about the past interactions, such as Cross's model [8]. Also, the reward they receive directly impacts how they reinforce their strategy and will dictate what queries are used to represent intents in the future.

5 HOW SHOULD THE DBMS ADAPT?

This section looks at the third open problem we have listed in Section 3. To compare the effectiveness of our model with those currently employed, we conduct the following experiments. One of the popular models that is used during interaction with a database is the Multi Armed Bandit, which models whether to return a result or not as the pulling of an arm and ranking them based on some score. The multi armed bandit model does not consider the user as an intelligent agent that can possibly learn or adapt their strategies. A common and effective algorithm used in the multi armed bandit model is the UCB-1 algorithm [2, 27, 28, 33]. We construct the strategies of the user and our model using a collection of queries and URLs from the same Yahoo! dataset used earlier in Section 4. UCB-1 also uses this collection of URLs as its corpus of documents to rank and return to the user. The two algorithms are compared for some period of time using the effectiveness metric Mean Reciprocal Rank (MRR) [13].

5.1 UCB-1

We compare our model against the multi armed bandit model using the state of the art algorithm UCB-1 [2]. It has been used in many different studies and often out performs its competitors [27, 28, 28, 31]. The algorithm uses a mixture of exploitation and exploration combined into a single ranking function, shown in Equation 12.

$$Rank_t(q,e) = \frac{W_{q,e,t}}{\gamma_{q,e,t}} + \alpha\sqrt{\frac{2ln\ t}{\gamma_{q,e,t}}} \tag{12}$$

In Equation 12, t is the current time during the interaction. UCB-1 calculates a score for a document, or intent, e given a query sent by the user q. Exploitation in the algorithm comes from the first portion of the equation where γ is how many times an intent was shown to the user and W represents many times the user clicked on the returned intent. The second portion of the equation represents the exploration of the equation where α is an exploration weight value set between $[0, 1]$.

5.2 Roth and Erev in our model

In our model we use Roth and Erev's adaptation method, illustrated in Equations 13 and 14. After each round of the game, this method updates the DBMS's strategy according to the reward received in the previous rounds, i.e., $r(t)$.

$$S_{ji}(t+1) = \begin{cases} S_{ji}(t) + r(t) & l_i = e(t) \\ S_{ji}(t) & l_i \neq e(t) \end{cases} \tag{13}$$

$$D_{ji}(t+1) = \frac{S_{ji}(t+1)}{\sum_{i'}^{m} S_{ji'}(t+1)} \tag{14}$$

Roth and Erev's model reinforces the probabilities directly from the reward value $r(t)$ that is received when the user queries for intent $e(t)$. $S_{ji}(t)$ in matrix $S(t)$ maintains the accumulated reward of returning result l_i to satisfy intent e_i over the course of the user and database system interactions up to round (time) t. Of course, because the DBMS strategy D is row-stochastic, each result not returned in a successful interaction, i.e., an interaction with $r > 0$, will be implicitly penalized as when the probability of a result increases, all others will decrease to keep D row-stochastic.

5.3 Comparing UCB-1 and Roth and Erev

There are multiple scenarios that occur in real world interactions that UCB-1 does not consider. Here we list a few of them that we have tested and compared with our model that handles these specific scenarios.

5.3.1 Pooling of Intents. Often in practice the user does not have access or the knowledge to use a unique query for every intent they wish to express. This leads to a strategy on the user side that we refer to as *pooling*. Pooling is a user strategy that has more intents than queries, where there exists some *ambiguity* as to which query should be used for each intent on the user side and which intent should be returned on the database side, as a single query could be used to represent multiple intents from the user. UCB-1 ranks returned results based on how often the user clicks on them with some exploration, which may not provide the best answers in some ranked K returned results if many intents are conveyed using a single query. Our model uses probabilistic reinforcement algorithms, however, which can reflect the frequency that users query for a specific intent and return the desired result more often on average. Our earlier example illustrated in Tables 1 and 2 shows a user strategy where some amount of pooling takes place which leads to a degree of ambiguity.

5.3.2 Probabilistic vs. Deterministic. In our model we can employ some reinforcement learning algorithm that reflects its strategy as some set of probabilities on which results to return to the user. By using a probabilistic method of returning results we are able to quickly determine which results the user is not interested in for a received query. This probability can later adapt and change if the

user's interests change over time. UCB-1, however, uses a deterministic ranking algorithm, which means that if the algorithm learns that a specific intent is queried often with a certain query, then that result will be ranked at the top consistently. It has to instead rely on the exploration portion of the equation to explore other possible results if the user's interests change. This may be slow as the only time exploration takes place is after some extended period of interaction with that query. Deterministic methods have the problem of showing the most frequently queried result near the top at all times and may take some time to adapt to the user changing their strategy. This slows down learning and adaptation to the user, whom may change their strategy due to learning on their own part.

5.3.3 User Learning. Users learn when interacting with databases, either from outside factors outside of our control or due to the information received through interaction. It is through this learning that users may adapt their strategies to the interaction with the database system. Our model considers interaction as a collaborative game between two rational agents that are constantly learning about one another. UCB-1, however, does not consider whether the user changes their strategies or not and simply reacts to the user's current actions. After some extended period of interaction, the user may settle on a pooled strategy. If the user has learned a pooled strategy, then UCB-1 may struggle to satisfy the user's information needs as it will constantly be switching between which results to return for a received query. UCB-1 relies on getting 'lucky' with some amount of exploration to show these results to the user. Our model, however, considers the results to return with probabilities that reflect the frequency of how often they are queried.

5.4 Experimental Setup

We compared UCB-1 with our model using Roth and Erev's reinforcement learning algorithm. Our simulations were performed over a Yahoo! dataset of queries and URLs, the same dataset that was used in Section 4. We construct two identical user strategies, one to interact with each database learning algorithm. Each learning algorithm operates over the same set of possible intents to return. Roth and Erev's reinforcement model uses Reciprocal Rank as the satisfaction metric when reinforcing and UCB-1 uses a click model.

5.4.1 User Strategy Initialization. The user strategies are initialized such that they both start out identical. Using the Yahoo! query workload the user strategies start with some strategy already based on the attractiveness scores provided in the dataset. Attractiveness is a value between [0-1] calculated as in Algorithm 1 from the research in [9]. Using this attractiveness value, the user strategy is initialized such that the intents and query pairs that have a higher attractiveness start with a larger probability. The intent query pairs that do not have any score or there are not enough clicks in the query log to compute an attractiveness score have 0 probability.

One of the key differences between UCB-1 and our model is that our model takes into account the user learning, which happens in real world scenarios. The user strategy is updated using Roth and Erev's reinforcement learning model, which was determined to best represent how the user adapts from Section 4. The satisfaction metric for the reinforcement is Reciprocal Rank. Users will update their strategy after every interaction. Intents to be queried are picked at random with an even distribution.

Another difference between UCB-1 and our model is that often in real life users tend to pool their intents to a single query. UCB-1 has difficulty learning this kind of behavior. To simulate this type of real world scenario, we construct the user strategies with some degree of *ambiguity*. Ambiguity in the user strategy indicates how much pooling takes place. We say that a user strategy has a high degree of ambiguity if the user represents many intents with a single query. Of course, the user may not represent these intents with the same query all the time. For example, if the user strategy consists of 50 intents, then a strategy having a high degree of ambiguity may have all queries share 50% of the intents.

5.4.2 Database Strategy Initialization. The database algorithms and their weights are initialized the same for both algorithms. Roth and Erev in our model is initialized with a purely random strategy, with the same number of queries as the user strategy and a much larger number of intents then the user is looking for. UCB-1 starts with all of the values at 1, with the same set of possible documents to rank as the Roth and Erev strategy in our model. The exploration rate, α is initialized at 0.5, to allow for sufficient amount of exploration during the simulation.

