Minimizing User Effort in Large Scale Example-driven Data Exploration

by

Xiaoyu Ge

Bachelor of Science, University at Buffalo, 2012

Master of Science, University of Pittsburgh, 2019

Submitted to the Graduate Faculty of

the Dietrich School of Arts and Sciences in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2021

UNIVERSITY OF PITTSBURGH

DIETRICH SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

Xiaoyu Ge

It was defended on

July 22, 2021

and approved by

Prof. Panos K. Chrysanthis, PhD, Chair

Prof. Adriana Kovashka, PhD, Examiner

Prof. Alexandros Labrinidis, PhD, Examiner

Prof. Mohamed A. Sharaf, PhD, External Examiner

Dissertation Director: Prof. Panos K. Chrysanthis, PhD, Chair

Minimizing User Effort in Large Scale Example-driven Data Exploration

Xiaoyu Ge, PhD

University of Pittsburgh, 2021

Data Exploration is a key ingredient in a widely diverse set of discovery-oriented applications, including scientific computing, financial analysis, and evidence-based medicine. It refers to a series of exploratory tasks that aim to extract useful pieces of knowledge from data, and its challenge is to do so without requiring the user to specify with precision what information is being searched for. The goal of assisting users in constructing their exploratory queries effortlessly, which effectively reveals interesting data objects, has led to the development of a variety of intelligent semi-automatic approaches. Among such approaches, *Example-driven Exploration* is rapidly becoming an attractive choice for exploratory query formulation since it attempts to minimize the amount of prior knowledge required from the user to form an accurate exploratory query.

In particular, this dissertation focuses on *interactive* Example-driven Exploration, which steers the user towards discovering all data objects relevant to the users' exploration based on their feedback on a small set of examples. Interactive Example-driven Exploration is especially beneficial for non-expert users, as it enables them to circumvent query languages by assigning relevancy to examples as a proxy for the intended exploratory analysis. However, existing interactive Example-driven Exploration systems fall short of supporting the need to perform complex explorations over large, unstructured high-dimensional data. To overcome these challenges, we have developed new methods of data reduction, example selection, data indexing, and result refinement that support practical, interactive data exploration.

The novelty of our approach is anchored on leveraging *active learning* and *query optimization* techniques that strike a balance between maximizing accuracy and minimizing user effort in providing feedback while enabling interactive performance for exploration tasks with arbitrary, large-sized datasets. Furthermore, it extends the exploration beyond the structured data by supporting a variety of high-dimensional unstructured data and enables the refinement of results when

the exploration task is associated with too many relevant data objects that could be overwhelming to the user. To affirm the effectiveness of our proposed models, techniques, and algorithms, we implemented multiple prototype systems and evaluated them using real datasets. Some of them were also used in domain-specific analytics tools.

Table of Contents

Preface . . . xv

1.0 Introduction . . . 1

1.1 Limitations of Existing Work . . . 3

1.2 Hypothesis & Objective . . . 5

1.3 Our Contributions . . . 6

1.4 Thesis Outline . . . 8

2.0 Background . . . 9

2.1 Fully Automated vs. Semi-Automated Data Exploration . . . 9

2.2 Example-driven Exploration . . . 11

2.2.1 Typical Workflow of Example-driven Exploration . . . 13

2.2.2 Existing Example-driven Exploration Approaches . . . 14

2.3 Active Learning . . . 15

2.3.1 Offline Active Learning . . . 15

2.3.2 Stream-Based Active Learning . . . 18

2.4 Other Interactive Exploration Approaches . . . 19

2.5 Interactive Search for Unstructured Data . . . 20

2.6 Result Refinement Techniques . . . 22

2.6.1 Relevance Ranking Techniques . . . 22

2.6.2 Diversity Techniques . . . 23

2.6.3 Multi-Criteria Objective Optimization . . . 25

2.6.4 Data Summarization . . . 27

2.7 Summary . . . 27

3.0 Effortless Exploring of Structured Data . . . 29

3.1 Introduction . . . 29

3.2 The REQUEST Framework 30
3.2.1 Data Reduction 32
3.2.1.1 Data-Driven Sampling 33
3.2.1.2 User-Driven Pruning 33
3.2.2 Query Selection 34
3.2.2.1 Uncertainty Sampling 34
3.2.2.2 Naive Bayes Classifier 34
3.2.2.3 Committee of Naive Bayes Classifiers 36
3.2.2.4 Randomized Uncertainty 36
3.2.2.5 Randomized Accept/Reject Uncertainty (RARU) 37
3.3 The REQUEST Schemes 37
3.3.1 The None+RARU Scheme 38
3.3.2 The None+VotedRARU Scheme 38
3.3.3 The UDP+RARU Scheme 38
3.3.4 The UDP+VotedRARU Scheme 39
3.4 Experimental Evaluation 39
3.4.1 Experiment Setup 39
3.4.2 Experimental Results 41
3.5 Summary 48
4.0 Towards Scalable Interactive Data Exploration 50
4.1 Introduction 50
4.2 Uncertainty Estimation Index 52
4.2.1 UEI Components 53
4.2.2 UEI in Action 56
4.2.3 Time Complexity of UEI 58
4.3 Experimental Evaluation 58
4.3.1 Experiment Setup 59

4.3.2 Experiment Results 60
4.4 Applicability of UEI 63
4.5 Summary 64
5.0 Exploring Big Unstructured Data 65
5.1 Introduction 65
5.2 The Exploration Navigator (ExNav) 66
5.2.1 The ExNav Solution 66
5.2.2 Data Embedding 68
5.2.3 Exploration Space Pruning 69
5.2.4 Query Strategy 70
5.2.4.1 Uncertainty Sampling 70
5.2.4.2 Gradient Boosting Trees 71
5.2.4.3 Randomized Accept/Reject Uncertainty (RARU) 72
5.3 Experimental Evaluation 73
5.3.1 Experiment Setup 73
5.3.2 Experimental Results 75
5.4 Summary 83
6.0 Exploration Result Refinement 85
6.1 Introduction 85
6.2 Problem Challenges 87
6.3 Problem Formulation 91
6.3.1 Background 91
6.3.1.1 Relevance 91
6.3.1.2 Diversity 92
6.3.1.3 Coverage 94
6.3.2 Diversified Top-k (DT-k) Problem 95
6.3.2.1 Problem Formulation 95

6.3.2.2 Problem Complexity 95
6.3.2.3 Secondary Objective 96
6.4 PrefDiv Algorithms 96
6.4.1 Naive Solution 96
6.4.2 Preferential Diversity 97
6.4.3 Relevance Proportionality 99
6.4.4 Optimize Diversity Constraints for Coverage 102
6.5 Experimental Evaluation 105
6.5.1 Experimental Testbed 106
6.5.2 Evaluation Metrics 108
6.5.3 Experimental Results 109
6.5.3.1 Normalized Relevance 109
6.5.3.2 Coverage 109
6.5.3.3 Execution Time 113
6.5.3.4 Parameter A of PrefDiv 114
6.5.3.5 Relevancy vs. Diversity 114
6.5.3.6 Additional Observations 115
6.6 Impacts of PrefDiv 117
6.7 Summary 117
7.0 Conclusions 119
7.1 Our Contributions 119
7.2 Open Questions 122
7.3 Broader Impact 124
Bibliography 126

List of Tables

1 Experimental Parameters of REQUEST Schemes 40

2 Experimental Parameters of UEI 60

3 Experimental Parameters of ExNav 74

4 ExNav Runtime with Different Query Strategies 81

5 Top-5 and Bottom-5 Tuples With Respect to the Cost 88

6 Top-5 Tuples Based on Cost That Are Diversified With Respect to Attributes "Food Type" and "Score". 89

7 List of Notations Used in Chapter 6 92

List of Figures

1 Illustration of the Manual Exploration Process. 2

2 Illustration of the Taxonomy of Data Exploration Techniques. Each Left Node Refers to One Example Data Exploration System. Our Contributions Are Marked With Squares. 10

3 2-D Database With 3 Relevant Regions. 12

4 Illustration of Uncertainty Sampling. 16

5 Accuracy, 2D, 1 Small Region 41

6 Accuracy, 2D, 1 Medium Region 41

7 Accuracy, 2D, 3 Small Regions 42

8 Accuracy, 2D, 3 Medium Regions 42

9 Accuracy, 2D, 5 Small Regions 42

10 Accuracy, 2D, 5 Medium Regions 42

11 Accuracy, 2D, 1 Large Region 42

12 Runtime, 2D, 1 Small Region 42

13 Accuracy, 2D, 3 Large Regions 43

14 Runtime, 2D, 1 Medium Region 43

15 Accuracy, 2D, 5 Large Regions 43

16 Runtime, 2D, 1 Large Region 43

17 UDP+VotedRARU with Increasing Regions Sizes (1 Region) 45

18 UDP+VotedRARU with Increasing Regions Numbers (Large Region) 45

19 Accuracy,1 Small Region, Small Dataset 46

20 Runtime, 1 Small Region, Small Dataset 46

21 Accuracy, 3 Small Regions, Small Dataset 46

22 Runtime, 3 Small Regions, Small Dataset 46

23 Accuracy, 5 Small Regions, Small Dataset 47

24 Runtime, 5 Small Regions, Small Dataset 47
25 1 Small Region, F-Measurement of UDP+VotedRARU, Different Dimensions 47
26 1 Small Region, Runtime (sec) of UDP+VotedRARU, Different Dimensions 47
27 1 Small Region, F-Measurement of UDP+VotedRARU, Different Dataset Sizes 48
28 1 Small Region, Runtime (sec) of UDP+VotedRARU, Different Dataset Sizes 48
29 Illustrates UEI With a 2D Data Space, Each Grid Represents a Subspace, the Dot in the Center of Each Grid Represents a Symbolic Point p. With Chunks Stored As Separated Files on the Secondary Storage. 53
30 Before Storing the Data, UEI Vertically Decompose the Data Into an Inverted Index Form, and Then Store Them in Separate Chunks. 54
31 UEI vs. MySQL Accuracy (Small Target Region). 61
32 UEI vs. MySQL Accuracy (Medium Target Region). 61
33 UEI vs. MySQL Accuracy (Large Target Region). 61
34 UEI vs. MySQL Response Time. 61
35 Accuracy 1 Large Region (Text) 76
36 Accuracy 1 Small Region (Text) 76
37 Accuracy 1 Large Region (Image) 76
38 Accuracy 1 Small Region (Image) 76
39 Accuracy 1 Large Region (Graph) 77
40 Accuracy 1 Small Region (Graph) 77
41 Accuracy 1 Medium Region (Text) 77
42 Accuracy 2 Medium Region (Text) 77
43 Accuracy 1 Medium Region (Image) 78
44 Accuracy 2 Medium Region (Image) 78
45 Accuracy 1 Mediums Region (Graph) 78
46 Accuracy 2 Mediums Region (Graph) 78
47 Accuracy 3 Medium Region (Text) 79

48 Accuracy 3 Medium Regions (Text) . . . 79
49 Accuracy 3 Medium Region (Image) . . . 79
50 Accuracy 3 Medium Regions (Image) . . . 79
51 Accuracy 3 Mediums Region (Graph) . . . 80
52 Accuracy 3 Medium Regions (Graph) . . . 80
53 Accuracy 1 Medium Region (Text) . . . 82
54 Accuracy 3 Medium Regions (Text) . . . 82
55 Accuracy 1 Medium Region (Image) . . . 82
56 Accuracy 3 Medium Regions (Image) . . . 82
57 Accuracy 1 Medium Region (Graph) . . . 83
58 Accuracy 3 Medium Regions (Graph) . . . 83
59 Single Vertex v_1 With 100% Coverage. . . . 90
60 A Set of Vertices $\{v_4, v_5\}$ With 60% Coverage. . . . 90
61 Illustration of Similarity and Dissimilarity. . . . 93
62 Illustration of the Optimal Radius, When $k = 2$. . . . 103
63 Normalized Intensive Value of Cameras . . . 110
64 Normalized Intensive Value of NYC . . . 110
65 Coverage of Cameras. . . . 110
66 Coverage of NYC . . . 110
67 Execution Time of Cameras . . . 111
68 Execution Time of NYC . . . 111
69 Normalized Intensive Value of Foursquare's San Francisco Data . . . 111
70 Result Set Coverage, With the Optimal Radius and K = 30. . . . 111
71 Coverage of Foursquare's San Francisco Data . . . 112
72 Result Set Normalized Relevance, With the Optimal Radius and K = 30. . . . 112
73 Execution Time of Foursquare's San Francisco Data . . . 112
74 Result Set Execution Time, With the Optimal Radius and K = 30. . . . 112

75 Relevance VS. Diversity (NYC). 115
76 Relevance VS. Diversity (SF). 116
77 Relevance VS. Diversity (Cameras). 116
78 Illustration of the Taxonomy of Data Exploration Techniques and Our Contributions. Each Left Node Refers to One Example Data Exploration System. Our Contributions Are Marked With Squares. 120

Preface

I would like to express my most sincere thanks to everybody who made my journey to Ph.D. possible and joyful.

First of all, I want to express my deepest gratitude to my advisor, Prof. Panos K. Chrysanthis, who has always supported me both academically and personally. Not only I've learned from Prof.Chrysanthis the way to conduct high-quality research, but he has also demonstrated to me the real qualities of teacher excellence. He also encouraged and guided me through many difficult moments during my journey of graduate study.

I would like to thank my thesis committee, Prof. Alexandros Labrinidis, Prof. Adriana Kovashka, and Prof. Mohamed A. Sharaf, for their invaluable feedback and discussions on my research. Parts of this thesis are the results of collaboration with colleagues from the University of Pittsburgh. In particular, I want to thank Prof. Shi-Kuo Chang for his valuable support and advice and our fellow doctorate student, Xiaozhong Zhang, who has worked closely with me on several research projects. I also would like to thank all the other former and current members of the Advanced Data Management Technologies Laboratory (ADMT) lab: Anatoli Shein, Angen Zheng, Brian Nixon, Constantinos Costa, Cory Thoma, Nick R. Katsipoulakis, Rakan A. Alseghayer, Roxana Gheorghiu, and Vineet Raghu.

I am also grateful to have many precious friends in Pittsburgh who made my life enjoyable. In particular, I want to mention my best friend Yanbing Xue, who was always willing to help and gave me countless support and advice.

I would also like to thank NIH for their funding support of this thesis work, as this thesis is partially supported by NIH award U01HL137159 and reflects only my opinions.

Finally, I am very grateful to my wife Yiwen Fan, my daughter Yixuan (Evelyn) Ge, my mother Hua Li, my grandparents, and my cousins for their unlimited love, encouragement, and support.

Thank you, everyone!

1.0 Introduction

Data Exploration is a key ingredient in a widely diverse set of discovery-oriented applications, such as scientific computing, financial analysis, evidence-based medicine, and genomics [Sellam and Kersten, 2013, Çetintemel et al., 2013, Kersten et al., 2011]. As traditional DBMSs are designed to answer well-formulated and precise queries, they are rather inappropriate for exploratory discovery-oriented applications, in which queries are typically unknown in advance (e.g., [Idreos et al., 2015, Mishra and Koudas, 2009]). Particularly, in data-driven analysis, there is a need to perform data exploration tasks, in which the user often faces two non-trivial and intertwined challenges, namely: 1) users are unfamiliar with the underlying database schema, and 2) users are unable to construct precise queries that represent their interests due to lack of prior knowledge or capability [Sellam and Kersten, 2013, Mishra and Koudas, 2009]. Consequently, as illustrated in Figure 1, such exploration tasks often involve a large number of tedious *hit-and-trial* iterations, each requiring users to construct an exploratory query and then analyze its results. After exploring the underlying database for a long time, the users will finally know which data is interesting only after they find it. As the volume and complexity of the data sets keep increasing, users will need assistance from the data exploration systems to guide them through the search.

Motivated by the need to address the challenges outlined above, recent research efforts have been directed towards designing data exploration techniques that aim to assist users in formulating and constructing their respective exploratory queries (e.g., [Anagnostopoulos et al., 2010, Feild and Allan, 2013, Islam et al., 2013, Qarabaqi and Riedewald, 2014, Dimitriadou et al., 2014, Diao et al., 2015, Li et al., 2015]). Examples of such techniques include *Query Recommendation* (e.g., [Anagnostopoulos et al., 2010, Feild and Allan, 2013, Li et al., 2008, Eirinaki et al., 2014]), *Query Refinement* (e.g., [Islam et al., 2013, Qarabaqi and Riedewald, 2014, Vartak et al., 2016, Qarabaqi and Riedewald, 2016]), and *Example-driven Exploration* (e.g., [Dimitriadou et al., 2014, Li et al., 2015, Dimitriadou et al., 2016, Ge et al., 2016b, Peng et al., 2017]). Among those tech-

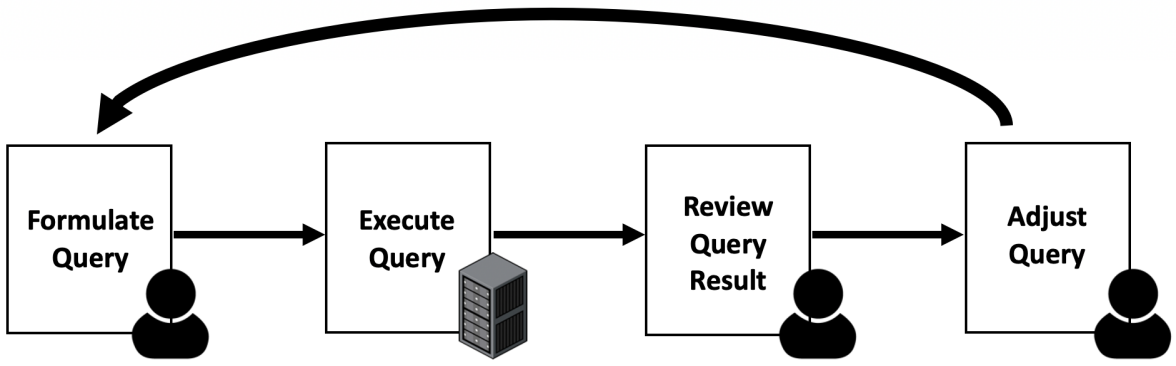

Figure 1: Illustration of the Manual Exploration Process.

niques, Example-driven Exploration is rapidly becoming an attractive choice for query formulation, especially for non-expert users who can circumvent query languages by using examples as input [Mottin et al., 2017]. As such, examples act as a proxy for the intended exploratory analysis as they are representatives of the desired result set. This is as opposed to Query Recommendation techniques that require intensive query logs and user profiles, which are often unavailable when users are exploring datasets for the first time. Similarly, Query Refinement techniques require the user to provide some initial imprecise queries to be progressively refined into a more certain one, a process which clearly requires users to possess some good understanding of the underlying database schema as well as the ability to formulate meaningful queries. The advantages of Example-driven Exploration are further emphasized when combined with interactive techniques (e.g., [Dimitriadou et al., 2014, Ge et al., 2016b, Mottin et al., 2017]).