5.5 Results

First we compare the MRR that each strategy receives over time. The user strategy has a high degree of ambiguity with 33 intents and 2 queries. Each query is used for at least 12 of the other intents query. Thus, each query will be sent for the same intent for at least 15 of the intents. The results from this comparison are illustrated in Figure 1.

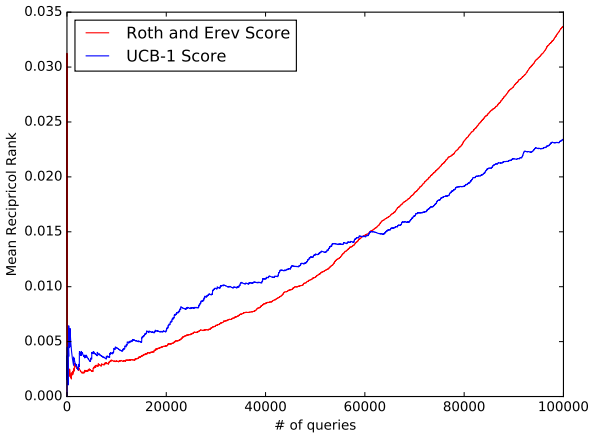


Figure 1: MRR for 100,000 interactions

Second, we look at the average Reciprocal Rank over all the intents. The user strategy again has a high degree of ambiguity with 30 intents and 2 queries, where each query needs to share at least 15 intents with the other query. These results are illustrated in Figure 2.

Both of these results show that the Roth and Erev reinforcement algorithm when used with our model takes into account the user

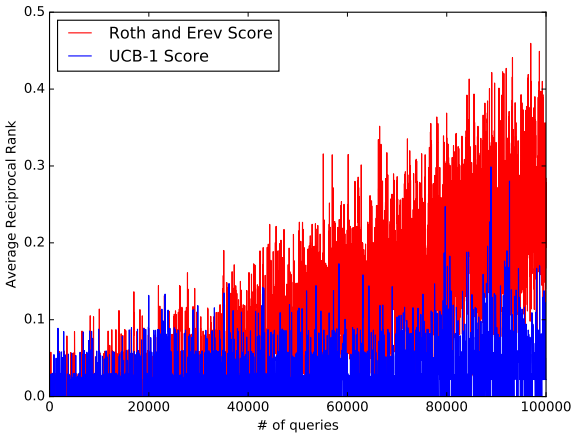


Figure 2: Average Reciprocal Rank over intents for 100,000 interactions

learning and the ambiguity that takes place in real world situations. Looking at Figure 1 we can see that the average reward is increasing faster. When running the simulation for a much longer period of time, we actually notice that our model converges on a strategy that has a consistently higher MRR value than UCB-1. Figure 2 sheds more light on why this is as we see that the reciprocal rank is consistently at a rather low value. This is due to the fact that UCB-1 has a deterministic strategy and cannot return versatile results to the user accounting for all the intents that they are querying for with a limited number of queries.

Another interesting fact is that UCB-1 performs better in the beginning of the simulation. This is because not all of the intents have been queried much yet and it is still able to satisfy the user for the small amount of intents it has learned. As the interaction continues and more intents are used near the same frequency we see that UCB-1 cannot satisfy the user as often as our model.

6 RELATED WORK

Researchers have proposed querying and exploration interfaces over structured and semi-structured databases that help users to express their information needs and find reasonably accurate results [1, 5, 6, 11, 16, 18, 19, 21, 23]. We extend this body of work by considering users as active and potentially rational agents whose decisions and strategies impact the effectiveness of database exploration. We also go beyond effectively answering a single query and aim at improving the effectiveness of overall interaction of users and database systems.

Researchers in other scientific disciplines, such as economics and sociology, have used signaling games to formally model and explore communications between multiple rational agents [12, 22]. Avestani et al. have used signaling games to create a shared lexicon between multiple autonomous systems [3]. We, however, focus on modeling users' information needs and emergence of mutual language between users and database systems. In particular, database systems and users may update their information about the interaction in different time scales. Researchers have modeled the decision of a user to continue or stop searching a certain topic over a collection of documents using stochastic games [24].

We, however, seek a deeper understanding of information need representations and the emergence of a common query language between the user and the database system during their interactions. Further, we investigate the interactions that may span over multiple sessions. Of course, a relatively precise mutual understanding between the user and the database system also improves the effectiveness of ad-hoc and single-session querying.

Concurrent to our effort, Zhang et al.have proposed a model to optimize the navigational search interfaces over document retrieval systems such that a user finds her desired document(s) by performing the fewest possible actions, e.g., clicking on links [36]. Our goal, however, is to model and improve the mutual understanding of intents and their articulations between the user and the database system. Since data querying and exploration are performed over series of interactions between two potentially rational agents, if one agent unilaterally optimizes its reward without any regard to the strategies of the other agent, the collaboration may not lead to a desired outcome for any of the agents. Thus, instead of unilateral optimization, our goal is to find a desired equilibrium for the game by considering possible strategies and strategy adaptation mechanisms for both users and database systems.

7 CONCLUSION

Most users are not able to express precisely their intents in the form of database queries so the database systems understands them. Thus, users' queries do not often reflect their true information needs. The users and database system may be able to establish a mutual language of representing information needs through interaction. We described our framework that models the interaction between the user and the database system as a collaborative game of two potentially rational agents in which the players would like reach a common method of representing information needs. We empirically investigated the exploration behavior of users using a real-world query workload. Our results show that users typically use some degree of rationality when interacting with the database system, remembering previous interactions and adapting their strategy to them. We also compared our model versus another popular model used, the multi armed bandit with the UCB-1 algorithm. Our results show that correctly modeling the interaction by considering the user and database as both rational agents improves user satisfaction.

8 ACKNOWLEDGEMENTS

Arash Termehchy is supported in part by the National Science Foundation under grant IIS-1423238. Liang Huang is supported in part by the National Science Foundation under grant IIS-1656051, DARPA N66001-17-2-4030 (XAI), and an HP Gift.

REFERENCES

[1] Azza Abouzied, Dana Angluin, Christos H. Papadimitriou, Joseph M. Hellerstein, and Avi Silberschatz. 2013. Learning and verifying quantified boolean queries by example. In *PODS*.

[2] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2-3 (2002), 235–256.

[3] Paolo Avesani and Marco Cova. 2005. Shared lexicon for distributed annotations on the Web. In *WWW*.

[4] J. A. Barrett and K. Zollman. 2008. The Role of Forgetting in the Evolution and Learning of Language. *Journal of Experimental and Theoretical Artificial Intelligence* 21, 4 (2008), 293–309.

[5] Thomas Beckers and others. 2010. Report on INEX 2009. *SIGIR Forum* 44, 1 (2010).

[6] Angela Bonifati, Radu Ciucanu, and Slawomir Staworko. 2015. Learning Join Queries from User Examples. *TODS* 40, 4 (2015).

[7] Robert R Bush and Frederick Mosteller. 1953. A stochastic model with applications to learning. *The Annals of Mathematical Statistics* (1953), 559–585.

[8] Yonghua Cen, Liren Gan, and Chen Bai. 2013. Reinforcement Learning in Information Searching. *Information Research: An International Electronic Journal* 18, 1 (2013), n1.

[9] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web*. ACM, 1–10.

[10] Surajit Chaudhuri, Gautam Das, Vagelis Hristidis, and Gerhard Weikum. 2006. Probabilistic Information Retrieval Approach for Ranking of Database Query Results. *TODS* 31, 3 (2006).

[11] Yi Chen, Wei Wang, Ziyang Liu, and Xuemin Lin. 2009. Keyword Search on Structured and Semi-structured Data. In *SIGMOD*.

[12] I. Cho and D. Kreps. 1987. Signaling games and stable equilibria. *Quarterly Journal of Economics* 102 (1987).

[13] Nick Craswell. 2009. Mean reciprocal rank. In *Encyclopedia of Database Systems*. Springer, 1703–1703.

[14] John G Cross. 1973. A stochastic learning model of economic behavior. *The Quarterly Journal of Economics* 87, 2 (1973), 239–266.

[15] Ido Erev and Alvin E Roth. 1995. *On the Need for Low Rationality, Gognitive Game Theory: Reinforcement Learning in Experimental Games with Unique, Mixed Strategy Equilibria*.

[16] Norbert Fuhr and Thomas Rolleke. 1997. A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems. *TOIS* 15 (1997).

[17] Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2013. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval* 16, 1 (2013).

[18] Vagelis Hristidis, Luis Gravano, and Yannis Papakonstantinou. Efficient IR-Style Keyword Search over Relational Databases. In *VLDB 2003*.

[19] Stratos Idreos, Olga Papaemmanouil, and Surajit Chaudhuri. 2015. Overview of Data Exploration Techniques. In *SIGMOD*.

[20] H. V. Jagadish, Adriane Chapman, Aaron Elkiss, Magesh Jayapandian, Yunyao Li, Arnab Nandi, and Cong Yu. 2007. Making Database Systems Usable. In *SIGMOD*.

[21] Nodira Khoussainova, YongChul Kwon, Magdalena Balazinska, and Dan Suciu. 2010. SnipSuggest: Context-aware Autocompletion for SQL. *PVLDB* 4, 1 (2010).

[22] David Lewis. 1969. *Convention*. Cambridge: Harvard University Press.

[23] Hao Li, Chee-Yong Chan, and David Maier. 2015. Query From Examples: An Iterative, Data-Driven Approach to Query Construction. *PVLDB* 8, 13 (2015).

[24] Jiyun Luo, Sicong Zhang, and Hui Yang. 2014. Win-Win Search: Dual-Agent Stochastic Game in Session Search. In *SIGIR*.

[25] Christopher Manning, Prabhakar Raghavan, and Hinrich Schutze. 2008. *An Introduction to Information Retrieval*. Cambridge University Press.

[26] Ben McCamish, Arash Termehchy, Behrouz Touri, and Eduardo Cotilla Sanchez. 2016. A Signaling Game Approach to Databases Querying. In *AMW*.

[27] Taesup Moon, Wei Chu, Lihong Li, Zhaohui Zheng, and Yi Chang. 2012. An online learning framework for refining recency search results with user click feedback. *ACM Transactions on Information Systems (TOIS)* 30, 4 (2012), 20.

[28] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*. ACM, 784–791.

[29] Alvin E Roth and Ido Erev. 1995. Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term. *Games and economic behavior* 8, 1 (1995), 164–212.

[30] Lloyd S Shapley and others. 1964. Some topics in two-person games. *Advances in game theory* 52, 1-29 (1964), 1–2.

[31] Marc Sloan and Jun Wang. 2013. Iterative expectation for multi period information retrieval. *arXiv preprint arXiv:1303.5250* (2013).

[32] Aleksandr Vorobev, Damien Lefortier, Gleb Gusev, and Pavel Serdyukov. 2015. Gathering Additional Feedback on Search Results by Multi-Armed Bandits with Respect to Production Ranking. In *WWW*.

[33] Aleksandr Vorobev, Damien Lefortier, Gleb Gusev, and Pavel Serdyukov. 2015. Gathering additional feedback on search results by multi-armed bandits with respect to production ranking. In *Proceedings of the 24th international conference on World wide web*. International World Wide Web Conferences Steering Committee, 1177–1187.

[34] Yahoo! 2011. Yahoo! webscope dataset anonymized Yahoo! search logs with relevance judgments version 1.0. http://labs.yahoo.com/Academic_Relations. (2011). [Online; accessed 5-January-2017].

[35] H Peyton Young. 2004. *Strategic learning and its limits*. OUP Oxford.

[36] Yinan Zhang and Chengxiang Zhai. 2015. Information Retrieval as Card Playing: A Formal Model for Optimizing Interactive Retrieval Interface. In *SIGIR*.

Data Sketches for Disaggregated Subset Sum Estimation

Daniel Ting
Tableau Software
1162 N 34th St
Seattle, Washington 98103
dting@tableau.com

ABSTRACT

We introduce and study a new data sketch for processing massive datasets. It addresses two common problems: 1) computing a sum given arbitrary filter conditions and 2) identifying the frequent items or heavy hitters in a data set. For the former, the sketch provides unbiased estimates with state of the art accuracy. It is specifically designed to handle the challenging scenario when the data is disaggregated. In this case, there is a per unit metric of interest that can only be computed as an expensive pre-aggregation of the raw, disaggregated data. For example, the metric of interest may be total clicks per user while the raw data is a click stream containing multiple rows per user. By creating a small, in-memory sketch of a massive dataset, a consumer may interactively query the data nearly instantaneously while still being able to slice or filter by almost any dimension. The sketch is suitable for use in a wide range of applications including computing historical click through rates for ad prediction, reporting user metrics from user event streams, and measuring network traffic for IP flows.

We informally prove and empirically show that the sketch has good properties for both the disaggregated subset sum estimation and frequent item problems on i.i.d. data. It not only picks out the frequent items but also gives strongly consistent estimates for the proportion of each frequent item. For subset sum estimation, it asymptotically draws a probability proportional to size sample that is optimal for estimating the sum over the data. Empirically, despite the disadvantage of operating on disaggregated data, our method matches or bests priority sampling, a state of the art method on pre-aggregated data. When compared to naive uniform sampling, it performs orders of magnitude better on skewed data. We also propose extensions to the sketch that allow it to be used in combining multiple data sets, in distributed systems, and for time decayed aggregation.

This paper is a work in progress.

CCS CONCEPTS

•Mathematics of computing →Probabilistic algorithms; •Theory of computation →Sketching and sampling;

KDD 2017 Workshop on Interactive Data Exploration and Analytics (IDEA'17), Halifax, Nova Scotia, Canada

DOI:

KEYWORDS

Data sketching, subset sum estimation, counting, frequent item, heavy hitters, sampling

1 INTRODUCTION

When analyzing massive data sets, even simple operations such as computing a sum or mean is costly and time consuming. These simple operations are frequently performed both by people investigating the data interactively and asking a series of questions about it as well as in automated systems which must monitor or collect a multitude of statistics.

Data sketching algorithms enable the information in these massive datasets to be efficiently processed, stored, and queried. This allows them to be applied, for example, in real-time systems, both for ingesting massive data streams or for interactive analysis.

In order to achieve this efficiency, sketches are designed to only answer a specific class of question, and there is typically error in the answer. In other words, it is a form of lossy compression on the original data where one must choose what to lose in the original data. A good sketch makes the most efficient use of the data so that the errors are minimized while having the flexibility to answer a broad range of questions of interest. Some sketches, such as HyperLogLog, are constrained to answer very specific questions with extremely little memory. On the other end of the spectrum, sampling based methods such as priority sampling [13] or coordinated sampling [3], [7] are able to answer almost any question on the original data but at the cost of far more space to achieve the same approximation error.

We introduce a sketch, unbiased space saving, that simultaneously addresses two common data analysis problems: the disaggregated subset sum problem and the frequent item problem. This makes the sketch more flexible than previous sketches that address one problem or the other. Furthermore, it is efficient as it provides state of the art performance on the disaggregated subset sum problem and has a stronger consistency guarantee for frequent item count estimation than previous results for i.i.d. streams.

The disaggregated subset sum estimation is a more challenging variant of the subset sum estimation problem [13], the extremely common problem of computing a sum or mean over a dataset with arbitrary filtering or grouping conditions. In the disaggregated subset sum problem [5], [17] the data is "disaggregated" so that a per item metric of interest is split across multiple rows. For example in an ad click stream, the data may arrive as a stream of single clicks that are identified with each ad while the metric of interest is the total number of clicks per ad. The frequent item problem is the problem of identifying the heavy hitters or most frequent items in a dataset. Several sketches exist for both these individual problems.

In particular, the sample and hold methods of [5], [15], [17] address the disaggregated subset sum estimation problem. Frequent item sketches include the space saving sketch [23], Misra-Gries sketch [24], and lossy counting sketch. [22].