The main idea underlying the *interactive* Example-driven Exploration techniques, which is the focus of this thesis, is to automatically steer the user towards discovering all data objects relevant to the users' exploration based on their feedback on a small set of examples [Dimitriadou et al., 2014, Diao et al., 2015, Li et al., 2015, Ge et al., 2016b]. For each exploration task, users are iteratively presented by example objects (e.g., tuples or bar charts) from the database for feedback, and in turn, the data exploration platform progressively constructs and refines their exploratory query. In

particular, the user is asked to label a set of carefully selected example objects from the database as either *relevant* (interesting) or *irrelevant* (not interesting) to their exploration task. Based on the provided feedback, the exploration system generates a predictive model, which is used to determine the next set of example objects to be presented to the user. In the background, the exploration system leverages the predictive model to refine the exploratory query that retrieves more objects relevant to the user's task.

Challenges The *effectiveness* of Example-driven Exploration techniques rely on:

(1) ***Correctness*** *maximizing the accuracy* in predicting the user's exploratory queries,

(2) ***Usability*** *minimizing the number of examples* that are presented to the user for their feedback,

(3) ***Efficiency*** *minimizing the processing time* needed to generate useful examples from an arbitrary large database.

Clearly, those three objectives are often in conflict! For instance, maximizing the accuracy of a predictive model often requires a large number of labeled examples, whereas relying on a small set of labeled examples results in a low-accuracy predictive model, and in turn, produces an imprecise query formulation that falls short in capturing the user's interests. Meanwhile, maximizing the accuracy of a predictive model often involves more computations in selecting the best examples to be presented to the user for labeling, whereas relying on less carefully selected examples results in either a low-accuracy predictive model or high-effort in labeling a large set of low-quality examples.

1.1 Limitations of Existing Work

While research in data exploration based on examples is rapidly attracting attention, the ever-increasing volumes of collected data, the growing complexity of exploration tasks, and the demanding needs of data analysts highlight the shortcomings of existing approaches in terms of efficiency, effectiveness, and richness of supported queries.

The primary emphasis of current interactive Example-driven Exploration approaches was given to establishing the foundations for effective exploration and focused on constructing simple selection range search queries, where the extracted data are a set of individual data tuples stored in relational tables. However, users' interest can range from a set of individual relevant tuples, a set of tuples that collectively exhibit a useful aggregation property, or even a set of *unstructured data objects* such as image, text, or graph. Consequently, supporting such rich and complex models of queries and data objects goes beyond the capacity of existing approaches and is expected to improve the *expressiveness* of future Example-driven Exploration solutions.

Additionally, current solutions (e.g., [Dimitriadou et al., 2014, Dimitriadou et al., 2016, Huang et al., 2019]) are primarily based on semi-supervised active learning techniques, which are well suited for offline learning rather than interactive online exploration. Hence, to balance accuracy and processing time, existing systems either require data to fit in main memory or fall short in providing guaranteed interactive response time when data is stored on secondary storage. Clearly, existing techniques cannot scale to interactively generate useful examples from arbitrarily large databases, this points to the need for cross-layer optimizations for fast generation of examples from data on secondary storage, and in turn, providing the desired *efficiency* needed for interactive data exploration.

Lastly, current approaches aim to construct a precise query that describes the user's interest. However, even with queries that precisely captures the user's interest, the amount of data returned in many cases may still be overwhelming, and thus, impractical for the user to explore manually. Obviously, to overcome such a challenge, a set of representative objects needs to be selected from the set of initial results. As pointed out in previous studies [Drosou and Pitoura, 2015], to ensure the original results are properly represented by the set of representative objects, the representative set must consider a variety of aspects of the results such as relevance, dissimilarity, and coverage. Such problem of producing the representative results has previously been named *Top-k result diversification problem* [Drosou and Pitoura, 2015], such that the goal is for any large sets of data items to produce a subset of non-redundant representative items of data based on the user's processing

capacity while minimizing the information loss in the produced representative subset. However, it is known that the Top-k result diversification problem is NP-hard [Khan and Sharaf, 2015], thus, it is challenging to leveraging the Top-k result diversification solutions for the Example-driven Exploration application, given its interactive nature, and even the state-of-the-art approximation approaches are still too expansive to compute for any system that requires real-time interaction with the user [Drosou and Pitoura, 2015]. Therefore, novel solutions that enable efficient and effective Top-k result diversification with a sub-second response time are undeniably needed to ensure the exploration results can be properly interpreted by human users.

1.2 Hypothesis & Objective

The underlying hypothesis of this thesis is that *in order to efficiently and effectively guide users towards unearthing valuable insights from large datasets, data exploration must provide*: ***interactive efficiency***, ***analysis-based effectiveness***, and ***user-in-the-loop engagement***.

In the context of Example-driven Exploration, and in the light of the challenges outlined above, interactive efficiency captures the performance of the data exploration platform. A usable exploration platform must offer well below one second response time in generating each set of examples presented to the user. Analysis-based effectiveness refers to the user's ability to control and customize the exploration process and the corresponding outputs based on the data analysis task in hand and according to their preferences for achieved accuracy and invested effort. User-in-the-loop engagement refers to actively involving the user in the exploration process by means of iteratively providing feedback on the achieved results.

The objective of our work is to develop novel interactive Example-driven Exploration approaches to address the challenges outlined above and implement their corresponding prototype platform that embodies our hypothesis and addresses the aforementioned limitation of existing approaches. Our developed approach support effective Example-driven Exploration that enables data analysts to dig into their data to understand what's in it with ease and extract new knowledge from

it. The novelty of our approach is anchored on leveraging on *active learning*, query optimization, and result refinement techniques to support practical data exploration.

1.3 Our Contributions

In this thesis, we provide the solutions to the challenges faced when developing and deploying next-generation data exploration platforms following the interactive Example-driven Exploration paradigm that is practical, requiring no schema knowledge and little data analytics expertise. Specifically, our solutions addressed its key modeling and algorithmic challenges:

Practical Query Learning The goal of seeking the most informative next example to be labeled from a large dataset is aligned with the existing approaches of *active learning*. Although active learning is shown to be most suitable for accurate offline model training, it is impractical to support query learning in interactive data exploration. The reason is that traditional active learning approaches require an exhaustive search over the entire database before choosing the most informative example to be labeled by the user. In this thesis, we leveraged active learning along with query optimization techniques to develop a novel Example-driven Exploration system, coined *REQUEST*, which consists of both effective data reduction and efficient example selection methods that strike a balance between maximizing accuracy and minimizing user effort in labeling while providing interactive performance. Our results, on real-world datasets from Sloan Digital Sky Survey [sds, 2021], show that our schemes on average require 1 - 2 orders of magnitude fewer feedback questions than the random baseline and 3 - 16x fewer questions than the state-of-the-art while maintaining interactive response time.

Scalable Exploration Beyond Main Memory To achieve desired interactiveness, existing Example-driven Exploration systems operate on main-memory databases, which inherently limits the scalability of these systems. In this thesis, we devised a novel indexing mechanism, called *Uncertainty Estimation Index (UEI)*, which supports the interactivity and scalability of the Example-driven Exploration systems. UEI combines hierarchical in-memory indexing with a columnar

and inverted-indexing-based secondary storage mechanism. It achieves scalability and efficiency through a dynamic estimation of the set of data that are most beneficial to the current exploration. By intelligently manage the in-memory cache, UEI enables Example-driven Exploration systems to scale beyond the main memory restriction while maintains the desired accuracy and convergency speed. Our experiments show that (1) UEI version outperforms the DBMS one by providing more than 50x runtime efficiency when the size of the dataset exceeds the main memory capacity, and (2) is capable of achieving sub-second interactive response time for data that is 100 times larger than the available memory while achieving the desired exploration accuracy and effectiveness.

Flexible Unstructured Data Exploration Existing Example-driven Exploration systems focus solely on structured data, which represents a small portion of the data available today. In this thesis, we developed novel data exploration approaches that enable the user to effortlessly explore the world of unstructured data for insights that are often unreachable from traditional search and exploration methods. In particular, we devised an Example-driven Exploration system tailored for the unstructured dataset, called *ExNav*, which consists of effective exploration and space pruning approaches that exploit the space of advanced machine learning, data embedding, and active learning algorithms. Our experimental evaluation using multiple real-world unstructured dataset (i.e., text, image, and graph) show that ExNav can reduce users' effort by up to 9x while still achieving the same accuracy as the state-of-the-art Example-driven Exploration systems that are not designed for the unstructured data.

Informative Exploration Results Refinement The challenge of scalable data exploration can be examined from two viewpoints. Traditionally, scalability has been seen from a systems point of view, where challenges can be attributed to an increasing rate of data on the one hand, and processing power, and storage limitation on the other hand. However, scalability can also be viewed from a human point of view. Given the exponential volume of data, the challenge here is how to avoid overwhelming users with irrelevant results. Existing Example-driven Exploration systems aim to construct a precise query that describes the user's interest. However, in many cases, even with the most precise query, the amount of data returned may still be overwhelming, and

thus, impractical for the users to comprehend. In this thesis, we devised a novel result refinement algorithm, coined *Preferential Diversity (PrefDiv)*, which addresses the problem of the *Top-k result diversification* that lies in the center of such challenge. Our evaluation with multiple real-world datasets indicates a speedup of up to 159x, and outperforms the state-of-the-art algorithms on multiple fronts.

1.4 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 describes the necessary background that is relevant to our work. Chapter 3 discusses the details of our *REQUEST* system, which is the initial attempt in leveraging active learning for effective Example-driven Exploration. Chapter 4 discusses the details of our novel indexing mechanism *Uncertainty Estimation Index (UEI)*, and how it supports the interactivity and scalability of the Example-driven Exploration systems. Chapter 5 discusses the details of our *ExNav* system, which, for the first time, tailors the Example-driven Exploration specifically for the unstructured dataset. Chapter 6 discusses the details of a novel result refinement algorithm, called *Preferential Diversity (PrefDiv)*, which provides efficient refinement of query and exploration results, and thus, enables the user to consume the exploration results effortlessly. Chapter 7 concludes the thesis by summarizing our contributions and outlining the potential open questions.

2.0 Background

In this chapter, we provide an overview of the necessary background of this work and discuss existing works that are closely related to ours.

2.1 Fully Automated vs. Semi-Automated Data Exploration

Data exploration techniques typically refer to a series of approaches that aims to assist users in extracting useful pieces of knowledge from data without requiring explicit prior knowledge of what the user is searching for. Such a challenging objective has led to numerous approaches to facilitate the process of data exploration. As illustrated in Figure 2, existing data exploration techniques can be broadly distinguished based on the required degree of user involvement into fully automated (e.g., [Tang et al., 2017, Vartak et al., 2015, Crotty et al., 2015, Chirigati et al., 2016, Ehsan et al., 2016]) and semi-automated (e.g., [Dimitriadou et al., 2014, Li et al., 2015, Dimitriadou et al., 2016, Ge et al., 2016b, Peng et al., 2017, Islam et al., 2013, Qarabaqi and Riedewald, 2014]).

A fully automated data exploration approach usually requires the user to supply a subset of data from the database, and then the system recommends useful insights (e.g., other similar or diverse objects, visualizations, aggregates) based on the corresponding dataset provided. Such approaches typically require little interaction with the users and are useful for a quick discovery of insights. However, as pointed out in [Binnig et al., 2017], these approaches have a high risk of providing recommendations that mislead users to consider random fluctuations as significant discoveries. Furthermore, as the recommendation provided by these approaches are based solely on some proposed metrics (e.g., deviation, p-value, etc...), thus they often fail to recommend insights that are of users' true interest.

In contrast, semi-automated data explorations techniques (e.g., [Dimitriadou et al., 2014,

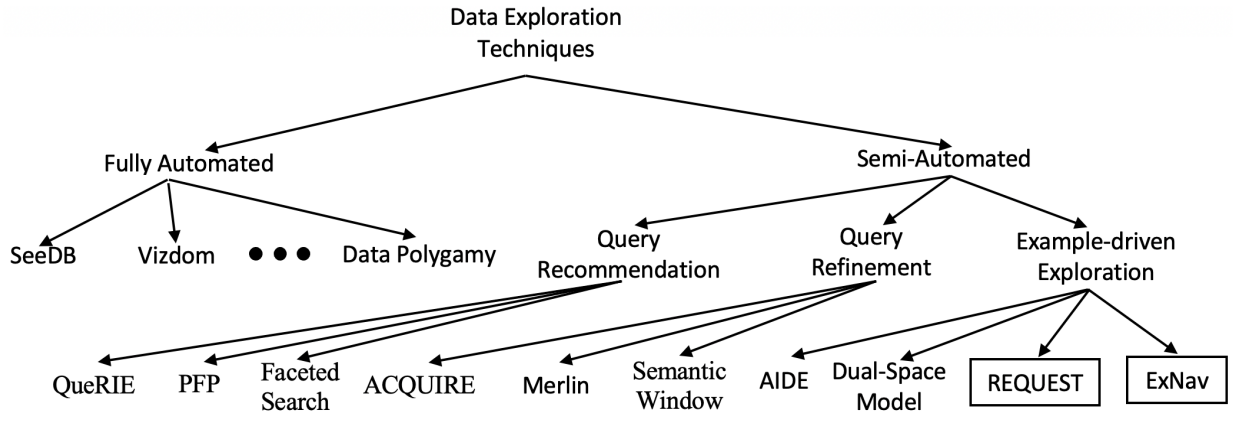

Figure 2: Illustration of the Taxonomy of Data Exploration Techniques. Each Left Node Refers to One Example Data Exploration System. Our Contributions Are Marked With Squares.

Ge et al., 2016b, Islam et al., 2013, Qarabaqi and Riedewald, 2014, Anagnostopoulos et al., 2010, Feild and Allan, 2013, Li et al., 2008, Eirinaki et al., 2014, Vartak et al., 2016, Qarabaqi and Riedewald, 2016]) leverage on users' interests to assist users in formulating their exploratory queries through a dialogue-style exploration process. Specifically, during the exploration process, both the user and the system are constantly providing and receiving feedback from each other. In this way, users are steered towards the queries that capture their true interest, avoiding the risk of being misled by false discoveries.

The design of semi-automated data explorations techniques can be generally divided into three categories, 1) *Query Recommendation* (e.g., [Anagnostopoulos et al., 2010, Feild and Allan, 2013, Li et al., 2008, Eirinaki et al., 2014]), 2) *Query Refinement* (e.g., [Islam et al., 2013, Qarabaqi and Riedewald, 2014, Vartak et al., 2016, Qarabaqi and Riedewald, 2016]), and 3) *Example-driven Exploration* (e.g., [Dimitriadou et al., 2014, Dimitriadou et al., 2016, Ge et al., 2016b, Peng et al., 2017]). As mention in our motivation, out of these techniques, Example-driven Exploration stands out as the more suitable choice for non-expert users because it automatically and progressively formulates the exploratory query through a series of simple

feedback (provided by the user) on the relevance of example data without requiring any schema knowledge or data analytics expertise.

2.2 Example-driven Exploration

In this thesis, we concentrate on the Example-driven Exploration, which supports a semi-automated way of identifying relevant data objects and subsequently formulating queries for data exploration tasks with high precision. Common exploratory queries are in the form of range selection queries that can select all the relevant objects to a user and the exploration task is to determine the predicate values that define the ranges in the query. Figure 3 shows an example of an exploratory query on a 2-dimensional database, where each data object (i.e., tuple) is represented by a 2-attribute data point and the dashed rectangles represent three regions (i.e., ranges) that are determined by the exploration task. This range exploratory query could be:

SELECT (x,y)

FROM 2-D Database

WHERE $(x_{10}, y_{10}) \geq (x, y) \geq (x_{25}, y_{40})\ OR\ (x_{45}, y_{10}) \geq (x, y) \geq (x_{80}, y_{30})\ OR\ (x_{40}, y_{60}) \geq (x, y) \geq (x_{50}, y_{90})$;

Particularly, Example-driven Exploration constructs this query with an iterative conversation between the user and the system. In each iteration, the system selects an example from the underlying database and asks the user "is this object relevant?". Based on the feedback that the user provides, the system employs a predictive model to steer the exploration process, which repeatedly refines the exploratory query until all (or most) interesting objects are captured.

Unfortunately, as discussed earlier, interactive Example-driven Exploration borns with three conflicting objectives, 1) maximize the accuracy in determining the users' exploratory queries, 2) minimize the number of user feedback, and 3) ensure interactive response time between each subsequently presented examples. In particular, the accuracy is defined as follows:

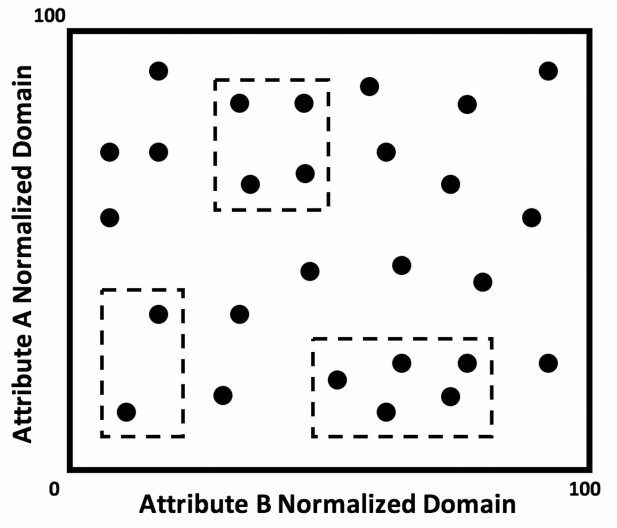

Figure 3: 2-D Database With 3 Relevant Regions.

Definition 1. *The accuracy of an example-driven style exploration system is measured with F-measure. Such that the F-measure is defined as:*

$$F(N) = \frac{2 \cdot \text{Precision}(N) \cdot \text{Recall}(N)}{\text{Precision}(N) + \text{Recall}(N)} \tag{2.1}$$

Here, "Precision" measures the portion of truly relevant data objects among all the data objects considered as relevant by the exploration system. If a data object is irrelevant but considered as relevant by the exploration system, it is considered as false positive. "Recall" measures the ratio between the truly relevant data objects captured by the current system to all the truly relevant data objects in the entire dataset.

The reason for Example-driven Exploration to employ the F-measure as the way to measure accuracy is because the number of objects that are relevant to a user's exploration is often extremely rare compares to the number of irrelevant data objects and the exploratory queries that Example-driven Exploration seeks to construct are often highly selective queries. Therefore, the accuracy measure must emphasize on the errors related to the relevant objects. If another accuracy measure such as *error rate* was chosen, then for an exploratory query that has a 0.1% selectivity, even if the

system constructs an exploratory query that returns an empty set of relevant objects (i.e., consider all objects as irrelevant), it would still have an accuracy of 99.9%. For this reason, Example-driven Exploration more appropriately rely on the F-measure as the way to measure the accuracy of an exploratory query, because it does emphasize the accuracy regarding the relevant data objects. Based on the above description, it is clear that the effectiveness of the Example-driven Exploration heavily depends on its method to select examples for feedback (i.e., labeling).

Algorithm 1 Typical Workflow of Example-driven Exploration

Require: The raw data set D; Batch Size B

Ensure: A set of results R

1: Labeled set $L \leftarrow \emptyset$
2: Unlabeled set $U \leftarrow D$
3: $M \leftarrow$ initialize query strategy
4: **while** user continues the exploration **do**
5: **for** $i = 1$ to B **do**
6: Choose one x from U using M
7: Solicit user's label on x
8: $L \leftarrow L \cup \{x\}$
9: $U \leftarrow U - \{x\}$
10: **end for**
11: $M \leftarrow$ adjusted with L to update M.
12: **end while**
13: Return the set of results R classified as positive by M.

2.2.1 Typical Workflow of Example-driven Exploration

As shown in Algorithm 1, a typical Example-driven Exploration system works in the following steps: first it incorporates a query strategy, which is used for selecting the example objects to be presented to the user for labeling (line 3). As long as the user is willing to label more examples

(line 4), Example-driven Exploration system will keep invoking the query strategy M to select a new example object x from D, and present it to the user to label it as *relevant* or *irrelevant* (lines 5-9). Once the amount of labeled samples received from the user reaches a sample batch size (denoted as B), which is a tunable parameter of the Example-driven Exploration system to balance the effectiveness and efficiency, then the predictive model employed by the query strategy will be updated according to the label assigned to x (line 11). In particular, the label assigned to x will be used for adjusting the query strategy, which is an essential step towards selecting the object presented to the user in the next iteration. Once the iterative labeling process is completed, the obtained results will be returned to the user (line 13).

2.2.2 Existing Example-driven Exploration Approaches

AIDE [Dimitriadou et al., 2014] was the first interactive Example-driven Exploration approach, which proposed a rule-based query strategy. In each exploration iterations, it performs the following three operations. *Misclassified Exploitation*: If a new data object was discovered as relevant in the previous iteration, AIDE will randomly select more objects around this newly discovered relevant object and present to the user for labeling. *Boundary Exploitation*: AIDE also randomly selects examples from around the boundaries of each relevant region (i.e., each set of conjunctive predicates in the exploratory query). *Relevant Object Discovery*: For the areas of the data space that are considered irrelevant by the current exploratory query, AIDE selects some examples to further ensure their (ir)relevance. Particularly, for those areas AIDE partitions the space with d-dimensional grids, then randomly picks a sample from the center of each grid cell. The grid size is adjusted with each refinement of the exploratory query, to further split each grid cell into smaller grid cells in subsequent iterations. Given the operations described above, in each iteration AIDE will present the user with a batch of examples to label which is the union of all the examples generated by those three operations. The set of feedbacks that the user provided will then be used to construct and refine the underlying exploratory query. During the exploration, AIDE uses a decision tree as the internal representation of the exploratory query, which would be transformed

into the actual exploratory query once the exploration is finished.

AIDE's rule-based predictive model has three major limitations: 1) it relies on a heavily parameterized model for navigating the search space, which makes it rather difficult to deploy in practice, 2) it reduces the search space of unlabeled data at the expense of requiring the user to label more samples to achieve high accuracy, 3) it lacks the capability to capture very small relevant regions (i.e., less than 0.01% of the cardinality of the whole dataset), and 4) it rely on main memory to cache the entire exploration dataset, which hinders the scalability of the system.

More recently in [Huang et al., 2019], an enhancement of AIDE called *Dual-Space Model* has been proposed, which uses a new uncertainty sampling-based query strategy and a new dual-space pruning technique that focuses solely on exploration tasks with a single relevant region. It also optimizes these tasks for faster model convergence. However, Dual-Space Model only works with exploration tasks that consist of only a single relevant region, which renders it less useful in real-world scenarios, since it is almost impossible to know in advance the number of relevant regions associated with a explore task prior to the exploration task.

2.3 Active Learning

Active learning approaches can be distinguished into *offline* (Section 3.2.1) and *online* (Section 3.2.2) based on whether the data volume is dynamically expanding as in the context of streaming data.

2.3.1 Offline Active Learning

As the goal of the predictive model in Example-drive Exploration is to seek the most informative next example to be labeled from a large dataset, it is aligned with the existing approaches of *Active Learning* [Settles, 2009]. Active Learning is an interactive learning framework to achieve accurate classification with minimum human supervision. Particularly, a typical active learning

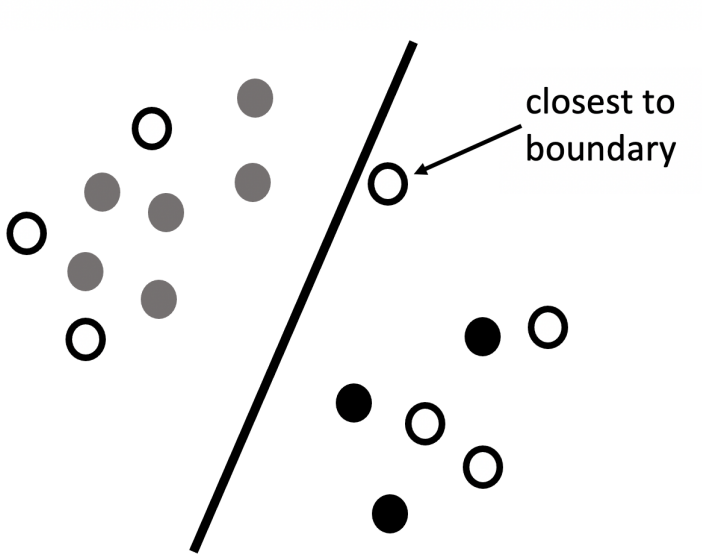

Figure 4: Illustration of Uncertainty Sampling.

approach would employ a *query selection* method to sequentially select which unlabeled example (i.e., object) in the database should be presented to the user next for labeling. A query selection method attempts to minimize the labeling costs by selecting the most informative examples for building the classification model. Different query selection methods to define the "informativeness" of examples have been proposed in the literature, among which the most popular two are *Uncertainty sampling* [Lewis and Gale, 1994] and *query-by-committee (QBC)* [Seung et al., 1992].

Uncertainty Sampling Uncertainty sampling is arguably the most popular query selection method employed by active learning approaches (e.g., [Settles, 2009, Lewis and Gale, 1994, Settles and Craven, 2008]). As illustrated in Figure 4, the intuition underlying uncertainty sampling is that patterns with high uncertainty are hard to classify, and thus if the labels of those patterns are obtained, they can boost the accuracy of the classification models. Particularly, in binary classification models (e.g., with class labels 0 and 1), the most uncertain example $\mathbf{x}$ is the one which can be assigned to either class label $z(\mathbf{x})$ with probability 0.5.

Inspired by such idea of uncertainty, also known as least confidence, [Lewis and Gale, 1994]

proposes a measurement of uncertainty for binary classification models:

$$u^{(lc)}(\mathbf{x}) = 1 - p(\hat{y}|\mathbf{x}) \tag{2.2}$$

where $u^{(lc)}(\mathbf{x})$ is the uncertainty score with least confidence measurement of $\mathbf{x}$. $\hat{y}$ means the predicted class label of the unlabeled $\mathbf{x}$. Accordingly, after measuring the uncertainty of each unlabeled sample, the unlabeled sample with highest uncertainty is selected:

$$\mathbf{x}^* = \text{argmax}_{\mathbf{x}} u(\mathbf{x}) \tag{2.3}$$

where $u(\mathbf{x})$ can be any other measurement of informativeness over the unlabeled sample $\mathbf{x}$.

Query-by-committee Another widely used query selection method is the well known *query-by-committee (QBC)* ([Seung et al., 1992], [Sarawagi and Bhamidipaty, 2002],[Abe and Mamitsuka, 1998]). QBC is basically a voting strategy, in which the uncertainty score is calculated based on a committee of multiple classification models, where each model is trained over a subset of the labeled set. The prediction of an unlabeled point under this voted uncertainty is calculated as:

$$p(1|\mathbf{x}) = \frac{\sum_{i \in C} \hat{y}^{(i)}}{|C|} \tag{2.4}$$

where $p(1|\mathbf{x})$ indicates the probabilistic prediction that $\mathbf{x}$ belongs to class 1, C indicates the committee, $|C|$ indicates the size of the committee and $\hat{y}^{(i)}$ indicates the prediction of $\mathbf{x}$ using the ith model in the committee. In this case, the uncertainty score is calculated as:

$$u^{(v)}(\mathbf{x}) = 1 - p(\hat{y}|\mathbf{x}) \tag{2.5}$$

where $u^{(v)}(\mathbf{x})$ is the uncertainty score with voted measurement of $\mathbf{x}$. $\hat{y}$ indicates the predicted class label of the unlabeled sample $\mathbf{x}$.

Traditional Active Learning vs. Data Exploration On the one hand, active learning approaches have been shown to be effective in terms of: 1) maximizing classification accuracy, and 2) minimizing number of user-labeled samples. On the other hand, applying traditional active

learning approaches has the drawback of introducing long delays as it requires performing multiple iterations of exhaustive search over the database. For instance, in uncertainty sampling, presenting one example to the user for labeling requires computing the uncertainty scores for all the unlabeled objects in the database to select the one with the highest uncertainty score. This process is repeated with each example selection, resulting in long waiting time before the user is able to label the next example, which in turn renders that approach impractical for many applications that access relatively large volumes of data.

2.3.2 Stream-Based Active Learning

In addition to the above-discussed offline active learning techniques, there is one set of active learning techniques that focus on streaming data. As the data volume under consideration grows continuously in data streams, labeling all data objects in a data stream is considered expensive and impractical. Consequently, several stream-based active learning techniques have been proposed to facilitate the training of classifiers in the streaming context (e.g., [Zhu et al., 2007, Zliobaite et al., 2011, Mohamad et al., 2018]). In [Zhu et al., 2007], the author proposed a stream-based active learning technique that seeks to build a classifier ensemble by selectively label instances from different portions of data streams. It does so by first divided the data stream into a series of data chunks. Subsequently, one classifier for each data chunk is built by leveraging an active learning strategy that deliberately selects the data object with the largest disagreement (i.e., ensemble variance) between all previously created classifiers. Although such a technique creates the classifier in an online streaming fashion, for each chunk, an exhaustive search is still required to find the data object that has the largest ensemble variance. Therefore, it cannot be adopted directly for the interactive data exploration setting. Furthermore, since finding the object with the largest ensemble variance requires to exam every object in the chunk with each classifier in the current classifier ensemble, it is computationally more intensive than most mainstream query selection methods such as the uncertainty sampling.

In [Zliobaite et al., 2011], the author proposed a new active learning technique that leverages

uncertainty sampling as the query selection method for stream-based active learning. Unlike the previous approach [Zhu et al., 2007], this technique does not require the stream to be split into chunks, and thus, only one classifier is built for the entire data stream. To ensure the quality of the classifier, [Zliobaite et al., 2011] selects only the data objects that have an uncertainty score higher than an uncertainty threshold that is auto-adjusted accordingly to the average uncertainty score for a given period of time. However, despite the more efficient uncertainty sampling has been adopted as the query selection method, similar to the [Zhu et al., 2007], the stream-based active learning technique proposed in [Zliobaite et al., 2011] also requires to exam the uncertainty score of each data object, and thus, is equivalent to performing an exhaustive search in a non-streaming context.

More recently, in [Mohamad et al., 2018], the author has proposed to leverages the approximation of the expected error as the query selection method for more robust training of the classifiers. Similar to [Zliobaite et al., 2011], [Mohamad et al., 2018] does not require the data stream to be split into chunks and seeks to build a classifier by select the examples that provide a large reduction in expected error. Specifically, each example object that needs to be labeled is selected with a probability that is computed by comparing its expected error with the labeled data object that has the largest reduction in the error among all labeled data objects. Since this method also requires examining each data objects for their expected error reduction, and thus, cannot be adopted directly for the interactive data exploration due to its exhaustive scan.

2.4 Other Interactive Exploration Approaches

A recent tutorial [Mottin et al., 2017] on data exploration included a nice overview of existing interactive exploration approaches. Among these, two approaches are closely related to our work: (1) *Faceted search* is a query refinement technique that iteratively suggests query attributes, which helps the user drill down into structured databases. It requires the user to either continuously provide attribute values until the desired set of relevant data objects are discovered [Roy et al., 2008, Roy et al., 2009, Niranjan Kamat, 2014], or provide certain qual-

ity measurements and its corresponding threshold [Dash et al., 2008]. (2) *Semantic windows* [Kalinin et al., 2014] is a query recommendation technique that allows the user to interactively explore the data space with multidimensional shape-based and content-based predicates that are pre-defined. Unfortunately, the utility of this approach is restricted only to the case where such shape-based, or content-based patterns exactly match the user's interest. Both faceted search and semantic windows require the user to manually control the exploration direction, as opposed to our proposed approach, where the system automatically steers the user towards all interesting tuples.

2.5 Interactive Search for Unstructured Data

Another set of works that are relevant to ours is the interactive search for unstructured data, which has generated a lot of interest in the current era of Big Data, especially in searching for image objects. In [Ferecatu and Geman, 2007], the authors proposed an early work of interactive image search. Specifically, in this method, the author proposed to present a set of exemplar images in each iteration, and these examples are selected by finding images that are predicted as most likely to be the target image with a customized Bayesian model. To increase the coverage of the set of selected exemplar images, the author pre-computes a large set of clusters based on the similarity of the images, and the query selection is conducted at the granularity of clusters instead of at the image level. Consequently, once a cluster is selected by the Bayesian model, the representative image with the highest posterior will be selected to represent the entire cluster that it belongs to. The limitation of this work is that it still requires to perform an exhaustive search over the entire space of the clusters, and as the clusters are pre-computed before the search, therefore, the effectiveness of the method heavily depends on the quality of the clustering, which can be hard to control in real-world scenarios.

More recently, in [Kovashka and Grauman, 2013], the author proposed an alternative interactive image search method that allows users to refine their results by giving feedback on exemplar images. It does so by actively select "pivot" exemplars for which the user feedback will most

reduce the system's uncertainty. One uniques aspect of this approach is in its exemplars that are presented to the user. Instead of presenting just the image, the system presents both the image and a question that builds on one of the metadata attributes of the image. For instance, after presenting an exemplar image, the system will ask the user whether the presenter image is more or less "crowded" than his/her target image. Here, the "crowdedness" is one of the metadata attributes of the image. Underneath the system, a set of binary search trees is built on each of the metadata attributes of the image objects to allow the system to prune the search space according to the feedback that the user has provided. Consequently, the exploration process will be similar to the process of building a decision tree classifier, where each user feedback helps to prune part of the search space along the line of one of the metadata attributes. A selection function is further leveraged to determine the metadata attribute that will be involved in the subsequent questions by predicting the information gain of choosing each metadata attributes' corresponding binary search trees with all the user feedback obtained so far. However, due to its pruning process, such approach is only ideal for data that contains all or mostly numerical metadata attributes and search tasks with only one set of relevant objects, and is less effective for data that consist mostly categorical data with large variance, and when the user is seeking for multiple distinct sets of relevant objects. Furthermore, as the approach assumes a set of human-understandable metadata attributes are available to support the exploration, it not easily applicable to unstructured data that does not support effective derivation of such metadata information (e.g., text documents, interconnected graphs, and voice recordings).

Most recently, in [Modi and Kovashka, 2017], an enhancement of the above-mentioned method (i.e., [Kovashka and Grauman, 2013]) has been proposed, which introduces additional constraints to the above interactive searching process to account for both the model confidence and question diversity when selecting the questions. Consequently, these additional constraints appear to have improved the results in searching the user's target image as compared to the original method [Kovashka and Grauman, 2013]. However, the limitations of the original method, as mentioned above, have not been addressed. Therefore, it is still not suited to be adopted directly

for the interactive data exploration problem.

In this thesis, to ensure effective exploration can be conducted on a variety of different unstructured data types, our unstructured data exploration system *ExNav* (Chapter 5) relies only on the embeddings of the corresponding unstructured data and does not require the support of any additional human interpretable metadata.

2.6 Result Refinement Techniques

Example-driven Exploration techniques seek to assist the users in discovering all of the target objects associated with their exploration task and return the data object in either raw data format or in a query that precisely captures the target data objects. Despite being strongly relevant to the exploration, the set of results discovered through the Example-driven Exploration techniques can still be overwhelmingly large, and thus, impractical to be interpreted by the users. Note that such a problem is not exclusive to the Example-driven Exploration as any user queries can potentially suffer from similar issues. To address this challenging problem, numerous result refinement techniques (e.g., ranking, diversification) have been proposed to optimize the viewing orders of the results. Below we will discuss some of the result refinement techniques that are relevant to our work.

2.6.1 Relevance Ranking Techniques

As comprehensively surveyed in [Stefanidis et al., 2011], a typical result ranking technique can be distinguished based on the type of preferences they support for filtering and ordering data. Most of the ranking techniques primarily handle only one type of preference, either *quantitative* or *qualitative*. However, each preference type has its own advantages and disadvantages. Hybrid schemes (e.g., [Kiessling and Kostler, 2002, Gheorghiu et al., 2014, Gheorghiu et al., 2015]) that support both qualitative and quantitative preferences have been proposed in an attempt to exploit

the advantages of both types of preferences while eliminating their disadvantages.

Among the hybrid schemes, HYPRE model [Gheorghiu et al., 2015] is one recently developed model that integrates qualitative and quantitative preferences by means of *preference strength* or *intensity*. In other words, a preference in the HYPRE model is not seen as a binary option; instead, it allows users to express their preferences along with the intensity of that particular preference, i.e., how "strongly" a user feels about a fact. In the HYPRE model, users submit both qualitative and quantitative preferences along with an intensity value. In order to incorporate the qualitative preferences into the total order generated by the quantitative preferences, the qualitative preferences are converted into quantitative preferences by deriving an intensity value based on the existing qualitative preference intensity value and a quantitative preference intensity value (or a default value if this does not exist). The HYPRE model stores preferences in a labeled directed and acyclic graph. Each node in the graph represents a query predicate. Quantitative preferences are represented using edges that have the same starting and ending point. Qualitative preferences are represented by edges between two different nodes. Each edge is labeled with a value that represents the preference's intensity. Preference intensity is a decimal value between 0 and 1 and is used to express either a negative preference, a positive preference, or equality/indifference.

More recently, in [Celis et al., 2018], the author studied the problem of producing rankings while preserving a given set of fairness constraints. In particular, the proposed algorithm takes as input a utility function, a collection of sensitive attributes (e.g., gender, race), and a collection of fairness constraints that restrict the number of items with each sensitive attribute that are allowed to appear in the final results. It outputs a ranking that maximizes the relevance with respect to the given utility function while respecting the fairness constraints.

2.6.2 Diversity Techniques

Result diversification has been studied in many different contexts and with various definitions [Drosou and Pitoura, 2012b], such as similarity, semantic coverage [Agrawal et al., 2009], and novelty [Clarke et al., 2008]. In our work, we focus on the similarity definition and use MaxMin

and MaxSum, which are two widely used diversification models, as baselines.