Our sketch is an extension of the space saving frequent item sketch, and as such, has stronger frequent item estimation properties than sample and hold. In particular, unlike sample and hold, theorem 6.1 gives both that a frequent item will eventually be included in the sketch with probability 1, and that the proportion of times it appears will be consistently estimated for i.i.d. streams. In contrast to frequent item sketches which are biased, our unbiased space saving sketch gives unbiased estimates for any subset sum, including subsets containing no frequent items.

Our contributions are in three parts: 1) the development of the unbiased space saving sketch, 2) the generalizations obtained from understanding the properties of the sketch and the mechanisms by which it works, and 3) the theoretical and empirical results establishing the correctness and efficiency of the sketch for answering the problems of interest. In particular, the generalizations allow multiple sketches to be merged so that information from multiple data sets may be combined as well as allowing it to be applied in distributed system. Other generalizations include the ability to handle signed and real-valued updates as well as time-decayed aggregation. We empirically test the sketch on both synthetic and real ad prediction data. Surprisingly, we find that it even outperforms, priority sampling, a method that requires pre-aggregated data.

This paper is structured as follows. First, we describe the disaggregated subset sum problem, some of its applications, and related sketching problems. We then introduce our sketch, unbiased space saving, as a small but significant modification of the space-saving sketch. We examine its relation to other frequent item sketches, and show that they differ in a "reduction" operation. This is used to show that any unbiased reduction operation yields an unbiased sketch for the disaggregated subset sum estimation problem. The theoretical properties of the sketch are then examined. We conjecture its consistency for the frequent item problem and for drawing a probability proportional to size sample and provide informal arguments that we believe can be made precise. Finally, we present experiments using real and synthetic data.

2 DISAGGREGATED SUBSET SUM PROBLEM

Many data analysis problems consist of a simple aggregation over some filtering and group by conditions.

```
SELECT sum(metric), dimensions
FROM table
WHERE filters
GROUP BY dimensions
```

This problem has several variations that depend on what is known about the possible queries and about the data before the sketch is constructed. For problems in which there is no group by clause and the set of possible filter conditions are known before the sketch is constructed, counting sketches such as the CountMin sketch [9] and AMS sketch [2] are appropriate. When the filters and group by dimensions are not known and arbitrary, the problem is the subset sum estimation problem. Sampling methods such as priority sampling [13] can be used to solve it. These work by exploiting a measure of importance for each row and sampling important rows with high probability. For example, when computing a sum, the rows containing large values contribute more to the sum.

The disaggregated subset sum estimation problem is a more difficult variant where there is little to no information about row importance and only a small amount of information about the queries. For example, many user metrics, such as number of clicks, are computed as aggregations over some event stream where each event has the same weight 1 and hence, the same importance. Filters and group by conditions can be arbitrary except for a small restriction that one cannot query at a granularity finer than a specified unit of analysis. In the click example, the finest granularity may be at the user level. One is allowed to query over arbitrary subsets of users but cannot query a subset of a single user's clicks. The data is "disaggregated" since the relevant per unit metric is split across multiple rows. We will refer to something at the smallest unit of analysis as an *item* to distinguish it from one row in the data.

Since pre-aggregating to compute per unit metrics does not reduce the amount of relevant information, it follows that the best accuracy one can achieve is to first pre-aggregate and then apply a sketch for subset sum estimation. This operation, however, is extremely expensive, especially as the number of units is often large. Examples of units include users and ad id pairs for ad click prediction, source and destination IP pairs for IP flow metrics, and distinct search queries or terms. Each of these have trillions or more possible units.

Several sketches based on sampling have been proposed that address the disaggregated subset sum problem. These include the bottom-k sketch [6] which samples items uniformly at random, the class of "NetFlow" sketches [14], and the sample and hold sketches [5], [15], [17]. Of these, the Sample-and-Hold sketches are clearly the best as they use strictly more information than the other methods to construct samples and maintain aggregate statistics. We describe them in more depth in section 4.4.

The unbiased space-saving sketch we propose throws away even less information than previous sketches. Surprisingly, this allows it to match the accuracy of priority sampling, a nearly optimal subset sum estimation algorithm [29], which uses pre-aggregated data. In some cases, our sketch achieves better accuracy despite being computed on disaggregated data.

2.1 Applications

The disaggregated subset sum problem has many applications. These include machine learning and ad prediction [28], analyzing network data [14], [5], detecting distributed denial of service attacks [27], database query optimization and join size estimation, as well as analyzing web users' activity logs or other business intelligence applications.

For example, in ad prediction the historical click-through rate and other historical data are among the most powerful features for future ad clicks [18]. Since there is no historical data for newly created ads, one may use historical click or impression data for previous ads with similar attributes such as the same advertiser or product category [26]. In join size estimation, it allows the sketch to estimate the size under the arbitrary filtering conditions that a user might impose.

Algorithm 1 Space-Saving algorithms

- Maintain an m list of $(item, count)$ pairs initialized to have count 0.
- For each new row in the stream, let x_{new} be its item and increment the corresponding counter if the item is in the list. Otherwise, find the pair $(x_{min}, \hat{N}_{min})$ with the smallest count. Increment the counter and replace the item label with x_{new} with probability p.
- For the original space-saving algorithm $p = 1$. For unbiased count estimates $p = 1/(\hat{N}_{min} + 1)$.

It also can be naturally applied to hierarchical aggregation problems. For network traffic data, IP addresses are arranged hierarchically. A network administrator may both be interested in individual nodes that receive or generate an excess of traffic or aggregated traffic statistics on a subnet. Several sketches have been developed to exploit hierarchical aggregations including [8], [25], and [30]. Since the disaggregated subset sum sketches handles arbitrary group by conditions, it can compute the next level in a hierarchy.

2.2 Frequent item problem

The frequent item or heavy hitter problem is related to the disaggregated subset sum problem. Our sketch is an extension of space saving, [23], a frequent item sketch. Like the disaggregated subset sum problem, frequent item sketches are computed with respect to a unit of analysis that requires a partial aggregation of the data. Only the most frequent items are of interest though. Most frequent item sketches are deterministic and have deterministic guarantees on both the identification of frequent items and the error in the counts of individual items. However, since counts in frequent item sketches are biased, further aggregation on the sketch can lead to large errors as bias accumulates as shown in section 6.2.

Our work is based on a frequent item sketch, but applies randomization to achieve unbiased count estimates. This allows them to be used in subset sum queries. Furthermore, it maintains good frequent item estimation properties as shown in section 6.

3 UNBIASED SPACE-SAVING

Our sketch is based on the space-saving sketch [23] used in frequent item estimation. For simplicity, we consider the case where the metric of interest is the count for each item. The space saving sketch works by maintaining a list of m bins labeled by distinct items. A new row with item i increments i's counter if it is in the sketch. Otherwise, the smallest bin is incremented, and its label is changed to i. Our sketch introduces one small modification. If $\hat{N}_{min}$ is the count for the smallest bin, then only change the label with probability $1/(\hat{N}_{min} + 1)$. This change provably yields unbiased counts as shown in theorem 3.1 More formally, the algorithms are given in algorithm 1.