The goal of these two diversification models is to select a subset S from the object space R, so that the minimum or the total pairwise distances of objects in S is maximized. Recently, a number of variations of the MaxMin and MaxSum diversification models have also been proposed (e.g., [Panigrahi et al., 2012, Drosou and Pitoura, 2012a]) to address the problem of diversifying continuous data. Formally, MaxMin and MaxSum are defined as follows:

Definition 1. MaxMin *generates a subset of* R *with maximum* $f = min_{p_i,p_j \in S} dt(p_i, p_j)$ *where* dt *is some distance function* $p_i \neq p_j$ *for all subsets with the same size.*

Definition 2. MaxSum *generates a subset of* R *with maximum* $f = \Sigma_{o_i,o_j \in S} dt(o_i, o_j)$ *where* dt *is some distance function* $o_i \neq o_j$ *for all subsets with the same size.*

DisC Diversity [Drosou and Pitoura, 2012b] is the most recently proposed diversity framework and solves the diversification problem from a different perspective. In DisC Diversity, the number of retrieved diverse results is not an input parameter. Instead, users define the desired degree of diversification in terms of a tuning parameter r (radius). DisC Diversity considers two objects o_i and o_j in the query result R to be similar objects if the distance between o_i and o_j is less than or equal to a tuning parameter r (radius). It selects the representative subset $S \in R$ according to the following conditions: (1) For any objects in R, there should be at least one similar object in S; and (2) All objects in S should be dissimilar to each other. These two conditions ensure both the coverage and the dissimilarity property of a diverse result set.

In addition, DisC Diversity also introduces two problems, *Covering* and *CoveredBy* [Drosou and Pitoura, 2015]. These can be used to model the issue of generating a representative result set that is both diverse and relevant to a user's individual preference (without using preferences). The Covering problem is used to model the case where users want highly relevant items to cover a large area around them. In order to achieve this goal, a relatively larger radius is assigned to items with larger weights. The CoveredBy problem is used to model a case where a user wants to see more relevant objects. In that case, a smaller radius is assigned to items with larger weights. These two problems together illustrate the possibility of using DisC to handle relevance together

with diversity.

Another way to generate a diverse, representative set of results is through clustering. One example of this would be *k-Medoids*, which is a well-known clustering algorithm that attempts to minimize the distance between points in a cluster and the center point (medoid element) of that cluster. The k-Medoids algorithm can be classified into two stages: In its first stage, it generates a set of k clusters C = {c1, c2, ..., ck} based on some distance function dt. In the second stage, one element from each cluster is selected to be part of the result set R. Several strategies for selecting an element from each cluster could be employed. For instance, one strategy is to choose the center point of each cluster that is expected to deliver high diversity, and another strategy would be to choose the point that has the highest intensity value for each cluster. However, since there is no parameter that can be tuned, either manually or automatically, to balance the trade-off between relevance and diversity, k-Medoids is unable to balance such a trade-off in fine granularity.

2.6.3 Multi-Criteria Objective Optimization

In the past, diversification and retrieval of relevant results have often been studied together as a multi-objective optimization problem with two objectives, where the first objective is relevance, and the second objective is dissimilarity [Ziegler et al., 2005]. The following are some representative techniques that are related to our work.

In [Qin et al., 2012], the authors considered the optimization of the diversified Top-K problem as finding the optimal solution for the maximum weight independent set problem, which has been proven to be an NP-hard problem. The authors proposed an approach, called *div-astar*, which uses a diversity graph that consists of N nodes, where each node corresponds to one item in the original data. This diversity graph is sorted according to the relevance score, and an a^* algorithm is used to find the optimal solution for diversifying Top-K Results. In addition to the div-astar solution, two enhancements have also been proposed, called *div-dp* and *div-cut*: div-dp takes advantage of dynamic programming to divide the initial graph into disconnected components, and div-cut is a cutpoint-based approach that further decomposes each disconnected component based on loosely

connected sub-graphs.

One widely used approach that was targeted directly at optimizing the trade-off between diversity and relevance was introduced by [Carbonell and Goldstein, 1998]. In this chapter, the authors proposed the famous twin-objective function called *Maximal Marginal Relevance* (MMR), which combines both relevance and diversity aspects in a single comprehensive objective function. Formally, MMR defines its objective function as:

$$\underset{D_i \in RnS}{\operatorname{argmax}}[\lambda(Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j))] \tag{2.6}$$

where λ is a scaling factor that specifies the preference between relevance and diversity. When $\lambda = 1$, the MMR function equals a standard relevance ranking function. When $\lambda = 0$, it computes a maximal diversity ranking.

Recently, a new bi-criteria objective optimization approach based on MMR has been proposed [Hussain et al., 2015]. This approach integrates *regret minimization* with traditional MMR to generate a new relevance score that takes into consideration the case of minimizing the disappointment of users when they see k representative tuples rather than the whole database. In this chapter, the authors proposed two approximation algorithms called *ReDi-Greedy* and *ReDi-SWAP*, which find the set of items consisting of k items having the highest score with respect to their MMR function.

In [Stefanidis et al., 2010], the author has conducted a study on personalized, keyword-based search over relational databases, which includes the notion of diversity and coverage. Specifically, the author provided good discussions on modeling the relevance, user preferences, diversity, and coverage for keyword-based searches over relational databases by means of Join Tree of Tuples. Join Tree are trees of tuples connected through primary to foreign key dependencies.

Swap is another recent Top-K diversification technique that is related to ours [Yu et al., 2009]; Swap starts with K items with the highest relevance scores. Among these K items, Swap picks an item with the lowest contribution to the diversity of the entire set, then swaps this item with the item that has the next highest relevance score. A candidate is successfully swapped with one of the items in the Top-K set if and only if it can contribute more in terms of the overall diversity of

the result set. In order to preserve the relevance aspect, Swap introduces an optional pre-defined threshold called UB that specifies how much decrease in relevance can be tolerated. UB can serve as a terminal condition that stops the algorithm when the item with the highest relevance among the remaining set is no longer high enough for the algorithm to perform a swap operation.

2.6.4 Data Summarization

Some recent works [Wen et al., 2018, Manas Joglekar, 2016] have studied the problem of providing interactive exploration and summarization support for tuples in a given table. The goal of this type of approach is to produce an informative hierarchy that organizes the underlying tuples essentially in k clusters. In order to display tuples as clusters, each cluster is folded into a single, representative tuple, with only the common attribute values among all members of the cluster being displayed. The rest of the attributes are shown as "?", which indicates that there are objects with different values with respect to these attributes inside the cluster. To explore each cluster, the user can gradually expand each "?" symbol contained in the current representative tuple of a cluster. Each time the user expands a "?" symbol, more tuples that contain a different value with respect to the corresponding attributes will be displayed.

2.7 Summary

In this chapter, we reviewed the related work in interactive data exploration and result refinement, and provided a taxonomy of available data exploration techniques that can be broken down into fully automated and semi-automated data exploration techniques. From the large volume of existing literature, it is certain that effective and efficient data exploration techniques are in high demand. We provided a brief overview of the fully automated data exploration technique and pointed out some of its' limitations, which essentially render it inappropriate for many real-world exploration workloads. For semi-automated data exploration techniques, we have further broken

down the existing approaches into three main categories: *Query Recommendation*, *Query Refinement*, and *Example-driven Exploration*, and provided necessary background on each of them. As our work is focused on Example-driven Exploration, we provided a detailed analysis of the existing works (e.g., active learning) that are strongly related to ours and shown their limitations, which are addressed by our new techniques presented in this dissertation (Chapters 3 - 5). Additionally, we summarized the related work in result refinement since it is essential to the usability of the Example-driven Exploration system, especially in the case where the results are to be consumed directly by the user (i.e., not fed into subsequent applications). These works in result refinement provided the foundation of our novel result refinement technique *PrefDiv* (Chapter 6).

3.0 Effortless Exploring of Structured Data

The work covered in this chapter was published in the Proceedings of 2016 IEEE International Conference on Big Data (IEEE BigData) [Ge et al., 2016b].

In this chapter, we discuss our solution that seeks to minimize the user effort when exploring large structured data with Example-driven Exploration. Our solution leverages and adopts the technique of *active learning* [Settles, 2009], which refers to a family of algorithms that seeks to minimize the number of labels needed when training a predictive model.

3.1 Introduction

The main idea underlying the Example-driven Exploration techniques is to automatically identify all data objects relevant to the user's exploration task based on their feedback on a small set of sample objects [Dimitriadou et al., 2014, Li et al., 2015]. In particular, the user is iteratively presented by sample tuples (i.e., objects) from the database, and in turn, the data exploration platform progressively constructs and refines their exploratory query. Clearly, this is an example of a "human-in-the-loop" task, in which the typical setting is to start asking the user "If you are interested in some object or not?", followed by repeatedly refining the questions until all (or most) interesting objects are discovered. As discussed in Chapter 2, to achieve the desired effectiveness, Example-driven Exploration approaches need to achieve both: *maximizing the accuracy* of the employed model in predicting the users' exploratory queries, *minimizing the number of samples* that are presented to the users for their feedback, and *minimizing the processing time* needed to generate useful examples from an arbitrary large database. However, these objectives are often in conflict. For instance, maximizing the accuracy of a predictive model often requires a large number of labeled samples, whereas relying on a small set of labeled samples results in a low-accuracy predictive model, and in turn imprecise query formulation that falls short in capturing the user's interests.

In this chapter, we propose a novel solution for Example-driven Exploration, called *REQUEST* (Data **Re**duction and **Que**ry **S**elec**t**ion), designed to address the conflicting objectives outlined above to enable both effective and efficient example-driven data exploration. In particular, REQUEST aims to assist users in constructing highly accurate exploratory queries while at the same time minimizing the number of samples presented to them for labeling. To achieve that goal, REQUEST integrates the following two components: 1) *Data Reduction*: it employs efficient data reduction techniques to minimize the amount of explored data, and in turn the number of required labeled samples (e.g., [Mozafari et al., 2014, Haas et al., 2015]), and 2) *Query Selection*: it employs effective active learning query selection methods to maximize the accuracy of the predictive model, and in turn the preciseness of the constructed queries (e.g., [Settles, 2009, Settles and Craven, 2008, Sarawagi and Bhamidipaty, 2002]).

Given our REQUEST general framework (Section 3.2), we further proposed several specific schemes that provide different levels of efficiency and effectiveness as guided by the user's preferences (Section 3.3). Our experimental evaluation using the Sloan Digital Sky Survey (SDSS) [sds, 2021] dataset shows that our schemes can achieve the same accuracy as state-of-the-art schemes while reducing the user's efforts incurred in providing feedback (i.e., labeling samples) by up to 93% (Section 3.4). Moreover, our schemes are also able to construct, with high accuracy, queries that are often undetectable by current techniques, even when a large number of samples are explored.

3.2 The REQUEST Framework

REQUEST is designed to achieve the following goals: 1) minimize the number of example objects that are presented to the user for labeling, 2) minimize the processing cost and delays incurred in selecting those examples, and 3) maximize the accuracy of the constructed query in capturing the user's interests. The main idea underlying REQUEST is to combine the advantages of data reduction techniques (used in data exploration platforms) with the advantages of query

Algorithm 2 The REQUEST Framework

Require: The raw data set D; Batch Size B

Ensure: A range query Q

1: Subset $D_s \leftarrow D$
2: Labeled set $L \leftarrow \emptyset$
3: Unlabeled set $U \leftarrow D_s$
4: $M \leftarrow$ initialize query selection method
5: **while** user continues the exploration **do**
6: **for** $i = 1$ to B **do**
7: Choose one x from U using M
8: Solicit user's label on x
9: $L \leftarrow L \cup \{x\}$
10: $U \leftarrow U - \{x\}$
11: **end for**
12: $M \leftarrow$ trained with L to update M.
13: Train a decision tree with L to obtain Q
14: **end while**
15: Return the most recently obtained Q.

selection methods (used in active learning approaches).

As shown in Algorithm 2, REQUEST works in two stages, namely: *data reduction* and *query selection*. In the first stage, REQUEST reduces the original dataset D to a subset D_s, such that $|D_s| < |D|$ (line 1). In order to achieve effective data reduction, we consider different methods, including traditional sampling and user-guided sampling based on the multi-instance active learning (MIAL) approach. Then REQUEST initializes two sets of data objects L and U for storing the labeled and unlabeled data objects, respectively (lines 2-3).

REQUEST also incorporates a query selection method that is used for selecting the example

objects to be presented to the user for labeling (line 4). In this chapter, we employ and extend variants of the well-known uncertainty query selection methods, which are described in detail in Section 3.2.2. As long as the user is willing to label more examples (line 5), REQUEST will keep invoking the query selection method M to select a new example object x from D_s, and present it to the user to label as relevant or irrelevant (lines 6-11). Once the amount of labeled samples received from the user reaches a sample batch size (denoted as B), which is a tunable parameter of the REQUEST to balance the effectiveness and efficiency, then the classifier model employed by the query selection will be updated according to the label assigned to x (line 12).

In particular, the label assigned to x will be used for retraining the classifier model, which is an essential step towards selecting the object presented to the user in the next iteration. Once the iterative labeling process is completed, a decision tree classifier is trained on all the labeled data, and a range selection query Q is constructed based on that tree, as described in Section 2.2 (line 13).

Next, we describe in detail the different methods for data reduction and query selection considered under our framework.

3.2.1 Data Reduction

In active learning approaches, query selection methods are applied over the entire set of unlabeled data objects to select the next example to be presented to the user for labeling. As discussed earlier, such exhaustive search incurs high processing costs and leads to users experiencing long delays between the presented examples. To the contrary, data exploration approaches employ data reduction techniques for minimizing the number of unlabeled objects to be searched. However, current data reduction approaches, such as the ones employed by AIDE (see Section 2.2) have the drawback of presenting the user with a large number of examples in order to construct a query with high accuracy. In the REQUEST framework, we consider simple yet effective data reduction methods that overcome the aforementioned limitations, namely: *Data-Driven Sampling*, and *User-Driven Pruning*.

3.2.1.1 Data-Driven Sampling In traditional sampling, a small subset of data objects D_s is randomly extracted from the complete database D. As such, the query selection method for selecting examples for labeling is applied on the sample D_s to reduce the processing cost and waiting delays (see Algorithm 2). For example, if N is the size of the complete data and p is the sampling ratio, then D_s will include $p \times N$ tuples, and the remaining tuples are discarded. Note that $p = 1.0$ is a special case referred to as *None*, in which no sampling is applied and all the database tuples are considered, which is equivalent to traditional active learning approaches in which no data reduction is employed.

3.2.1.2 User-Driven Pruning Inspired by the Multi-Instance Active Learning (MIAL) [Settles et al., 2008], we introduce another method for selectively reducing the search space of unlabeled objects. In MIAL, objects are grouped into a set of bags, and the user is requested to give their feedback on a bag of objects rather than an individual object. Accordingly, MIAL assumes that a bag is irrelevant if every object in that bag is of no interest to the user; otherwise, the bag is interesting.

Such property of MIAL making it well-suited to exploration tasks for which the volume of available data objects is large but the target is represented by only a small set of data objects. In [Settles, 2009], it was shown with examples that as fully labeling all data objects is expensive, it is possible to obtain labels both at the bag level and directly at the data object level, which inspires us to take advantage of coarse labelings to quickly eliminate irrelevant subspaces, then use finer query selection strategies at the object-level to polish the answer.

In REQUEST, we employ MIAL as the User-Driven Pruning (UDP) technique to prune subspaces that are completely irrelevant to the exploration task, and in turn, reduce the amount of data considered for labeling. To achieve this, REQUEST divides the multi-dimensional data space evenly into a number of d-dimensional hyper-rectangles. That is, each dimension of the data space is split evenly into a certain number of bins. Thus, if each dimension is split into m bins, there will be m^d hyper-rectangles in total. Then the user is presented with the ranges covered by each

hyper-rectangle and is asked to label the rectangle as negative if it does not contain any relevant objects or as positive otherwise. All negative rectangles are discarded, and all remaining positive ones are passed to the query selection method.

3.2.2 Query Selection

Query Selection is the one component of our REQUEST framework that aims to minimize the labeling effort while maximizing the accuracy of the constructed queries. It is worth mentioning that in this context, a "query" refers to the process of selecting an example object to be presented to the user for labeling. In REQUEST, we use uncertainty sampling to quickly learn the interest of the user and steer them towards the relevant data regions.

3.2.2.1 Uncertainty Sampling As mentioned in Section 2.3, uncertainty sampling is a popular active learning technique that aims to choose the data points which are most beneficial to build a classification model that precisely captures both relevant and irrelevant data regions.

According to Equation 2.2, to measure the uncertainty of a data object x, we need a model that would always report the probability of x being positive or negative. The decision tree classifier (that we employed for range query) is not strictly a probabilistic model, and therefore in most cases, the decision tree is not an ideal choice for calculating the uncertainty score. Thus, another probabilistic classification model, such as the Naive Bayes Classifier, is needed to determine the uncertainty score. This model will be built on the same labeled dataset as the decision tree so that they can reflect the same "knowledge" of the interest objects.

3.2.2.2 Naive Bayes Classifier Naive Bayes classifier [Lowd and Domingos, 2005] is a classification model, which can be perceived as a mixture of multiple logistic regression models. In that sense, Naive Bayes generates similar decision boundaries to a decision tree (because multiple hyperplanes form an enclosed decision boundary similar to hyper-rectangle or hyper-ball) with probabilistic scores for calculating uncertainty. Accordingly, REQUEST trains our Naive Bayes

classifier on the same set of data (all user labeled objects) at the same time it trains the decision tree classifier. Therefore for a given time point T the probabilistic score given by Naive Bayes classifier reflects the uncertainty of any unlabeled data object at time T.

To be more precise, the uncertainty of a data object under Naive Bayes classifier is determined as follows. Given an unlabeled data sample to be predicted, represented by $\mathbf{x}$, it assigns to this probability:

$$p(C_k|\mathbf{x}) \tag{3.1}$$

for each hidden class C_k. Another assumption of Naive Bayes classifier is the conditional independence, such that Naive Bayes classifier assumes that each feature is conditionally independent of every other feature:

$$p(\mathbf{x}_i|\mathbf{x}_{i^\complement}, C_k) = p(\mathbf{x}_i|, C_k) \tag{3.2}$$

where $\mathbf{x}_i$ is the ith feature of $\mathbf{x}$ and $i^\complement$ is $\mathbf{x}$ without the ith feature.

Thus, the joint probability can be expressed as:

$$\begin{aligned} &p(C_k|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) \\ \propto &p(C_k, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) \\ \propto &p(C_k) \prod_{i=1}^{m} p(\mathbf{x}_i) \end{aligned}$$

where i is the enumeration of all features. This means that under the above independent assumptions, the conditional distribution over the hidden variable is:

$$p(C_k|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) = \frac{1}{Z} p(C_k) \prod_{i=1}^{m} p(\mathbf{x}_i) \tag{3.3}$$

where $Z = p(\mathbf{x})$ is the normalization factor.

Such conditional distribution reflects the likelihood of a data object x being positive or negative (in our case, relevant or irrelevant), which is essentially the uncertainty score of x.

3.2.2.3 Committee of Naive Bayes Classifiers

Following the voted uncertainty method mentioned in Section 2.3, a small committee of Naive Bayes classifiers x can be used as an alternative way to determine the uncertainty score of an unlabeled object [Sarawagi and Bhamidipaty, 2002]. Under this method, the uncertainty score is calculated through Equation 2.5. Additionally, according to [Settles, 2009, Seung et al., 1992, Settles and Craven, 2008], a small committee (less than 6) work well in practice for such method.

In REQUEST, we employ both Naive Bayes Classifier and Committee of Naive Bayes Classifiers for calculating the uncertainty score. However, traditional uncertainty sampling suffers from two major drawbacks 1) *shortsightedness* (as pointed out in [Brachman et al., 2012]), and 2) *low scalability*. Shortsightedness refers to the issue that the uncertainty score of unlabeled samples is based only on the information obtained from labeled samples, which usually is a tiny portion compared to the unlabeled objects, therefore, causes a biased when selecting samples for labeling. Low scalability is caused by the fact that traditional uncertainty sampling requires performing an exhaustive search over all unlabeled datasets for every sample that is presented to the user.

3.2.2.4 Randomized Uncertainty

To address the first drawback mentioned above, the work in [Xue and Hauskrecht, 2016] combines uncertainty with some degree of randomness. Particularly, for each query, an example is probabilistically selected from all the unlabeled objects for the user to label. The probability that an unlabeled point x is selected is proportional to its uncertainty score:

$$p(\mathbf{x} \text{ is selected}) = \frac{u(\mathbf{x})}{\sum_{\mathbf{x}_u \in U} u(\mathbf{x}_u)} \tag{3.4}$$

where U is the unlabeled set and $u(\mathbf{x})$ is the uncertainty score of an unlabeled sample $\mathbf{x}$.

Since the probability that an unlabeled point x is chosen to be presented to the user is equal to its normalized uncertainty score, therefore, less uncertain samples may still have a small chance of being accepted, which has the effect of reducing the bias introduced by the labeled samples.

3.2.2.5 Randomized Accept/Reject Uncertainty (RARU)

When exploring large datasets, randomized uncertainty still suffers from low scalability for the fact that each example selection must go over all the unlabeled object to compute their normalized score. Thus, to address the second drawback of uncertainty sampling, we introduce an accept/reject example selection approach. Particularly, to choose an unlabeled sample to be presented to the user, we first randomly pick an unlabeled object x, then calculate the uncertainty score of x. To decide whether x can be presented to the user we use the uncertainty score of x as the way to determine its acceptance, such that the probability of an unlabeled data sample **x** being accepted under RARU is:

$$p(\mathbf{x} \text{ is selected}) = \underset{k \in 0,1}{\operatorname{argmin}} \frac{Pr(C_k|\mathbf{x})}{0.5} \tag{3.5}$$

where $Pr(C_k|\mathbf{x})$ is the probability of **x** being assigned a binary class label C_k, and 0.5 is a normalizing factor since a prediction score of 0.5 indicates the classifier is most uncertain about an object.

If x is accepted, it will be presented to the user for labeling. Otherwise, RARU will continue to select other unlabeled objects randomly until one object is accepted (according to Equation 5.4) and presented to the user.

Such Randomized Accept/Reject Uncertainty strategy provided an early termination to prevent the enumeration of all unlabeled data items while still preserving the feature of randomized uncertainty that mitigates the shortsightedness of the traditional uncertainty sampling.

3.3 The REQUEST Schemes

In this section, we introduce four specific schemes to support the Example-driven Exploration style of data exploration that are based on our REQUEST framework. These schemes are: 1) The None+RARU Scheme, 2) The None+VotedRARU Scheme, 3) The MIAL+RARU Scheme, and 4) The MIAL+VotedRARU Scheme.

3.3.1 The None+RARU Scheme

This scheme focuses on the Randomized Accept/Reject Uncertainty (RARU) as the query selection method with no data reduction techniques used. In None+RARU, a single Naive Bayes Classifier is employed to compute the uncertainty scores. An issue of this scheme is the initial sample acquisition, since initially no relevant objects are discovered. Therefore no uncertainty score can be computed. To solve this issue, we randomly sample unlabeled data objects from the database for labeling until the first relevant object is discovered, thus, enables the Naive Bayes classifier to compute the uncertainty score of unlabeled objects. Once the first relevant object is discovered, the choice of selecting subsequent samples for labeling would be determined according to Equation 5.4.

3.3.2 The None+VotedRARU Scheme

The None+VotedRARU scheme is similar to the None+RARU Scheme except the way how uncertainty score is computed. In None+RARU Scheme, a single Naive Bayes classifier is used to determine whether to reject/accept a sample from being presented to the user. In the case of the None+VotedRARU Scheme, this decision is co-decided by a committee of Naive Bayes classifiers. In particular, given the labeled data L, we use the k-fold cross-training method to train k different classifiers, and the uncertainty score of an unlabeled object is computed according to Equation 2.5.

3.3.3 The UDP+RARU Scheme

For this scheme, we employ the User-Driven Pruning (UDP) technique based on MIAL as the data reduction technique. We start with a partition of d-dimensional data space into multiple d-dimensional grids with a tunable parameter δ, such that each of the attributes is split into δ equal width ranges, and each grid covers a range of $100/\delta$ of the domain for each attribute. We then query the user on the ranges covered by each grid and prune those grids that are labeled as irrelevant. For each of the relevant grids, we would visit them in a round-ribbing fashion and apply the RARU

strategy (with Naive Bayes classifier) to select samples from each grid for labeling. Later, each labeled sample from any positive grids would be feed into a decision tree model to capture relevant regions.

3.3.4 The UDP+VotedRARU Scheme

This scheme is the same as the UDP+RARU Scheme, except that the VotedRARU strategy is used to compute the uncertainty score.

3.4 Experimental Evaluation

In this section, we will present the results of our experiments. We start this section by introducing the experiment setup. Then we demonstrate the performance of our schemes and other alternatives.

3.4.1 Experiment Setup

SDSS Dataset In our experiments, we used 40 GiB of the real-world dataset from Sloan Digital Sky Survey (SDSS) [sds, 2021] that consists of 10×10^6 tuples.

Environment Since we are unable to obtain the original implementation of AIDE, we faithfully implemented all algorithms with Java JRE 1.7, and all the experiments were run on an Intel Core i7 4-core CPU with 24GiB RAM. We used the Weka [Witten and et al., 1999] library for executing the J48 decision tree algorithm. All experiments reported are averages of 10 complete runs. We have considered five numerical attributes *rowc, colc, ra, field* and *fieldID* of the PhotoObjAll table.

Target Interest Regions The exploration task characterizes user interests and eventually predicts the interest regions by iteratively gathering user-labeled tuples. As mentioned before, we

Table 1: Experimental Parameters of REQUEST Schemes

Total number of data objects	10×10^6
Number of runs per result	10
Number of dimensions considered	2, 3, 4, 5
Number of relevant regions	1, 3, 5
Number of Committees	5
Cardinality of the relevant regions	0.1% (S), 0.4% (M), 0.8% (L)
Data Reduction Strategies	Sampling, UDP
Number of grids	3^D (D = Number of Dimensions)
Query Selection Methods	RARU, VotedRARU
Uncertainty Sampling algorithm	Naive Beyes
Decision Tree Algorithm	J48 (C4.5)
Considered Sample Batch Sizes	50
Baseline Schemes	AIDE, Random
Our Schemes	None+RARU, None+VotedRARU UDP+RARU, UDP+VotedRARU

focus on predicting range queries (the user interest regions), and in our experiments, we experiment with three different interest region amounts $\{1, 3, 5\}$. Further, we vary the single region complexity based on the data space coverage of the relevant regions. Specifically, we categorize relevant regions to *small*, *medium*, and *large*. Small regions have cardinality with an average of 0.1% of the entire experimental dataset, medium regions have a cardinality of 0.4%, and large regions have a cardinality of 0.8%.

User Simulation Given a target interest region, we simulate the user by executing the corresponding range query to collect the exact target set of relevant tuples. We rely on this "oracle" set

to label the tuples we extract in each iteration as relevant or irrelevant depending on whether they are included in the target region. We also use this set to evaluate the accuracy, i.e., F-measure (Section 2.2), of our final predicted range queries. Further, we consider each sample as one question to the user. Thus, the questions asked by the data reduction strategy (e.g., UDP) are also counted as one sample.

Parameters Table 1 shown a list of all settings and schemes of the experiment. By default, we use 1×10^6 distinct tuples, a batch size of 50, a committee size of 5, and attributes *rowc* and *colc*, unless otherwise specified. The words "f-measure" and "accuracy" are interchangeable in the text below.

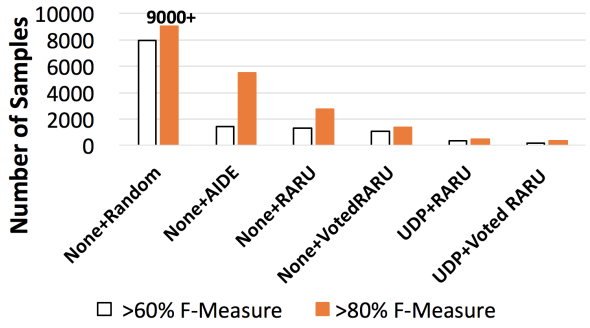

Figure 5: Accuracy, 2D, 1 Small Region

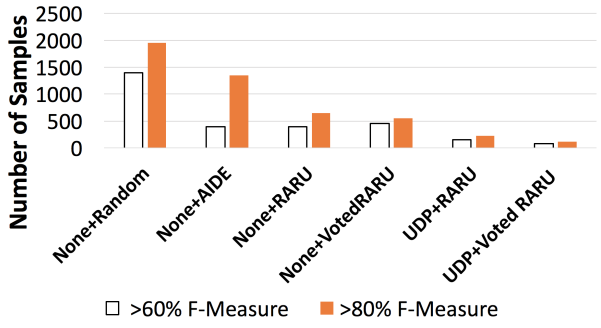

Figure 6: Accuracy, 2D, 1 Medium Region

3.4.2 Experimental Results

Accuracy Comparison Figures 5-15 shown the number of samples needed to reach an accuracy (f-measure) of 60% and 80% of all participating algorithms with different range queries. Here we vary the target region size from small to large and change the target region numbers with three different settings 1, 3, and 5. From these figures, we observed that our scheme UDP+VotedRARU consistently demonstrates the highest effectiveness when compared to other alternatives. Such that, to reach an accuracy of 60% UDP+VotedRARU only 150-200 samples for small regions, less than

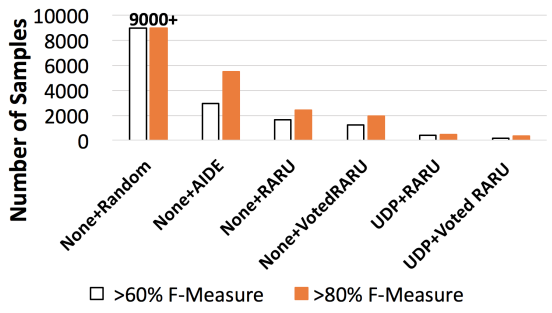

Figure 7: Accuracy, 2D, 3 Small Regions

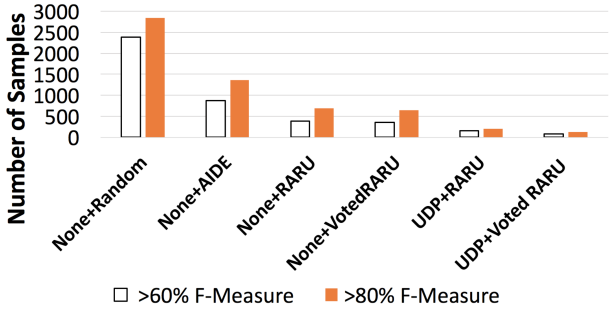

Figure 8: Accuracy, 2D, 3 Medium Regions

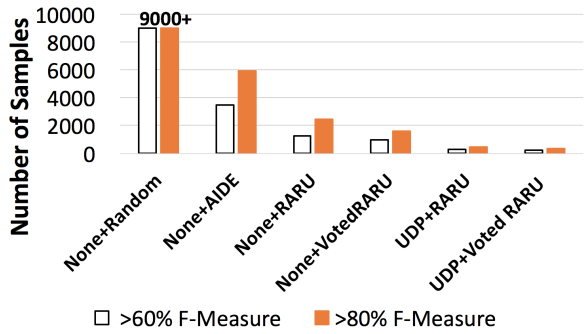

Figure 9: Accuracy, 2D, 5 Small Regions

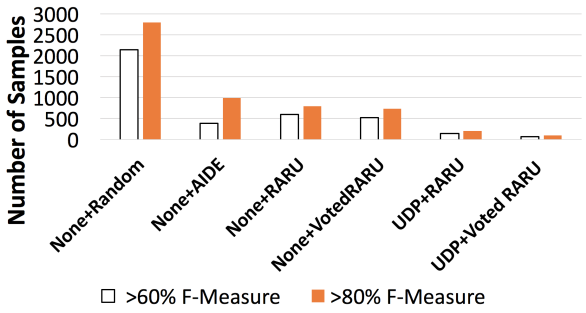

Figure 10: Accuracy, 2D, 5 Medium Regions

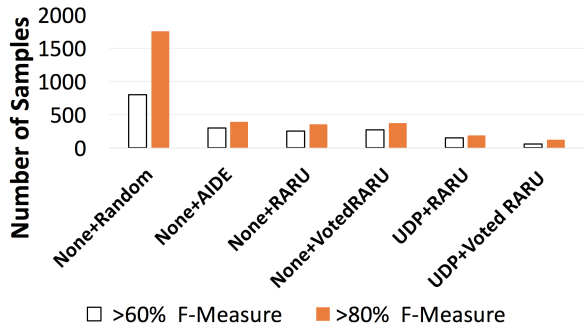

Figure 11: Accuracy, 2D, 1 Large Region

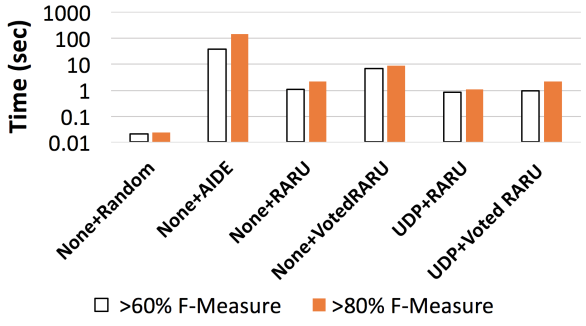

Figure 12: Runtime, 2D, 1 Small Region

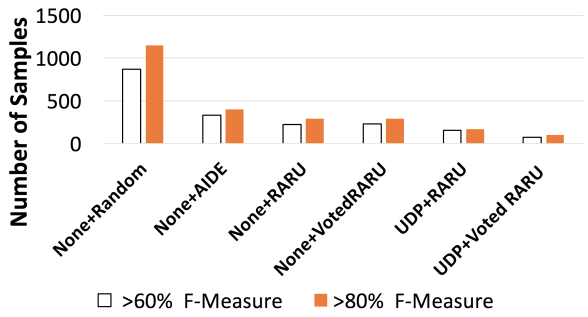

Figure 13: Accuracy, 2D, 3 Large Regions

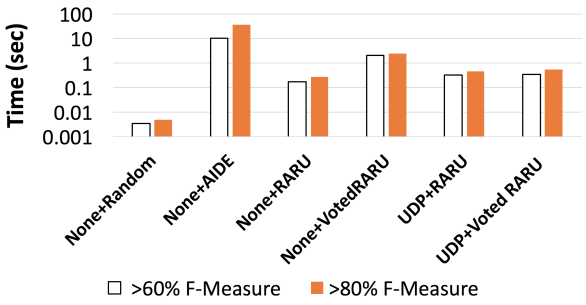

Figure 14: Runtime, 2D, 1 Medium Region

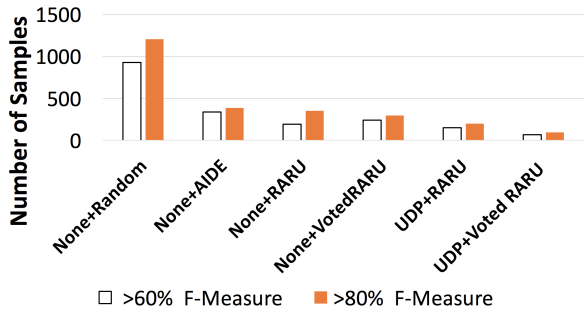

Figure 15: Accuracy, 2D, 5 Large Regions

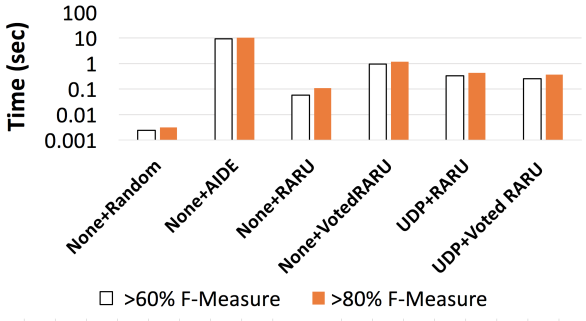

Figure 16: Runtime, 2D, 1 Large Region

80 samples for medium regions, and less than 70 samples for large regions on average. To reach an accuracy of 80%, UDP+VotedRARU requires, on average, 350 samples for small regions, 140 samples for medium regions, and 120 samples for large regions. Further, our UDP+RARU also shown excellent performance and is only slightly behind the UDP+VotedRARU.

Although, None+RARU and None+VotedRARU without using UDP do not achieve the same effect as working with UDP. However, they still demonstrate an acceptable performance due to the efficiency of RARU and are fully usable when facing medium and large regions, as both of them only require less than 800 samples and 370 samples to achieve an accuracy of at least 80% for medium and large regions.

AIDE also showed a good performance for medium and large regions, especially for 60% of accuracy, as in the most case, it requires 400 samples for medium and 330 samples for large regions to achieve an accuracy of 60%. For large regions, it requires less than 360 samples to achieve an accuracy of 80%. But AIDE fails to discover really small regions as it requires 1450-3500 samples to achieve a 60% of accuracy and would require more than 5500 samples to achieve 80% accuracy for small regions. The reason for such behavior is that when discovering very small regions, AIDE would generate a large number of samples due to its costly "zoom-in" operations. Comparing AIDE to our UDP+VotedRARU, our scheme requires 9x-16x fewer samples for small target regions, 5x-10x fewer samples for medium target regions, and 3x-5x fewer samples for large target regions than AIDE.

We also compare all algorithms with Random, which is a baseline algorithm that randomly (based on uniform distribution) selects samples from the exploration space, presents them to the user for feedback, and then builds a decision tree classifier based on these samples. The result shows that Random fails to discover small regions. Even when we increase the number of samples to 9000, it still fails to reach an accuracy of 60%. Random can discover medium and large regions but with a great cost, such that it requires $2000-3000$ samples for medium regions and $1200-1800$ samples for a large region to reach an accuracy of 80%.