Theorem 3.1. *For any item x, the randomized Space-Saving algorithm in figure 1 gives an unbiased estimate of the count of x.*

Proof. Let $\hat{N}_x(t)$ denote the estimate for the count of x at time t and $\hat{N}_{min}(t)$ be the count in the smallest bin. We show that the expected increment to $N_x(t)$ is 1 if x is the next item and 0 otherwise. Suppose x is the next item. If it is in the list of counters, then it is incremented by exactly 1. Otherwise, it incremented by $\hat{N}_{min}(t)+1$ with probability $1/(\hat{N}_{min}(t) + 1)$ for an expected increment of 1. Now suppose x is not the next item. The estimated count $\hat{N}_x(t)$ can only be modified if x is the label for the smallest count. It is incremented with probability $\hat{N}_x(t)/(\hat{N}_x(t)+1)$. Otherwise $\hat{N}_x(t+1)$ is updated to 0. This gives the update an expected increment of $\mathbb{E}\hat{N}_x(t+1) - \hat{N}_x(t) = (\hat{N}_x(t)+1)\hat{N}_x(t)/(\hat{N}_x(t)+1) - \hat{N}_x(t) = 0$ when the new item is not x. □

We note that although given any fixed item x, the estimate of its count is unbiased, each stored pair often contains an overestimate of the item's count. This occurs since any item with a positive count will receive a downward biased estimate of 0 when it is not in the sketch. Thus, conditional on an item appearing in the sketch, the count must be biased upwards to balance out the bias.

4 RELATED SKETCHES AND FURTHER GENERALIZATIONS

Although our primary goal is to demonstrate the usefulness of the unbiased space-saving sketch, we also try to understand the mechanisms by which it works and find extensions and generalizations that can be gleaned from that understanding.

In particular, we examine the relationship between unbiased space saving and existing deterministic frequent items sketches. We show that existing frequent item sketches all share the same structure as an exact increment of the count followed by a size reduction. This size reduction is implemented as an adaptive sequential thresholding operation which biases the counts. Our modification replaces the thresholding operation with a subsampling operation. This observation allows us to extend the sketch. This includes endowing it with an unbiased merge operation that can be used to combine datasets or in distributed computing environments.

The sampling design in the reduction step may also be chosen to give the sketch different properties. For example, time-decayed sampling methods may be used to weight recently occurring items more heavily. If multiple metrics are being tracked, the multi-objective sampling [4] may be used.

4.1 Probability proportional to size sampling

Our key observation in generalizing unbiased space saving is that the choice of label is a sampling operation. In particular, this sampling operation chooses the item with probability proportional to its size. We briefly review probability proportional to size sampling and priority sampling as well as the Horvitz-Thompson estimator which allows one to unbias the sum estimate from any biased sampling scheme.

For unequal probability samples, an unbiased estimator for the sum over the true population $\{x_i\}$ is given by the Horvitz-Thomson estimator $\hat{S} = \sum_i \frac{x_i Z_i}{\pi_i}$ where Z_i denotes whether x_i is in the sample and $\pi_i = P(Z_i = 1)$ is the inclusion probability. When only linear statistics of the sampled items are computed, the item values may be updated $x_i^{new} = x_i/\pi_i$.

When drawing a sample of fixed size, it is trivial to see that an optimal set of inclusion probabilities is given by $\pi_i \propto x_i$ when this is possible. In other words, it generates a probability proportional

to size (PPS) sample. In this case, each term in the sum is constant, so that the estimator is exact and has zero variance. When the data is skewed, drawing a probability proportional size sample may be impossible for sample sizes greater than 1. For example, given values 1, 1, and 10, any scheme to draw 2 items with probabilities exactly proportional to size has inclusion probabilities bounded by $1/10$, $1/10$, and 1. The expected sample size is at most $12/10 < 2$. In this case, one often chooses inclusion probabilities $\pi_i = \min\{\alpha x_i, 1\}$ for some constant α. The inclusion probabilities are proportional to the size if the size is not too large and 1 otherwise.

Many algorithms exist for generating PPS samples. In particular, the splitting procedure of [12] provides a class of methods to generate a fixed size PPS sample with the desired inclusion probabilities. Another method which approximately generates a PPS sample is priority sampling. Instead of exact inclusion probabilities which are typically intractable to compute, priority sampling generates a set of pseudo-inclusion probabilities.

4.2 Misra-Gries and frequent item sketches

The Misra-Gries sketch [24], [11], [20] is a frequent item sketch and is isomorphic to the space saving sketch [1]. The only difference is that it decrements all counters rather than incrementing the smallest bin when processing an item that is not in the sketch. Thus, the count in the smallest bin for the space-saving sketch is equal to the total number of decrements in the Misra-Gries sketch. Given estimates $\hat{\mathbf{N}}$ from a space-saving sketch, the corresponding estimated item counts for the Misra-Gries sketch are $\hat{N}_i^{MG} = (\hat{N}_i - \hat{N}_{min})_+$ where $\hat{N}_{min}$ is the count for the smallest bin and the operation $(x)_+$ truncates negative values to be 0. In other words, the Misra-Gries estimate is the same as space saving estimate soft thresholded by $\hat{N}_{min}$. Equivalently, the space-saving estimates are obtained by adding back the total number of decrements $\hat{N}_{min}$ to any nonzero counter in the Misra-Gries sketch.

The sketch has a deterministic error guarantee. When the total number of items is N and the total estimated count for items in the sketch is $\hat{n}_{tot} = \sum_i \hat{n}_i$ then the error for any item's count is at most $(n - \hat{n}_{tot})/(m+1)$.

Other frequent item sketches include the deterministic lossy counting and randomized sticky sampling sketches [22]. We describe only lossy counting as sticky sampling has both worse practical performance and weaker guarantees than other sketches.

A simplified version of Lossy counting applies the same decrement reduction as the Misra-Gries sketch but decrements occur at a fixed schedule rather than one which depends on the data itself. To count items with frequency $> N/m$, all counters are decremented after every m rows. Lossy counting does not provide a guarantee that the number of counters can be bounded by m. In the worst case, the size can grow to $m \log(N/m)$ counters. Similar to the isomorphism between the Misra-Gries and Space-saving sketches, the original Lossy counting algorithm is recovered by adding the number of decrements back to any nonzero counter.

4.3 Reduction operations

Existing deterministic frequent item sketches differ in only the operation to reduce the number of nonzero counters. They all have the form described in algorithm 2 and have reduction operations that can be expressed as a thresholding operation. Although it is isomorphic to the Misra-Gries sketch, space-saving's reduction operation can also be described as collapsing the two smallest bins by adding the larger bin's count to the smaller one's.

Algorithm 2 General frequent item sketching

- Maintain current estimates of counts $\hat{\mathbf{N}}(t)$
- Increment $\hat{N}'_{x_{t+1}}(t+1) \leftarrow \hat{N}_{x_{t+1}}(t) + 1$.
- $\hat{\mathbf{N}}(t+1) \leftarrow ReduceBins(\hat{\mathbf{N}}'(t+1), t+1)$

Modifying the reduction operation provides the sketch with different properties. We highlight several uses for alternative reduction operations.

The reduction operation for unbiased space saving can be seen as a PPS sample on the two smallest bins. A natural generalization is to consider a PPS sample on all the bins. We highlight three benefits of such a scheme. First, items can be added with arbitrary counts or weights. Second, the sketch size can be reduced by multiple bins in one step. Third, there is less quadratic variation added by one sampling step, so error can be reduced. The first two benefits are obvious consequences of the generalization. To see the third, consider when a new row contains an item not in the sketch, and let $\mathcal{J}$ be the set of bins equal to the size of the smallest bin $\hat{N}_{min}$. When using the thresholded PPS inclusion probabilities from section 4.1, the resulting PPS sample has inclusion probability $\alpha = |\mathcal{J}|/(1 + |\mathcal{J}|\hat{N}_{min})$ for the new row's item and $\alpha\hat{N}_{min}$ for bins in $\mathcal{J}$. Other bins have inclusion probability 1. After sampling, the Horvitz-Thompson adjusted counts are $1/|\mathcal{J}| + \hat{N}_{min}$. Unbiased space saving is thus a further randomization to convert the real valued increment $1/|\mathcal{J}|$ over $|\mathcal{J}|$ bins to an integer update on a single bin. Since unbiased space saving adds an additional randomization step, the PPS sample has smaller variance. The downside of this procedure, however, is that it requires real valued counters that require more space per bin.

Changing the sampling procedure can also provide other desirable behaviors. Applying forward decay sampling [10] allows one to obtain estimates that weight recent items more heavily. Other possible operations include adaptively varying the sketch size in order to only remove items with small estimated frequency.