Runtime Comparison Figures 12-16 shown the runtime (in logarithmic scale) of each algorithm to reach an accuracy of at least 60% and 80%. In all cases, a smaller target region would increase the runtime. Naturally, random is overall the fastest scheme as it requires almost no computation. Other than random, the runtime is acceptable for all of our schemes, such that for UDP+RARU and UDP+VotedRARU the runtime to achieve 80% of accuracy is only 1-2 seconds for small regions and less than 0.5 seconds for both medium and large region, which is expected as our schemes only generate a small number of query samples.

None+RARU, None+VotedRARU, and AIDE would require higher runtime than UDP+RARU and UDP+VotedRARU as they require more samples to reach 80% accuracy. But the runtime for both None+RARU and None+VotedRARU are still acceptable as they require less than 10 seconds

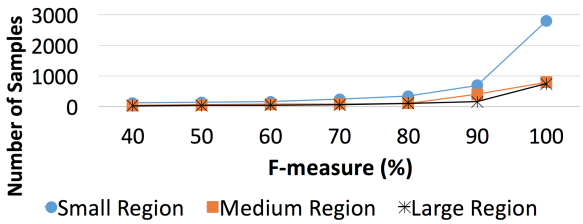

Figure 17: UDP+VotedRARU with Increasing Regions Sizes (1 Region)

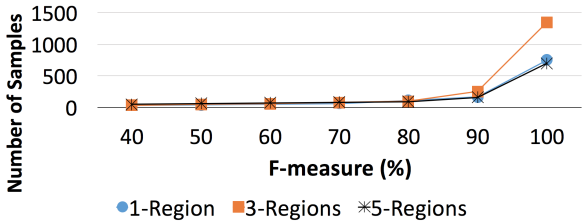

Figure 18: UDP+VotedRARU with Increasing Regions Numbers (Large Region)

for small and less than 1 seconds for medium and large regions to achieve 80% accuracy. Note that although we have implemented AIDE accurately and faithfully, some optimization may be missed. As a result, the runtime may be two times slower than a fully optimized version (according to the runtime reported in [Dimitriadou et al., 2014]). In all runtime experiments, we only report the real runtime that we have measured based on our implementation.

Zoom into the Best Scheme Figures 17-18 shown a closer look at the effectiveness of the best scheme UDP+VotedRARU as we increase the complexity of range query by varying the number of relevant regions and the size of the relevant region. We noticed that UDP+VotedRARU achieves a remarkable performance as it only requires 110 samples in medium and large regions and 350 samples (out of 10×10^5 tuples) for small regions to reach 80% accuracy. Further, the performance of UDP+VotedRARU only decreases slightly when the number of relevant regions is increased.

Traditional Uncertainty VS. RARU We compare our RARU with the traditional Uncertainty Sampling, both strategies employ Naive Bayes Classifier to compute uncertainty. Figures 19-24 show the effectiveness and efficiency of both schemes. In these figures, we used U to denote Uncertainty and VU to denote Voted Uncertainty. These experiments are based on a small dataset that contains only 100k tuples (due to the time complexity of traditional uncertainty sampling) and small target regions. Note that, for Figures 20-24 the runtimes are in logarithmic scale. As expected, the traditional Uncertainty Sampling overall reaches the same level of accuracy with less

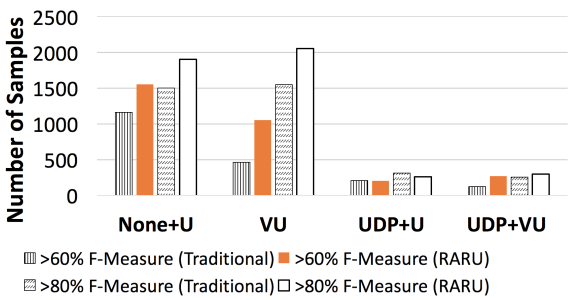

Figure 19: Accuracy,1 Small Region, Small Dataset

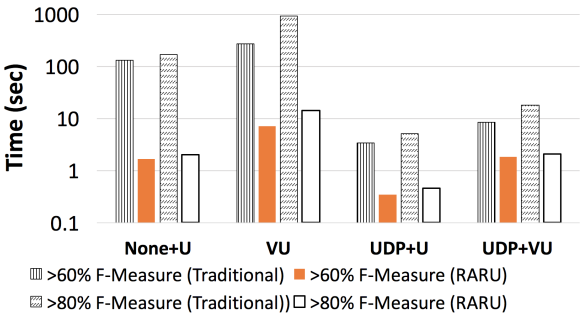

Figure 20: Runtime, 1 Small Region, Small Dataset

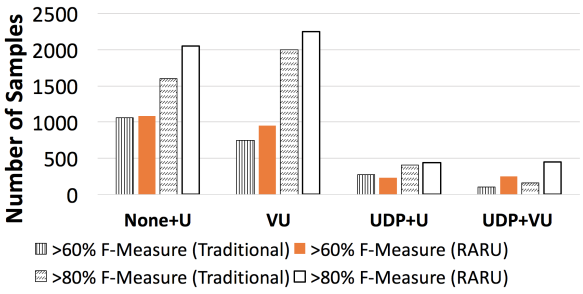

Figure 21: Accuracy, 3 Small Regions, Small Dataset

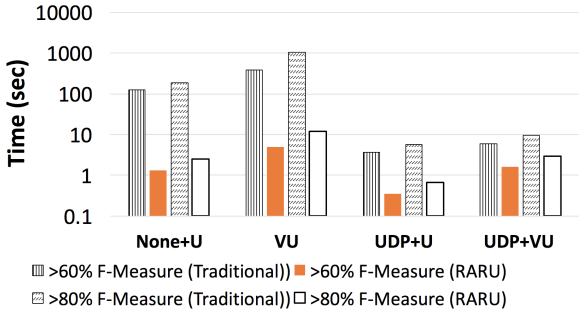

Figure 22: Runtime, 3 Small Regions, Small Dataset

number of samples than RARU. Such that compare to RARU; the traditional uncertainty saves up to 35% of samples when UDP is employed and up to 50% of samples when UDP is not employed. However, traditional Uncertainty Sampling can be up to 60 times slower than RARU when UDP is employed and up to 100 times slower than RARU when UDP is not employed. As the efficiency of traditional Uncertainty Sampling is extremely low, it is still unfeasible to apply it directly on the modern database systems for real-world interactive data exploration tasks.

Impact of the Data Reduction Further, the effectiveness of data reduction techniques is demonstrated with UDP, such that apply UDP as data reduction on average requires 3×-10× fewer

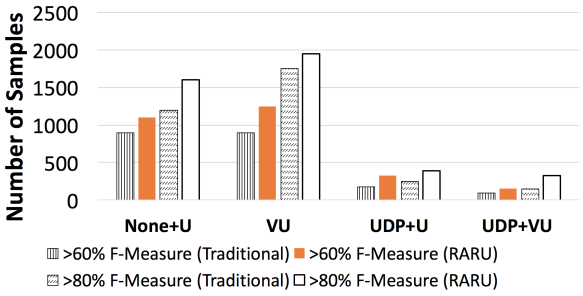

Figure 23: Accuracy, 5 Small Regions, Small Dataset

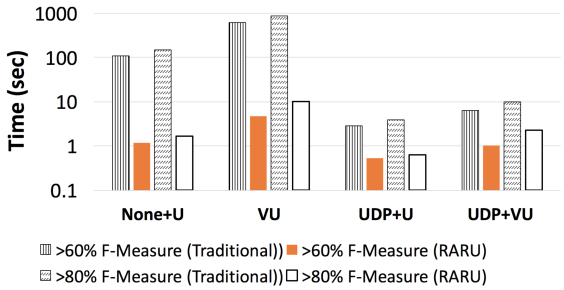

Figure 24: Runtime, 5 Small Regions, Small Dataset

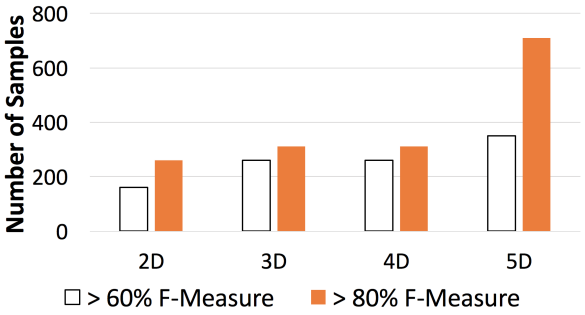

Figure 25: 1 Small Region, F-Measurement of UDP+VotedRARU, Different Dimensions

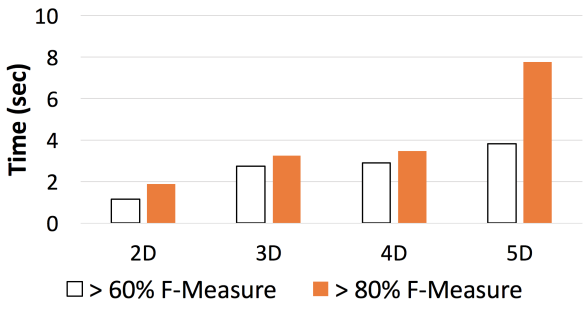

Figure 26: 1 Small Region, Runtime (sec) of UDP+VotedRARU, Different Dimensions

samples than no data reduction for both traditional Uncertainty Sampling and RARU. We also observed the same amount of reduction in runtime (to achieve 60% of accuracy) when UDP is employed, which is as expected since both the data space and the number of samples generated are reduced.

Dimensionality Figures 25-26 demonstrate the effectiveness and efficiency of UDP+VotedRARU as we increase the dimensionality of our exploration space from 2-D to 5-D with one small target region. Our target range query has conjunctions on two attributes. Our solution has correctly identified the two attributes that define the target region, thus, able to discard

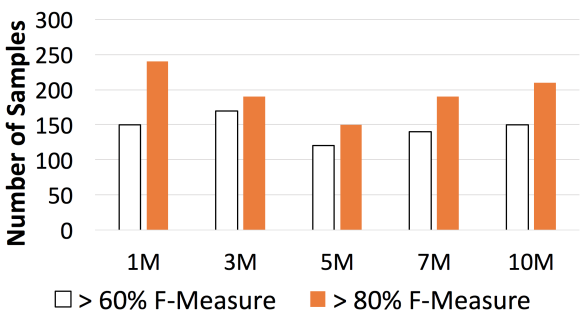

Figure 27: 1 Small Region, F-Measurement of UDP+VotedRARU, Different Dataset Sizes

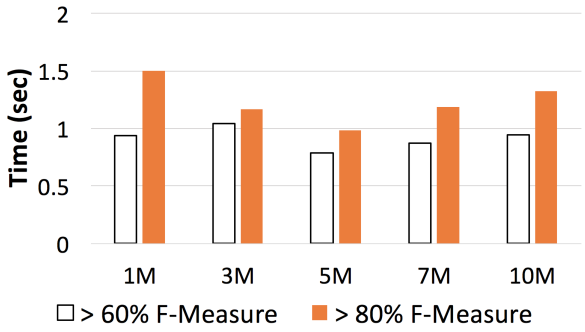

Figure 28: 1 Small Region, Runtime (sec) of UDP+VotedRARU, Different Dataset Sizes

not related attributes from the decision tree to obtain the target range query. As expected, high dimensions would require more samples to reach the same accuracy as low dimensions. However, the number of samples only increased slightly from 2D to 3D and 4D. Even for 5D, the number of samples needs to achieve 60% of accuracy only increased 35% compare to 4D, and the number of samples needed for 80% is still within 700 samples even for 5-dimensional space.

Database Size Figures 27-28, illustrates the scalability of UDP+RARU as we increasing the dataset size from 1 millions to 10 millions. The result has shown that our method is highly scalable as both the effectiveness and efficiency are independent of the size of the data set. This is because in RARU the time taken to generate one sample to present to the user is dependent on the distribution of the uncertainty score of all objects in the dataset and is independent of the size of the dataset.

3.5 Summary

Motivated by the challenge of reducing human effect in exploring large datasets, in this chapter, we discussed REQUEST, a novel framework for the example-driven style of data exploration. The REQUEST framework consists of two key components, namely, data reduction and query selection.

We show the applicability of REQUEST by proposing an efficient user-guided data reduction technique with Multi-Instance Active Learning (MIAL) and a novel query selection method, called Randomized Accept/Reject Uncertainty (RARU), that aims to provide high scalability. Specifically, we designed and experimented with four schemes: None+RARU, None+VotedRARU, MIAL+RARU, and MIAL+VotedRARU as solutions to the example-driven data exploration.

Our experimental results have shown that our proposed schemes achieve much higher performance in both effectiveness and efficiency when compared to the state-of-the-art. Further, the human efforts incurred in providing feedback were reduced by up to 93%.

4.0 Towards Scalable Interactive Data Exploration

The work covered in this chapter was published in the 2021 Proceedings of International Conference on Extending Database Technology (EDBT) [Ge and Chrysanthis, 2021].

In the previous chapter we presented our first contribution, the REQUEST framework. In REQUEST (Section 3.2), the proposed query selection strategy leverages the technique of active learning to effectively and efficiently steer the exploration process. In this chapter, we discuss our solution to address the scalability issue that is critical to the *active learning-based* Example-driven Exploration through a rather traditional strategy: *indexing*.

4.1 Introduction

As discussed in Section 2.3, numerous active learning techniques [Settles, 2009, Lewis and Gale, 1994, Seung et al., 1992, Cai et al., 2013, Zhang et al., 2017, Freytag et al., 2014] have been proposed to boost the convergence of training a predictive model. Among these query strategies, *uncertainty sampling* is the most commonly used one because of its simplicity and efficiency, as pointed out in [Settles, 2009]. Uncertainty sampling trains each predictive model in an iterative fashion, where in each iteration, it identifies the unlabeled items that are closest to the current decision boundary of the predictive model as these items are believed to be most *uncertain*. Uncertainty Sampling then solicits the user's label on the identified sample and utilizes it in the training of the predictive model.

Due to the above-mentioned benefits of uncertainty sampling, we have adapted it in our solutions to reduce users' effort and enable desired effectiveness and efficiency for interactive Example-driven Explorations. However, despite being more efficient than alternative active learning methods, in order to find the most uncertain object, uncertainty sampling still needs to perform an exhaustive search over the entire database. Therefore, in the case where the size of the data is larger than the main memory capacity, those additional data that resides on the secondary storage must be loaded into memory at each iteration. Due to the limitation imposed by physical I/O of

the secondary storages, it will take a significant amount of time for active learning techniques to scan datasets that are considerably larger than the main memory capacity. This essentially makes it impossible for any active learning-based Example-driven Exploration system to explore datasets that are larger than the available memory. As pointed in [Liu and Heer, 2014], run time efficiency is critical for any human-in-the-loop interactive systems as a response time excesses 500ms will severely impact the user's engagement and hence hinders the usability of the system. Moreover, as the exploration could occur on any subset of the attributes of the dataset, it is nearly impossible to apply any typical indexing in advance to support the exploration task over any arbitrary combination of the attributes.

In order to achieve interactiveness, existing Example-driven Exploration systems rely on main memory to cache the entire dataset, otherwise, the exploration will become extremely slow and unusable from the user's perspective. For datasets that are larger than the main memory, a subset of data objects will need to be sampled from the original dataset on secondary storage. While simple and intuitive, this approach could easily lead to very inaccurate results and a waste of user effort since the boundaries of the interesting data regions in the sampled space is likely to be different from the original space [Ge et al., 2016b]. Moreover, small sets of relevant data regions may even be ignored in the resulting sample set.

To overcome this problem, we propose a novel indexing mechanism coined *Uncertainty Estimation Index* (UEI) to facilitate the interactivity and scalability of active learning-based Example-driven Exploration systems (Section 4.2). Instead of relying solely on the main memory to cache the whole dataset during the exploration, UEI caches in memory only the necessary parts of the data that are needed by the current stage of the exploration. This is achieved through the combination of an effective estimation of the uncertainty of each data object and an efficient data storage mechanism. The essential observation that UEI is based on is that data objects often have additional information that can be used to infer their relationship with other objects, one of which is the *similarity* among data objects. Since the uncertainty essentially represents its distance to the current decision boundary in the high-dimensional data space, thus the uncertainty of an object

x is strongly related to the uncertainty of the surrounding objects [Lewis and Gale, 1994]. This observation allows UEI to informatively select the set of highly uncertain data objects to be loaded into memory before it is needed by the predictive model. Hence, the amount of memory needed to explore a dataset in real-time has been significantly reduced.

To evaluate UEI, we employed our data exploration system REQUEST (Chapter 3), and compare the performance of UEI against MySQL, which is commonly used by the existing interactive data exploration systems [Dimitriadou et al., 2016, Huang et al., 2019] (Section 4.3). Our results on large real-world dataset Sloan Digital Sky Survey (SDSS) [sds, 2021], show that UEI outperforms existing solutions by more than 50X in runtime efficiency when the size of the dataset goes beyond the main memory capacity, and is well capable of meeting the sub 500 millisecond interactive response time requirement for data that are at least 100 times larger than the main memory capacity while achieving minimum impact on the convergence of the predictive model. Furthermore, we have provided a discussion on the the wide applicability of UEI beyond interactive data exploration (Section 4.4).

4.2 Uncertainty Estimation Index

Maintaining interactive response time for large datasets that are beyond the main memory capacity has always been one of the major challenges of active learning-based interactive data exploration systems (e.g., Example-driven Exploration systems). In tackling this problem, we focused our efforts on uncertainty sampling and proposed a novel index approach coined *Uncertainty Estimation Index* (`UEI`). In the next section (Section 4.2.1) we provide the details of `UEI`'s main component, and discuss how `UEI` leverages the inverted index-based data secondary storages to support scalable exploration. Following that, in Section 4.2.2, we provide a walkthrough of a complete example to illustrate how a typical active learning-based interactive data exploration workflow performs when enhanced with `UEI`.

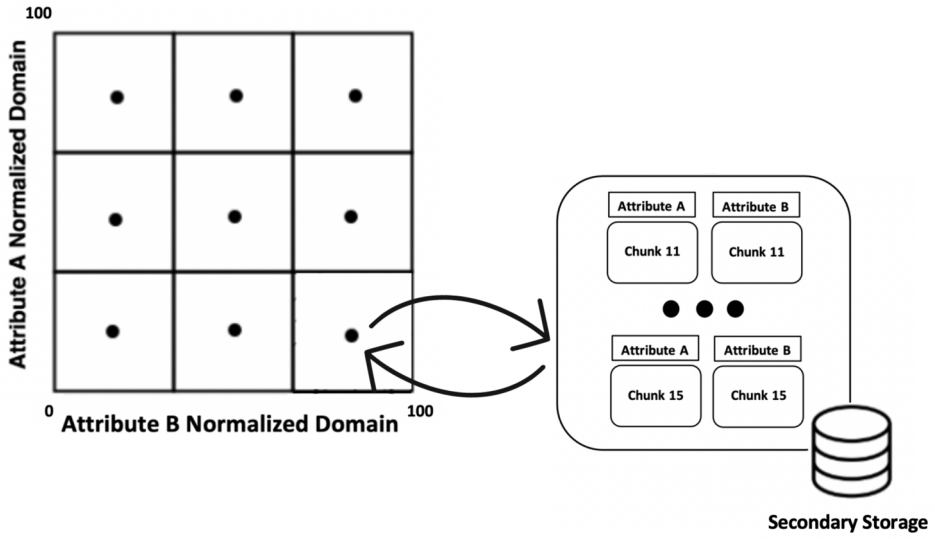

Figure 29: Illustrates UEI With a 2D Data Space, Each Grid Represents a Subspace, the Dot in the Center of Each Grid Represents a Symbolic Point p. With Chunks Stored As Separated Files on the Secondary Storage.