Furthermore, the reduction step does not need to be limited strictly to subsampling. Theorem 4.1 gives that any unbiased reduction operation yields unbiased estimates. This generalization allows us to analyze Sample-and-Hold sketches.

Theorem 4.1. *Any reduction operation where the expected post-reduction estimates are equal to the pre-reduction estimates yields an unbiased sketch for the disaggregated subset estimation problem. More formally, if* $\mathbb{E}(\hat{\mathbf{N}}(t)|S_{pre}(t)) = \hat{\mathbf{N}}_{pre}(t)$ *where* $S_{pre}(t), \hat{\mathbf{N}}_{pre}(t)$ *are the sketch and estimated counts before reduction at time step t and* $\hat{\mathbf{N}}(t)$ *is the post reduction estimate, then* $\hat{\mathbf{N}}(t)$ *is an unbiased estimator.*

Proof. Since $\hat{\mathbf{N}}_{pre}(t) = \hat{\mathbf{N}}_{post}(t-1) + (\mathbf{n}(t) - \mathbf{n}(t-1))$, it follows that $\hat{\mathbf{N}}(t) - \mathbf{n}(t)$ is a martingale with respect to the filtration adapted to $S(t)$. Thus, $\mathbb{E}\hat{\mathbf{N}}(t) = \mathbf{n}(t)$, and the sketch gives unbiased estimates for the disaggregated subset sum problem. □

We also note that reduction operations can be biased. The merge operation on the Misra-Gries sketch given by [1] can be seen as

performing a soft-thresholding by the size of the $(m+1)^{th}$ counter. This also allows it to reduce the size of the sketch by more than 1 bin at a time. It can be modified to handle deletions and arbitrary numeric aggregations by making the thresholding operation two-sided so that negative values are shrunk toward 0 as well. In this case, we do not provide a theoretical analysis of the properties.

Modifying the reduction operation also yields interesting applications outside of counting. In particular, a reduction operation on matrices can yield accurate low rank decompositions [21], [16].

4.4 Sample and Hold

To the author's best knowledge, the current state of the art sketches designed to answer disaggregated subset sum estimation problems are the family of sample and hold sketches [17], [15], [5]. These methods can also be described with a randomized reduction operation.

For adaptive sample and hold [5], the sketch maintains an auxiliary variable p which represents the sampling rate. Each point in the stream is assigned a $U_i \sim Uniform(0,1)$ random variable, and the items in the sketch are those with $U_i < p$. If an item remains in the sketch starting from time t_0, then the counter stores the number of times it appears in the stream after the initial time. Every time the sketch becomes too large, the sampling rate is decreased so that under the new rate p', one item is no longer in the sketch.

It can be shown that unbiased estimates can be obtained by keeping a counter value the same with probability p'/p and decrementing the counter by a random $Geometric(p')$ random variable otherwise. If a counter becomes negative, then it is set to 0 and dropped. Adding back the mean $(1-p')/p'$ of the *Geometric* random variable to the nonzero counters gives an unbiased estimator. Effectively, the sketch replaces the first time an item enters the sketch with the expected $Geometric(p')$ number of tries before it successfully enters the sketch plus it adds the actual count after the item enters the sketch. Using the memoryless property of *Geometric* random variables, it is easy to show that the sketch satisfies the conditions of theorem 4.1. It is also clear that one update step adds more error and unbiased space saving as it potentially adds $Geometric(p')$ noise with variance $(1-p')/p'^2$ to every bin. Furthermore, the eliminated bin may not even be the smallest bin. Since p' is the sampling rate, it is expected to be close to 0. By contrast, unbiased space saving has bounded increments of 1 for bins other than the smallest bin, and the only bin that can be removed is the current smallest bin.

The discrepancy is especially prominent for frequent items. A frequent item in an i.i.d. stream for unbiased space saving enters the sketch almost immediately, and the count for the item is nearly exact as shown in theorem 6.1. For adaptive sample and hold, the first $n_i(1-p')$ occurrences of item i are expected to be discarded and replaced with a high variance $Geometric(p')$ random variable. Since p' is typically small in order to keep the number of counters low, most of the information about the count is discarded.

Another sketch, step sample-and-hold, avoids the problem by maintaining counts for each "step" when the sampling rate changes. However, this is more costly both from storage perspective as well as a computational one. For each item in the sketch, computing the expected count takes time quadratic in the number of steps J_i in which the step's counter for the item is nonzero, and storage is linear in J_i.

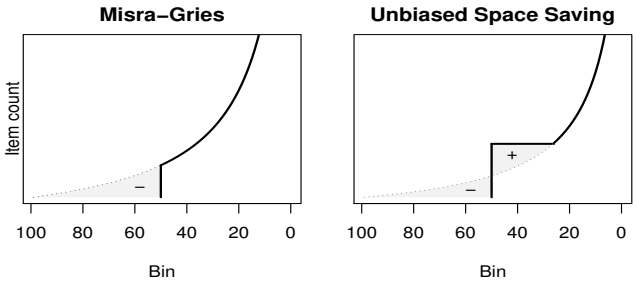


Figure 1: In a merge operation, the Misra-Gries sketch simply removes mass from the extra bins with small count. Unbiased space saving moves the mass from infrequent items to moderately frequent items. It loses the ability to pick those items as frequent items in order to provide unbiased estimates for the counts in the tail.

4.5 Merging and Distributed counting

The generalized reduction operations allow for merge operations on the sketches. Merge operations and mergeable sketches [1] are important since they allow a collection of sketches, each which answers questions about the specific data it was constructed on, to be combined to answer a question over all the data. For example, a set of frequent item sketches that give trending news for each country can be combined to give trending news for Europe as well as a multitude of other possible combinations. Another common scenario arises when sketches are aggregated across time. Sketches for clicks may be computed per day, but the final machine learning feature may combine the last 7 days. Furthermore, merges allow for simple distributed computation. In a map-reduce framework, each mapper can quickly compute a sketch, and only a set of small sketches needs to be sent over the network to perform an aggregation at the reducer.

As noted in the previous section, the Misra-Gries sketch has a simple merge operation which preserves its deterministic error guarantee. It simply soft thresholds by the $(m+1)^{th}$ largest counter so that at most m nonzero counters are left. Previously, no merge operation existed for space-saving except to first convert it to a Misra-Gries sketch. The conversion of soft-thresholds to approximate hard thresholds yields a merge operation for space-saving sketches. However, this does not preserve the total item count. Theorem 4.1 shows that by replacing the pairwise randomization with priority sampling or some other sampling procedure still allows one to obtain an unbiased space saving merge that can preserve the expected count in the sketch rather than biasing it downward.

The trade-off required for such an unbiased merge operation is that the sketch may detect fewer of the top items by frequency than the biased Misra-Gries merge. Rather than truncating and preserving more of the "head" of the distribution, it must move mass from the tail closer to the head. This is illustrated in figure 1.

5 PROPERTIES

We study the properties of the space-saving sketches here. These include asymptotic properties, empirical properties, behavior in pathological cases, and costs in time and space. In particular, we conjecture and provide an informal proof that when the data is i.i.d., the sketch eventually includes frequent items with probability 1. Other items are sampled with probability proportional to their size. This is also borne out in the experimental results where the observed inclusion probabilities match the theoretical ones and in estimation error where unbiased space saving matches or even exceeds the accuracy of priority sampling. In pathological cases, we demonstrate that deterministic space-saving fails at the subset estimation problem. Furthermore, these pathological sequences can arise naturally. Any sequence where items' arrival rates change significantly over time forms a pathological sequence.

6 ASYMPTOTIC CONSISTENCY

We conjecture and provide a proof sketch that shows that the data sketch contains all frequent items eventually on i.i.d. stream. Thus it does no worse than deterministic space-saving asymptotically for frequent item estimation on such streams while having much better aggregation behavior on pathological streams.