4.2.1 UEI Components

A key observation underlying `UEI` is that data objects often have additional information that can be used to inference their relationship with other objects, one of which is the similarity (i.e., distance) between data objects [Lewis and Gale, 1994]. In other words, the uncertainty value of an object x is strongly related to the uncertainty of the surrounding objects. For example, if x is located near to one of the current decision boundaries, then x would have higher uncertainty due to a mixed set of relevant and irrelevant neighbors. More precisely, we observed that the uncertainty of a data object x can be approximated through its spatial relationships with the labeled data objects. `UEI` explores this spatial relationship to load into memory the set of highly uncertain data objects before they are needed by the query strategy.

Specifically, as illustrated in Figure 29, the main idea of `UEI` is to divide the exploration space D into equal-size subspaces (i.e., d-dimensional grids) g_i's of D ($g_i \in D$), and build a set of

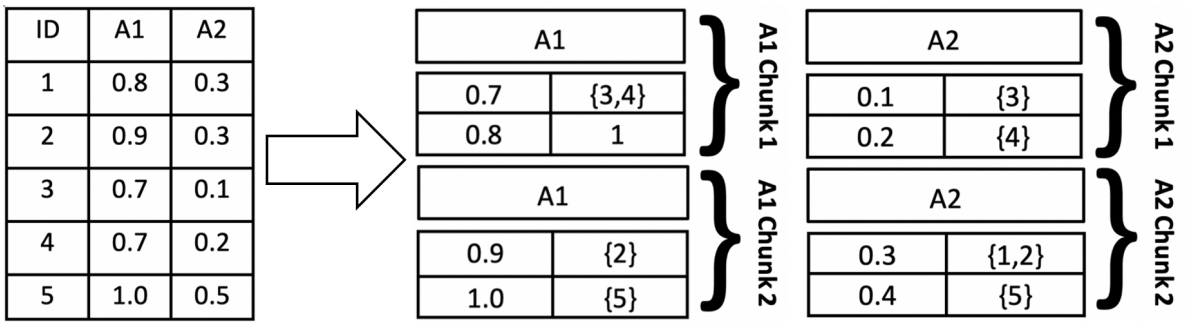

Figure 30: Before Storing the Data, UEI Vertically Decompose the Data Into an Inverted Index Form, and Then Store Them in Separate Chunks.

symbolic (virtual) index points $P = \{p_1, ...p_c\}$, such that each index point p_i represents a subspace g_i. In each iteration, UEI estimates the uncertainty of each subspace based on the uncertainty of its corresponding index point p, then loads only the data in the subspace that is predicted to be most uncertain into memory.

To do so, UEI comprises five components: 1) an index set P of symbolic index points p_i; 2) a mapping method $m : p \longmapsto C$ that maps each index point p to a set of data chunk C; 3) a data cache U that caches a subset of uniformly sampled unlabeled data; 4) a set of labeled data L that contains all data that has been labeled by the user, and 5) the exploration dataset D stored in a fully inverted columnar format on a hard drive. The first four components reside in the main memory.

UEI divides the operation of exploration into two phases: an *Index Initialization* phase, and *Interactive Exploration* phase. The first phase only needs to be executed once per each new dataset. The second phase is specific to each exploration and discussed in the next section (Section 4.2.2).

Index Initialization Phase As illustrated in Algorithm 3, to work with a new dataset, UEI first vertically (i.e., attribute-wise) decompose the whole data set and sort each dimension (i.e., attribute) based on the values in ascending order (lines 2 - 4). Since each dimension of a typical exploratory dataset (e.g., scientific, business analysis) can contain values of an arbitrary length, and one specific value for a dimension may appear multiple times, we compress the data by organizing

Algorithm 3 Typical Exploration Flow with UEI

Require: The raw data set D
Ensure: Result set T
1: $P \leftarrow \emptyset, L \leftarrow \emptyset, U \leftarrow \emptyset$
2: $DC \leftarrow verticalDecompose(D)$
3: **for** $d = 1$ to $|DC|$ **do**
4: $sort(DC_d)$
5: $C \leftarrow splitIntoChunks(DC_d)$
6: **end for**
7: $G \leftarrow splitIntoSubspaces(D)$
8: **for** each grid $g_i \in G$ **do**
9: $p_i \leftarrow computeCenter(g_i)$
10: $P \leftarrow P \cup \{p_i\}$
11: **end for**
12: $U \leftarrow sample(D, \gamma)$
13: $M \leftarrow$ initialize predictive model for uncertainty estimation
14: **while** user continues the exploration **do**
15: drop any previously loaded data regions from U
16: $M \leftarrow$ trained with L to update M
17: $P \leftarrow updateUncertainty(P, M)$
18: $p_i^* \leftarrow$ choose the most uncertainty index point from P
19: $g_i^* \leftarrow$ load data region with $m(p_i^*)$
20: $U \leftarrow U \cup g_i$
21: choose one x from U using M
22: solicit user's label on x
23: $L \leftarrow L \cup \{x\}$
24: $U \leftarrow U - \{x\}$
25: **end while**
26: $T \leftarrow resultRetrieval(L)$
27: Return the set of interesting data objects T

it in a key-value fashion ($< key, \{values\} >$), where each value of the dimension would be used as a key and the ids of the corresponding objects as $values$ (as illustrated in Figure 30).

During the process of storing the data, `UEI` splits the distinct values of each dimension d into a set of equal-sized data chunks $C^d = \{c^d{}_i, ..., c^d{}_u\}$, where each chunk will be stored as a separate file on the disk, and the size of each chunk can be adjusted based on the size of the data and the available hardware resources (line 5). `UEI` also ensures that the values of each dimension are

stored in sequential order, meaning values stored in each subsequence chunk ${c^d}_{i+1}$ will be larger than the values that have been stored in ${c^d}_i$ for efficient lookup.

Once the data are partitioned and stored on the disk, UEI would start construct the set of symbolic index points P by divide the original data space D into a set of equilateral d-dimensional subspaces $G = \{g_i, ..., g_j\}$, where $|G| = |P|$, then for each subspace g_i, UEI constructs a symbolic index point p_i that represents g_i by using the coordinates of the "virtual" center point of g_i (lines 7-11). To ensure UEI can be deployed in resource restricted environments, the number of symbolic index points can be adjusted based on the size of the dataset and the available hardware resources.

In order to construct and load each subspace g_i into memory, UEI employed a hash-based mapping method m that records for each symbolic index point p_i, the set of chunks that are needed to construct g_i. As chunks are stored separately on the disk, this approach allows UEI to quickly identify the data that needs to be loaded. Since each data subspace g is stored as series of one-dimensional data chunks, to reconstruct each g when needed, UEI utilizes a hash table for efficiently merge of those data chunks. During the merge process, UEI iterates through each dimension and loads the corresponding chunks to the memory one at a time, and each entry in the chunk would be visited in a sequential manner. For each object ID that is recorded in a loaded data chunk c_i, the value associated with the ID will be inserted into the corresponding entry in the hash table. Once a chunk has been examined, UEI will release the memory space used to hold the data chunk and reuse the space for the subsequent chunk.

4.2.2 UEI in Action

In the previous section, we have discussed the components of UEI, as well as how the data are being stored and indexed, which essentially covers the first half (i.e., lines 1 - 11) of the exploration workflow, shown in Algorithm 3. In this section, we will discuss the interactive exploration phase of UEI, which illustrates how a typical active learning-based interactive data exploration task can be performed when incorporating UEI (lines 12 - 27, Algorithm 3).

Interactive Exploration Phase After the index set has been constructed, UEI begins the ex-

ploration by filling the unlabeled set U. Specifically, for the original data space D, UEI would uniformly sample a set of data from the underlying dataset (line 12), where the size of the samples γ can be adjusted based on the system hardware specs (e.g., available main memory size). As a result, a set of unlabeled objects U would be sampled and cached in the main memory. These unlabeled objects will then be used in the acquisition of the set of initial examples that will be labeled by the user to construct the initial predictive model M_0 for uncertainty estimation. Query strategy will randomly sample examples from U until the set of initial examples contains at least one positive example and one negative example (line 13).

In each iteration, UEI updates the uncertainty of all index points $p_i \in P$ based on the most recently trained predictive model M_{t-1} (line 17), which serves as the uncertainty estimator. Here the uncertainty of a data object essentially equals to the probability of one object being either positive or negative class, with a value that equal to 50% being the most uncertain. Then, the index point p_i^* for which the current exploration model is most uncertain, will be chosen (line 18), such that:

$$p_i^* = \underset{p_i \in P}{\operatorname{argmax}}\, M_{t-1}(Y|p_i) \tag{4.1}$$

where Y = {0,1} is set of binary labels.

Based on the chosen p_i^*, UEI uses the mapping method m to identify and load (into the memory) all data chunks that correspond to the subspace g_i^*, which was represented by p_i^* (line 19). As mentioned earlier, the mapping method m is simply a hash table that maps a single index point p into a set of data chunks located on the disk. Later, the data of subspace g_i^* together with the unlabeled dataset U will be used by the query strategy (i.e., uncertainty sampling) in the selection of the example to be labeled in the current iteration (lines 20 - 22). To reduce memory usage, by default UEI kept only one uncertain data region g_i^* in the memory at any given time. Once the user is satisfied with the exploration result, the *resultRetrieval* method will be invoked to retrieve the exploration results and present them to the user (lines 26 - 27).

Tuning Interactive Exploration In addition to the above typical exploration flow, UEI further allows the user to specify a response latency threshold σ that determines the latency between each

exploration iteration (i.e., two subsequent examples). Using the user-specified σ, UEI determines whether or not to defer the swap between the current in-memory uncertain region g_i^* and the next uncertain region g_{i+1}^*, when g_i^* is no longer the most uncertain region.

In the case when an extremely low σ is specified that makes it impossible for the system to load the entire subspace g_i^* into main memory, UEI would start fetching the corresponding data chunks that associated with g_{i+1}^* (in the background) θ iterations before g_{i+1}^* is loaded into the memory. Here, θ is a tunable variable that can also be inferred based on the average loading time τ of data regions, and the configurable latency threshold σ, such that $\theta = \lceil \frac{\tau}{\sigma} \rceil$.

4.2.3 Time Complexity of UEI

Clearly, the time complexity of UEI is dominated by the interactive exploration phase. As discussed in Section 4.2.1, the initialization phase is done once for each dataset and the time required for UEI to prepare and store a dataset D on secondary storage is simply *linear* with respect to the number items n stored in D. s As discussed in Section 4.2.2, each iteration of the interactive exploration phase in UEI is dominated by the time taken to load the data from the chunks stored on the disk into the memory, which is *linear* with respect to the number of dimensions k and the number entries e stored in the loaded chunks. In contrast, each iteration in the current interactive data exploration approaches needs to load and examine all data items n ($e <<< n$).

Therefore, the time complexity of the UEI-enhanced data exploration generally is reduced from $O(kn)$ where n is the number of data objects to $O(ke)$ where e is the number of entries associated with the loaded chunks for the current most uncertain subspace g_i^*.

4.3 Experimental Evaluation

In our experimental evaluation of our UEI, we use our REQUEST prototype as introduced in previous chapter 3, with two schemes, one incorporating UEI, and one utilizing a commonly used

relational DBMS, namely, MySQL. After describing our experimental setup in Section 4.3.1, we present the findings of our experimental evaluation in Section 4.3.2.

4.3.1 Experiment Setup

The experimental setup was similar to the one used to evaluate REQUEST (Section 3.4). Below we will discuss the details of our experimental setup.

Dataset We used 40 GB of real-world datasets from Sloan Digital Sky Survey (SDSS) [sds, 2021] that consists of 10×10^6 tuples.

Interactive Data Exploration System In our experiments, we employed our REQUEST framework with traditional uncertainty sampling and the *dual weighted k-nearest neighbor* (DWKNN) [Gou et al., 2012] as the uncertainty estimator.

Environment All the experiments were run on a machine with Intel Core i7 Processor, 32 GiB RAM, and 2 TB of NVMe SSD. The fast NVMe SSD was used to eliminate any potential bottleneck due to physical IO limitations. All experiments reported are averages of 10 complete runs. We have considered five numerical attributes *rowc, colc, ra, dec* and *field* of the PhotoObjAll table.

Target Interest Regions The exploration task characterizes user interests and eventually predicts the relevant regions by iteratively gathering user labeled tuples. We experimented with 1 region per each exploration task. In addition, we vary the single region complexity based on the data space coverage of the relevant regions. Specifically, we categorize relevant regions to *small*, *medium*, and *large*. Small regions have cardinality with an average of 0.1% of the entire experimental dataset, medium regions a cardinality of 0.4%, and large regions a cardinality of 0.8%. Furthermore, the dimensionality of the target interest regions is the same as the dimensionality of the dataset across the entire experiment.

User Simulation For experiment evaluation purposes, we simulate the user behavior using the following method. For each target interest region, we simulate the user by executing the corresponding range query to collect the exact target set of relevant tuples. We rely on this "oracle" set

Table 2: Experimental Parameters of UEI

Number of runs per result	10
Number of dimensions (D)	5
Number of relevant regions	1
Cardinality of relevant regions	0.1% (S), 0.4% (M), 0.8% (L)
Uncertainty Estimator	DWKNN [Gou et al., 2012]
Label Type	Binary
Data Storage Engine	UEI, MySQL
Size of Individual Data Chunk	470KB
Number of Symbolic Index Points	3125
Latency Threshold	500ms
Performance Measurement	F-Measure (Accuracy)

to assign confidence scores p to the tuples we extract in each iteration based on their location in the data space against the target region.

More specifically, for each relevant region, there is a region center and a set of region widths, one for each dimension. We define the *maximum relative distance* d of an example against the region center as:

$$d = max_{i=1..l}(|x_i - c_i|/w_i) \tag{4.2}$$

where l is the dimension number, $|\cdot|$ is the absolute value operator, x_i, c_i and w_i are the attribute value of the example, of the center and region width in each dimension.

Parameters Table 2 summarizes the important settings in the experiments.

4.3.2 Experiment Results

To test `UEI` ability to provide an interactive response time for datasets that are beyond the size of main memory capacity, in our experiments, we stored 10 million data objects with both `UEI` and MySQL, and restricted the memory footprint for both `UEI` and MySQL to be within 400MB, which is approximately 1% of the entire dataset. Figures 31 to 34 illustrates the effectiveness and

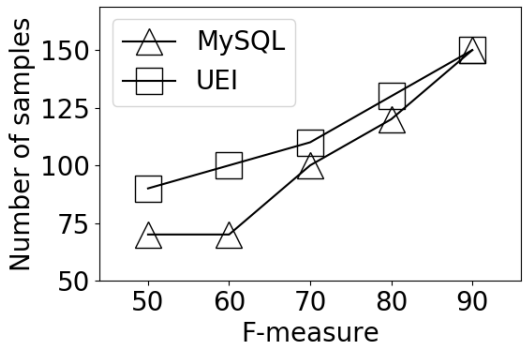

Figure 31: UEI vs. MySQL Accuracy (Small Target Region).

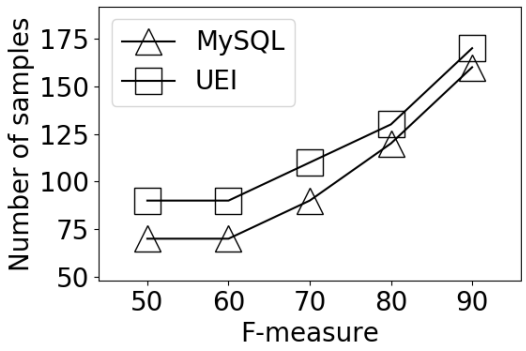

Figure 32: UEI vs. MySQL Accuracy (Medium Target Region).

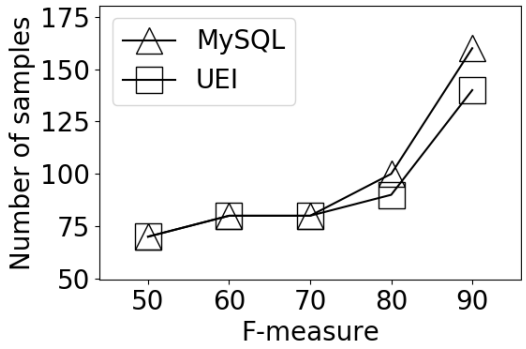

Figure 33: UEI vs. MySQL Accuracy (Large Target Region).

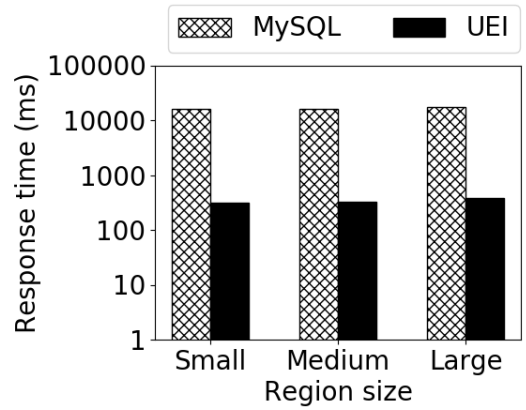

Figure 34: UEI vs. MySQL Response Time.

efficiency of our proposed indexing method `UEI`.

UEI Accuracy (Figures 31 to 33) Compared to MySQL, we have noticed that our proposed `UEI` requires more labeled examples in the early stage of the experiment (e.g., below 70% of accuracy). This is due to the fact that in the early stage, the classifier does not have enough training samples to learn an accurate uncertainty estimator (i.e., DWKNN classifier) that captures precisely the user's interesting regions. This causes the predictive model to select less informative examples

to be labeled by the user. Since UEI uses uncertainty as the criteria for both the example selection and the loading of data regions (i.e., subspaces), therefore the negative effect of an inaccurate classifier has been magnified.

However, as the accuracy of the uncertainty estimator improves with more labeled examples, we observed a significant boost in performance of UEI in the later stages (e.g., above 80% accuracy) of the exploration. This is expected as the the predictive model gets more and more accurate with respect to the decision boundaries, and thus can estimate more accurate uncertainties. Since UEI rely on the uncertainty estimation for both the query strategy and cache management, therefore, it benefits more noticeably than the MySQL, which only rely on uncertainty estimation for query strategy.