Assume that items are drawn from a possibly infinite, discrete distribution with probabilities $p_1 \geq p_2 \geq \ldots$ and, without loss of generality, assume they are labeled by their index into this sequence of probabilities. Let m be the number of bins and t be the number of items processed by the sketch. We will also refer to t as time. Let $\mathcal{I}(t)$ be the set of items that are in the sketch at time t and $Z_i(t) = 1(i \in \mathcal{I}(t))$ indicate if the label i is in the sketch at time t. Define an absolutely frequent item to be an item drawn with probability $> 1/m$ where m is the number of bins in the sketch. Our precise conjecture and the proof sketch of its veracity are as follows.

Conjecture 6.1. *If $p_1 > 1/m$, then as the number of items $t \to \infty$, $Z_1(t) = 1$ eventually.*

Proof. The goal is to show that label 1 will eventually become "sticky." This requires (1) that some bin gets the label 1 and (2) that the bin "escapes" from being the smallest bin before it is relabeled, and (3) that it remains that way so that the label cannot be changed. For (1), it is easy to see that there are $\Omega(\log t)$ times that the smallest bin will flip to label 1. Denote the size of the smallest bin $N_{min}(t)$ and the estimated count for label 1 as $N_1(t)$. Trivially $N_{min}(t) < t/m \leq tp_1$.For the remaining two requirements, we compare the size of the bin with label 1 to t/m. For (2), we note that the smallest bin grows at a rate of at least $\alpha = \sum_{j>m} p_j$. Hence, $t/m - N_{min}(t) \leq (1-\alpha)t/m + o_p(1)$. A bin growing at rate p_1 will take approximately $t' - t = (t/m - N_{min}(t))(p_1 - 1/m)^{-1}$ steps to catch up to t'/m. The probability that the label is overwritten during this time is bounded above by $(t' - t)/t'$ which simplifies to a constant that does not depend on the time t. Thus, every time the label flips to 1, there is at least a constant nonzero probability of "escape." Every time a bin escapes with label 1, $N_1(t) - t/m$ forms an asymmetric random walk starting at or slightly above 0. The probability of never returning to 0 and, hence, never being relabeled is at least some positive constant c. Since there is some constant positive probability a bin will become sticky after being relabeled 1, and there are infinitely many time it will acquire that label, it eventually must become sticky. □

6.1 Approximate PPS Sample

An interesting consequence of the above conjecture is that bins fall into two classes for i.i.d. streams. The first class are bins with labels that become "sticky." These are asymptotically "pure" bins where the proportion of items with the current label goes to 1. The second are bins where the label keeps changing because the rate of a labeled bin is never greater than the rate for the smallest bin, and hence, the size must remain close to the minimum bin size.

Each time an item is added to the smallest bin, the label for that bin is a probability proportional to size sample of size 1 from the items previously added to that bin. This informal argument leads to our second conjecture.

Conjecture 6.2. *The items in the sketch converge in distribution to a PPS sample on i.i.d. streams where either a label is sampled with probability 1 or with probability $\propto n_i$.*

We note, however, that the resulting PPS sample has limitations not present in PPS samples on pre-aggregated data. For pre-aggregated data, one has both the original value x_i and the Horvitz-Thompson adjusted value x_i/π_i where π_i is the inclusion probability. This allows the sample to compute non-linear statistics such as the population variance which uses the second moment estimator $\sum_i x_i^2 Z_i/\pi_i$. With the PPS samples from disaggregated subset sum sketching, only the adjusted values are observed.

6.2 Pathological sequences

Deterministic space-saving has remarkably low error when estimating the counts of frequent items [8]. In general, if the order data arrives is uniformly random or if the data stream consists of i.i.d. data, one expects the deterministic space-saving algorithm to share similar unbiasedness properties as the randomized version as in both cases the label for a bin can be treated roughly as a uniform random choice out of the items in that bin.

Pathological cases arise when an item's arrival rate changes over time rather than staying constant. Consider a sketch with 2 bins. For a sequence of c 1's, c 2's, a single 3, and a single 4, the deterministic space saving algorithm will always return 3 and 4, each with count $c + 1$. By contrast, randomized space-saving will return 1 and 2 with probability $(1 - 1/c)^2 \approx 1$ when c is large. Note that in this case, the count for each frequent item is slightly below the threshold that guarantees inclusion in the sketch, $c < n/2$. This example illustrates the behavior for the deterministic algorithm. When an item is not in the "frequent item head" of the distribution then the bins that represent the tail pick the labels of the most recent items without regard to the frequency of older items.

We note that such a pathological sequence can easily occur naturally. For instance, partially sorted data can naturally lead to such pathological sequences. Periodic bursts of an item followed by periods in which its frequency drops below the threshold of guaranteed inclusion are another example. Another pathological case for disaggregated subset sum problems arises when every item is distinct. The deterministic sketch consists of the last m

items rather than a random sample, and no meaningful subset sum estimate can be derived for deterministic space saving.

6.3 Running time and space complexity

The update operation is identical to the deterministic space saving update except that it changes the label of a bin less frequently. Thus, each update can be performed in $O(1)$ time [23] when the stream summary data structure is used. In this case the space usage is $O(m)$ in the number of bins.

7 EXPERIMENTS

We perform experiments with both simulations and real ad prediction data. For synthetic data, we draw the count for each item using a Weibull distribution that is discretized to integer values. That is $n_i \sim Round(Weibull(k, \alpha))$ for item i. The discretized Weibull distribution is a generalization of the geometric distribution that allows us to adjust the tail of the distribution to be more heavy tailed. We choose it over the Zipfian or other truly heavy tailed distributions as few real data distributions have infinite variance. Furthermore, we expect our methods to perform even better with greater data skew as shown in figure 2. For more easily reproducible behavior we applied the inverse cdf method $n_i = F^{-1}(U_i)$ so that the U_i are on a regular grid of 1000 values rather than $Uniform(0, 1)$ random variables. In each case, we draw at least 10,000 samples to estimate the root mean squared error. To simulate a variety of possible filtering conditions, we draw random subsets of 100 items. As expected, subsets which mostly pick items in the tail of the distribution have estimates with higher relative root mean squared error. Note that an algorithm with α times the root mean squared error of a baseline algorithm often requires α^2 times the space as the variance, not the standard deviation, scales linearly with size.

For real data, we use a Criteo ad click prediction dataset [1]. This dataset provides a sample of 45 million ad impressions. Each sample includes the outcome of whether or not the ad was clicked as well as multiple integer valued and categorical features. We pick a subset of 9 of these features. There are over 500 million possible tuples on these features and many more possible filtering conditions. The impressions without a click are sampled at a lower rate than those with clicks.

The Criteo dataset provides a natural application of the disaggregated subset sum problem. Historical clicks are a powerful feature in click prediction [26], [19]. While the smallest unit of analysis is the *ad* or the (*user*, *ad*) pair, the data is in a disaggregated form with one row per impression. Furthermore, since there may not be enough data for a particular ad, the relevant click prediction feature may be the historical click through rate for the advertiser or some other higher level aggregation. Past work using sketches to estimate these historical counts [28] include the CountMin counting sketch as well as the Lossy Counting frequent item sketch.

Although we do not directly compare against sample and hold methods, we note that figure 2 in [5] shows that sample and hold performs worse than priority sampling.

This added variability in the threshold and the relatively small sketch sizes for the simulations on i.i.d. streams may explain why

[1] http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset/

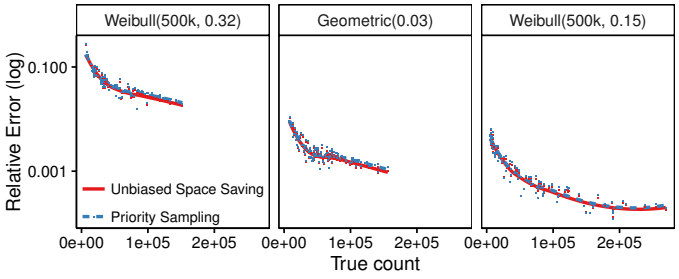


Figure 2: The empirical performance of Unbiased Space-Saving matches priority sampling, which required an expensive pre-aggregation step. The sketch accuracy improves when the skew is higher and when more and larger bins are contained in the subset. The number of bins is 200.

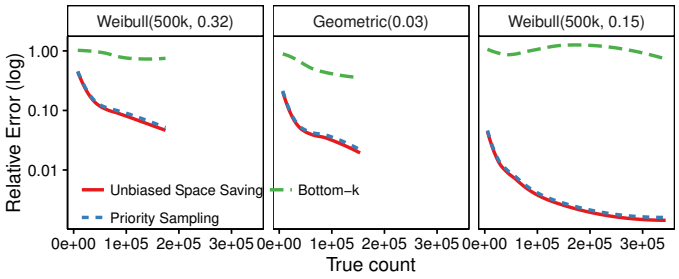


Figure 3: Unbiased space saving performs orders of magnitude better than uniform sampling of items (Bottom-k). The plots show the smoothed plot of relative error versus the true count. With 100 bins, the error is higher than with 200 bins given in figure 2 but the curve is qualitatively similar.

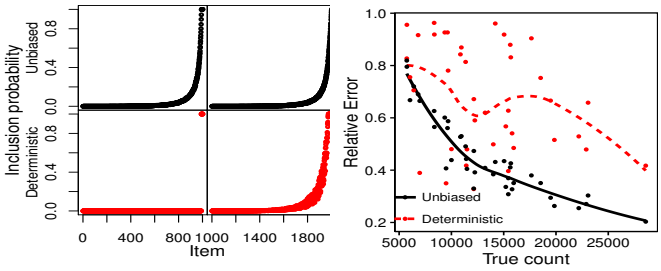


Figure 4: Deterministic Space-Saving performs poorly on pathological sequences. Left: Items 1 to 1000 only appear in the first half of the stream. The inclusion probabilities for a pathological sequence still behave like a PPS sample for unbiased space saving, but only the frequent items in the first half are sampled under deterministic space saving. Right: As a result, deterministic space saving is highly inaccurate when querying items in the first half of the stream.

unbiased space saving performs even better than what could be considered close to a "gold standard" on pre-aggregated data.

8 CONCLUSION

We have introduced a novel sketch, unbiased space saving, that answers both the disaggregated subset sum and frequent item problems. Surprisingly, for the disaggregated subset sum problem, the sketch can outperform even methods that run on pre-aggregated data. We prove that asymptotically, it can answer the frequent item problem for i.i.d. sequences with probability 1 eventually. Furthermore, it gives stronger probabilistic consistency guarantees on the accuracy of the count than the deterministic space saving sketch. For infrequent items, we prove that the items selected for the sketch are sampled approximately according to a PPS sample.

We study its behavior and connections to other data sketches. In particular, we identify the primary difference between many of the frequent item sketches is a slightly different operation to reduce the number of bins. We use that understanding to provide multiple generalizations to the sketch which allow it to be applied in distributed settings, handle weight decay over time, and adaptively change its size over time. This also allows us to compare unbiased space to the family of sample and hold sketches that are also designed to answer the disaggregated subset sum problem.

REFERENCES

[1] P. K. Agarwal, G. Cormode, Z. Huang, Jeff M Phillips, Z. Wei, and K. Yi. 2013. Mergeable summaries. *ACM Transactions on Database Systems* 38, 4 (2013), 26.

[2] N. Alon, Y. Matias, and M. Szegedy. 1999. The Space Complexity of Approximating the Frequency Moments. *J. Comput. System Sci.* 58, 1 (1999), 137–147.

[3] K. R.W. Brewer, L.J. Early, and S.F. Joyce. 1972. Selecting several samples from a single population. *Australian & New Zealand Journal of Statistics* 14, 3 (1972), 231–239.

[4] Edith Cohen. 2015. Multi-objective weighted sampling. In *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*. IEEE, 13–18.

[5] E. Cohen, N. Duffield, H. Kaplan, C. Lund, and M. Thorup. 2007. Sketching unaggregated data streams for subpopulation-size queries. In *PODS*. ACM.

[6] Edith Cohen and Haim Kaplan. 2007. Summarizing data using bottom-k sketches. In *PODC*.

[7] E. Cohen and H. Kaplan. 2013. What You Can Do with Coordinated Samples. In *RANDOM*.

[8] Graham Cormode and Marios Hadjieleftheriou. 2008. Finding frequent items in data streams. *VLDB* 1, 2 (2008), 1530–1541.

[9] G. Cormode and S. Muthukrishnan. 2005. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms* 55, 1 (2005), 58–75.

[10] G. Cormode, V. Shkapenyuk, D. Srivastava, and B. Xu. 2009. Forward decay: A practical time decay model for streaming systems. In *ICDE*. IEEE, 138–149.

[11] E. D. Demaine, A. López-Ortiz, and J. I. Munro. 2002. Frequency estimation of internet packet streams with limited space. In *European Symposium on Algorithms*. 348–360.

[12] J.C. Deville and Y. Tillé. 1998. Unequal probability sampling without replacement through a splitting method. *Biometrika* 85, 1 (1998), 89–101.

[13] Nick Duffield, Carsten Lund, and Mikkel Thorup. 2007. Priority sampling for estimation of arbitrary subset sums. *Journal of the ACM (JACM)* 54, 6 (2007), 32.

[14] C. Estan, K. Keys, D. Moore, and G. Varghese. 2004. Building a better NetFlow. In *ACM SIGCOMM Computer Communication Review*, Vol. 34. ACM, 245–256.

[15] Cristian Estan and George Varghese. 2002. *New directions in traffic measurement and accounting*. Vol. 32. ACM.

[16] Mina Ghashami, Edo Liberty, and Jeff M Phillips. Efficient Frequent Directions Algorithm for Sparse Matrices. (????).

[17] P. B. Gibbons and Y. Matias. 1998. New sampling-based summary statistics for improving approximate query answers. In *ACM SIGMOD Record*, Vol. 27. ACM, 331–342.

[18] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, and others. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the International Workshop on Data Mining for Online Advertising*. ACM, 1–9.

[19] D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. 2010. Improving ad relevance in sponsored search. In *WSDM*. ACM, 361–370.

[20] R. M. Karp, S. Shenker, and C. H Papadimitriou. 2003. A simple algorithm for finding frequent elements in streams and bags. *ACM Transactions on Database Systems (TODS)* 28, 1 (2003), 51–55.

[21] Edo Liberty. 2013. Simple and deterministic matrix sketching. In *KDD*. ACM.

[22] G. Manku and R. Motwani. 2002. Approximate frequency counts over data streams. In *VLDB*.

[23] A. Metwally, D. Agrawal, and A. El Abbadi. 2005. Efficient computation of frequent and top-k elements in data streams. In *ICDT*.

[24] J. Misra and D. Gries. 1982. Finding repeated elements. *Science of computer programming* 2, 2 (1982), 143–152.

[25] M. Mitzenmacher, T. Steinke, and J. Thaler. 2012. Hierarchical heavy hitters with the space saving algorithm. In *Meeting on Algorithm Engineering & Expermiments*. 160–174.

[26] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *WWW*. ACM, 521–530.

[27] V. Sekar, N. Duffield, O. Spatscheck, J. E. van der Merwe, and H. Zhang. LADS: Large-scale Automated DDoS Detection System.

[28] A. Shrivastava, A. C. König, and M. Bilenko. 2016. Time Adaptive Sketches (Ada-Sketches) for Summarizing Data Streams. *SIGMOD* (2016).

[29] Mario Szegedy. 2006. The DLT priority sampling is essentially optimal. In *STOC*. ACM, 150–158.

[30] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund. 2004. Online identification of hierarchical heavy hitters: algorithms, evaluation, and applications. In *Internet Measurement Conference (IMC)*. ACM, 101–114.

8