UEI Response Time (Figure 34) Finally, we have measured the response time for both UEI and MySQL based schemes. As shown in Figure 34, UEI achieves $50x$ faster response time than MySQL, and ensures the sub-second interactive response time across all data region sizes. Note the response time remains the same across all three target interest regions sizes, which is as expected because the runtime complexity of the uncertainty sampling-based systems only depends on the size of the dataset and not the size of the target interest regions.

From the experiments, it is clear that due to the fact that uncertainty sampling requires an exhaustive search over the entire data spare, thus the physical bandwidth of the secondary storage has become the major bottleneck that severely limits the scalability of active learning-based interactive data exploration systems.

Even though in our experiments, we have used NVMe based SSD with I/O throughput of around 3.4GB/s, the uncertainty sampling still takes over 12 seconds to complete the exhaustive search in each iteration. Therefore, it is still impossible to explore datasets that exceed the main memory capacity without UEI.

4.4 Applicability of UEI

Traditionally, indexing has been the core technique for optimizing response time in database systems. Recently, main-memory indexing and specialized access methods have been proposed to support domain-specific query processing and analytics (e.g., [Chatzigeorgakidis et al., 2019, Peng et al., 2020]). `UEI` is based on similar principles as these specialized access methods. However, `UEI`, to the best of our knowledge, is the first domain-specific access method with in-memory and disk components that support interactivity and scalability of active learning-based interactive data exploration systems.

In addition to the active learning-based interactive data exploration systems, `UEI` can also be utilized in other active learning-based *Human-in-the-loop* (HIL) systems. For example, in [Qian et al., 2019], the authors have proposed an active learning-based HIL system, called SystemER, for learning Entity Resolution models through user interactions. By leveraging a human in the loop and active learning, SystemER effectively reduced the number of labeled examples needed to learn high-quality ER models, and thus, reduced human effort. Another example of an active learning-based HIL application is fact-checking. In [Bhattacharjee et al., 2017], the author has proposed an effective active learning-based HIL system for identifying various types of potentially misleading or false information for news contents. Most recently, [Qian et al., 2020] has proposed to use active learning for learning the implicit structured representations of entity names, which can be useful for many entity-related tasks such as entity normalization and variant generation. To facilitate the process of learning such structured representations, a user-friendly interface called PARTNER has been designed to enhance the user's interaction experience. Although the above-mentioned systems have effectively utilized active learning in reducing user efforts, however, similar to active learning-based interactive data exploration systems, they require the data to be cached in main memory. Similar to active learning-based interactive data exploration, our `UEI` can be leverage in the above-mentioned HIL systems to effectively reduce the memory requirement, and in turn, enable these systems to scale out for larger datasets.

4.5 Summary

In this chapter, we present the *Uncertainty Estimation Index* (UEI), the first indexing mechanism that enables active learning-based interactive data exploration systems to explore datasets that exceed the main memory capacity. Instead of requiring all data to be loaded into the main memory as in existing active learning-based interactive data exploration systems for sub-second response times, UEI enables scalability by dynamically identify and caching the set of objects that are most uncertain to the current stage of the exploration. It achieves this by maintaining a small in-memory index that estimates the aggregated uncertainty value of the data items in the entire data subspaces. UEI also employees columnar-based secondary storage and combines it with an inverted index to support efficient loading of the necessary data items at each iteration.

Our experimental evaluation using real-world data show that a state-of-the-art interactive data exploration system using UEI greatly outperform a DBMS-based version of the same interactive data exploration system by providing more than 50x runtime efficiency when the size of the dataset exceeds the main memory capacity and is capable of achieving sub-second response time for data that is 100 times larger than the available memory while achieving the desired exploration accuracy and effectiveness.

Although, UEI is primarily focused on the interactive data exploration systems (e.g., Example-driven Exploration), where response time and run time efficiency is critical, but it can also be used in combination with any active learning-based human-in-the-loop (HIL) applications to achieve significantly higher scalability by removing the main memory restriction.

5.0 Exploring Big Unstructured Data

The work covered in this chapter was published in the Proceedings of 2020 IEEE International Conference on Big Data (IEEE BigData) [Ge et al., 2020].

In this chapter, we discuss our solution that extended the Example-driven Exploration beyond just the structured data. In particular, we discuss how our solutions address the unique challenges associated with exploring the unstructured data given its high-variety and high-dimensional nature.

5.1 Introduction

Despite being effective in retrieving relevant data items, existing Example-driven Exploration systems focus only on structured data with low dimensionality, which does not apply to unstructured data (e.g., image, text, and graph) due to their more complex nature and much higher dimensionality. However, such a paradigm can also be leveraged for searching and exploring unstructured data. For instance, in the case where the user has a mental representation of the characteristics of the COVID-19 articles that suit their interest, but they are not fully acquainted with the actual keywords, author, or topic that would select these interesting articles. Given the fact that a simple keyword of "COVID-19" leads to a deluge of COVID-19 news and articles, mining all or most relevant articles, news, and tweets that satisfy the user's specific interest becomes another "mission impossible". However, if we show the same user an example article or news title that they have seen and ask them "Is this the type of article that you have in mind? Yes or no?", people can answer such ordinal questions with more ease than absolute judgments (i.e., finding the exact words or queries to describe their interest).

The aforementioned scenario is not restricted to searching for articles or news only. Browsing for images, videos, or searching for nodes in large interconnected graphs, users face the same challenge of not being able to accurately describe items that are of interest using traditional search interfaces. However, a user can easily provide a relative judgment based on an actual example, that is, answering questions such as "Do you find this image example interesting: Yes or No?", rather

than describing in detail the characteristics of their intended images.

In this chapter, we describe our framework, coined *ExNav* (**Ex**ploration **Nav**igator), which is the first interactive data exploration framework that leverages Example-driven Exploration paradigm for unstructured data (Section 5.2). ExNav is capable of supporting any form of unstructured data as long as an underlying vector representation of the data can be created. It dynamically refines the exploration space based on a set of interactions with the user, such that in each interaction, the user is presented with an example and is asked to provide only binary feedback (e.g., interesting or not interesting) on each example. Furthermore, it implements a number of optimizations that further improve its exploration efficiency and ensures the desired interactive performance during the exploration.

We experimentally evaluated ExNav on three real-world unstructured datasets (i.e., [tex, 2021, ima, 2021, gra, 2021]), each with different data types (Section 5.3). The experimental results show that ExNav can reduce users' effort by up to 92.3% while still achieving the same accuracy as the state-of-the-art alternative. Moreover, ExNav is also able to identify with high accuracy relevant data items that are often undetectable by current techniques, even when a large number of samples are explored.

5.2 The Exploration Navigator (ExNav)

5.2.1 The ExNav Solution

Our solution ExNav is designed to be independent of the underlying types of the unstructured data and similar to our REQUEST for structured data, ExNav aims at the following goals 1) minimize the user labeling effort during an exploration task, and 2) obtain all data objects relevant to user exploration with the highest accuracy.

As shown in Algorithm 4, ExNav identifies relevant objects through an iterative interaction

Algorithm 4 The Exploration Navigator

Require: The raw data set D
Ensure: The set of relevant objects R
1: Convert D into a set of embeddings E
2: $E_s \leftarrow dataReduction(E)$
3: $L \leftarrow$ obtain initial set of samples
4: Unlabeled set $U \leftarrow E_s$
5: $M \leftarrow$ initialize query selection method
6: **while** user continues the exploration **do**
7: Choose one x from U using M
8: Solicit user's label on x
9: $L \leftarrow L \cup \{x\}$
10: $U \leftarrow U - \{x\}$
11: $M \leftarrow$ trained with L to update M
12: **end while**
13: Return relevant objects R captured by the most recent M

with the user. In each iteration, ExNav presents the user with one *example* (e.g., an image, a news title) from the entire dataset and requests the user's feedback on the relevance of this example to his/her exploration. Inside the system, ExNav decides and utilizes the *data embedding* method based on the data type involved in the exploration to cover each data item into its corresponding feature representations (i.e., embeddings) (Line 1). Leveraging these embeddings, ExNav employs a *predictive model* to wisely select the example to be presented that will maximize the benefit to the exploration task once its relevance with respect to the exploration is provided by the user (Lines 5-7). Once labeled, this example, along with its label (i.e., user feedback) will be incorporated with all previously labeled examples to update the predictive model for better subsequent explorations (Lines 8-11). The user terminates the exploration once they are satisfied with the exploration results, and the set of relevant data objects captured by the most recent predictive model will be returned (Line 13). In the following sections, we will present each main component of ExNav in detail.

5.2.2 Data Embedding

Currently, there exists a large body of embedding algorithms that encompasses various methods for learning of feature representations of different unstructured data types (e.g., image, graph, text) in numerical vector space. Typically, embedding methods aim to create embeddings (i.e., vector representations) based on the assumption that the similarity between data items in their original form should be reflected in the learned feature representations. To achieve this goal, a large variety of algorithms have been proposed for a wide range of data types using both supervised and unsupervised learning methods. For instance, Word2vec [Mikolov et al., 2013], Universal sentence encoder [Yang et al., 2020], and GloVe [Pennington et al., 2014] are some well recognized methods for embedding text into its corresponding vector space. Similarly, for image data, there exists a large body of embedding methods that range from deep learning-based methods (e.g., Resnet-18 and Alexnet) to more traditional keypoint-based embedding methods (e.g., Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF)). Even for more complex data types such as graphs, the problem of embedding each data node into its vector space is also well studied (e.g., [Tang et al., 2015, Grover and Leskovec, 2016]). These embedding methods provide good representations of its data items in the corresponding vector space that preserves the relative similarity between the data items in its original space. In our work, we leveraged these data embedding algorithms and designed ExNav to be a generic interactive data exploration framework that works with any unstructured data as long as an embedding representation can be produced.

Different embedding methods may produce embeddings with different dimensionality, which can range from 128 dimensions to 32768 dimensions [Hong et al., 2017]. Such a large variation in the length of the embeddings can lead to inconsistent performance in the subsequent exploration task and slow convergence of the predictive model. Furthermore, many of these embeddings are loosely embedded, and thus, many of their attributes (i.e., dimensions) do not provide much value in identifying relevant data items. To address this challenge, in ExNav, we use dimensionality reduction algorithms to compress each high dimensional embedding into more condensed lower-dimensional vector representations. In particular, we used a well-known dimension reduction al-

gorithm called *Principal Component Analysis* (PCA), which aims to minimize the reconstruction error between the compressed low dimensional vector and its original high dimensional representation.

5.2.3 Exploration Space Pruning

In a typical active learning setting, query strategies are applied over the entire set of unlabeled data objects to select the next example to be presented to the user for labeling. However, such exhaustive search incurs high processing costs, causing delays between each user-machine interaction, and leads to slower convergence of the predictive model. To address this challenge, data exploration approaches typically employ space pruning techniques for minimizing the number of unlabeled objects to be searched. Extending our idea of User-Driven Pruning (Section 3.2.1.2), in the ExNav framework, we developed simple yet effective space pruning methods, called *Multi-Instance Space pruning* (MIS pruning). Our MIS pruning is inspired by the Multi-Instance Active Learning (MIAL) [Settles et al., 2008], which is a set of popular approaches for reducing user labeling effort.

The idea in MIAL is that data objects can be grouped into a set of bags, and the user is asked to label each bag as positive or negative. Accordingly, MIAL assumes that a bag is negative if every object in that bag is negative; otherwise, the bag is positive. Subsequently, these bag-level labels will be used in the training of either bag-level, or instance-level classifiers [Carbonneau et al., 2019]. Compared to the more traditional instance-level active learning, MIAL has proven to be very effective in further reducing the labeling cost in many domains, and applications [Carbonneau et al., 2019]. For instance, in a text document, image, or biological sequence, consider the case where individual instances tend to form very distinct clusters. In such cases, it may often be easier to label a group of strongly similar objects rather than each individual one, and all objects of the group inherit this label. Previous works have shown that MIAL efficiently reduces annotation costs in various applications such as object detection [Ren et al., 2016], web search results [Zhu et al., 2015],and medical analysis [Quellec et al., 2017], as well as different data types

such as text [Ray and Craven, 2005, Zhang et al., 2013], image [Quellec et al., 2017], and audio [Briggs et al., 2012].

In contrast to MIAL that uses the bag-level labels for the training of classifiers, our MIS pruning uses the bag-level labels to identify and prune sub exploration spaces that are completely irrelevant to the exploration task, and in turn, reduces the amount of data that needs to be searched during the exploration. In particular, MIS pruning first uses the popular *K-Means* clustering algorithm to divide the items in the current exploration space into a set of bags based on their similarity. Given that each bag is created by grouping similar items, it ensures that the items within each bag are much more similar to each other compared to items across different bags. Thus, users only need to quickly skim through the items in each bag to see whether the bag is far away from the regions that contain relevant objects without needing to closely examine each object of the bag. Such pruning technique enables us to leverage the benefit of MIAL and bag-level labels while still preserving the high precision capability of instance-level active learning in identifying precisely all relevant regions.

5.2.4 Query Strategy

Query Strategy is the component of our ExNav framework that aims to minimize the labeling effort while maximizing the accuracy in discovering relevant objects. Recall as in Section 3.2.2, a "query" refers to the process of selecting an example object to be presented to the user for labeling. In ExNav, we leverage uncertainty sampling to quickly learn the interest of the user and steer them towards all relevant data regions.

5.2.4.1 Uncertainty Sampling As previously mentioned in Section 2.3, uncertainty sampling is a popular active learning technique, which aims to choose the data points which are most beneficial to build a classification model that precisely distinguishes relevant and irrelevant data objects. According to Equation 2.2, to measure the uncertainty of a data object x, we need a probabilistic-based predictive model that would report the probability of x being positive or negative. Based on

empirical studies, we selected a probabilistic-based classification model, called *Gradient Boosting Decision Tree Classifier*, [Ke et al., 2017] to determine the uncertainty score.

5.2.4.2 Gradient Boosting Trees Gradient boosting [Friedman, 2001] is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models. Gradient boosting is typically used with decision trees, and the trees are trained in a sequential fashion, such that the subsequent tree is trained towards the residual (i.e., the difference between the observed value and the predicted value) of the current tree. For a binary classification problem, the observed value is 1.0 for positive (relevant) data points and 0.0 for the negative (irrelevant) data points. And the predicted value is the predicted relevance probability of the data point.

For example, to build the m^{th} tree, the algorithm would first calculate the residuals from the $(m-1)^{th}$ tree using:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n. \tag{5.1}$$

where $L(y_i, F(x_i))$ is the loss function and $F_{m-1}(x)$ is the output from the $(m-1)^{th}$ tree in log-odds form. Next, the algorithm would fit a new decision tree (i.e., the m^{th} tree) to the residuals r_{im}, which will have J_m terminal regions (i.e., leaves), $R_{1m}, \ldots, R_{J_m m}$. Then, the algorithm would calculate an output for each region using:

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma) \tag{5.2}$$

where x_i's are the data points, y_i is the observed value for x_i, and $F_{m-1}(x_i)$ is the predicted value for x_i in log-odds form from the $(m-1)^{th}$ tree.

Finally, the output of the m^{th} tree would be:

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} \mathbf{1}_{R_{jm}}(x) \tag{5.3}$$

where ν is the learning rate to prevent over-fitting, and the summation is in place in case a data point ends up in multiple leaves.

Equation 5.3 is also used for the prediction of new data points, and the output is converted to probability using the sigmoid function. The predicted probability will, in turn, be used in the uncertainty calculation for the data point.

5.2.4.3 Randomized Accept/Reject Uncertainty (RARU) As mentioned in Section 3.2, to address the low scalability drawback of uncertainty sampling, we employed an accept/reject query strategy approach, called *Randomized Accept/Reject Uncertainty* (RARU). Particularly, in each iteration, RARU chooses the example to be presented to the user by randomly picking unlabeled objects from the entire set of unlabeled data, and then for each picked object x, RARU calculates the uncertainty score of x. Subsequently, RARU uses the uncertainty score of x as the way to determine its acceptance (i.e., whether to request user's feedback on x), such that the probability of an unlabeled data sample **x** being accepted under RARU is:

$$p(x \text{ is selected}) = \min_{k \in 0,1} \frac{Pr(C_k|\mathbf{x})}{0.5} \tag{5.4}$$

where $Pr(C_k|\mathbf{x})$ is the probability of **x** being assigned a binary class label C_k, and 0.5 is a normalizing factor since a prediction score of 0.5 indicates the classifier is most uncertain about an object.

RARU randomly visits each unlabeled object until one object is accepted according to the above Equation 5.4, and the accepted object (i.e., example) will then be presented to the user for labeling. Clearly, RARU provides an early termination to the exhaustive search of the unlabeled data objects while still preserving the feature of randomized uncertainty, which alleviates the issue of the shortsightedness of the traditional uncertainty sampling.

5.3 Experimental Evaluation

In this section, we will present the results of our experiments. We begin this section by introducing the experimental setup. Then we demonstrate the performance of our schemes and of other alternatives, in terms of accuracy, response time, and scalability.

5.3.1 Experiment Setup

Datasets In our experiments, we used three real-world unstructured datasets. The CBC news dataset (Text) contains 6786 news articles about COVID-19 [tex, 2021]. The Caltech-256 image dataset (Image) contains 30607 images of 256 different categories [ima, 2021]. The Mashup PPI dataset (Graph) contains information of 16,143 nodes and 300181 edges in a protein-protein interaction graph [gra, 2021].

Learning Representation We used off-the-shelf algorithms to generate the learning representations for the data.

- For the Text dataset, we used the Universal Sentence Encoder [Yang et al., 2020] to generate a 512-dimensional vector for each article.
- For the Image dataset, we used the ResNet [He et al., 2016] to generate a 512-dimensional vector for each image.
- For the Graph dataset, we used the pre-trained 500-dimensional vector from the Mashup project [Cho et al., 2016] for each node.

Each vector is a compact representation of a node that accurately captures the topological patterns of the corresponding node in the original graph. All vectors in each dataset are projected to a 32-dimensional space using PCA as the final learning representation for that dataset and are used across all models.

Schemes We experimented with one baseline scheme, one state-of-the-art scheme, and four ExNav schemes. The baseline scheme is the random scheme (RANDOM), where the system se-

Table 3: Experimental Parameters of ExNav

Number of data objects	6786 (All datasets)
Number of dimensions	32
Number of target relevent regions	1, 2, 3
Pruning question count	16
Example batch size	5
Max example allowed	3000
Target region cardinality	0.5% (S), 1.0% (M), 2.0% (L)
Considered Query Strategies	Traditional, RARU
Predictive Model	GBDT
Performance measure	F-Measure (Accuracy)
Schemes	RANDOM, REQUEST, ExNav_TU, ExNav_RARU, ExNav_TU_P, ExNav_RARU_P
Number of runs per result	10

lects examples randomly (based on uniform distribution) from the unlabeled set. The REQUEST scheme with all the default settings as described in Section 3.4.

- ExNav_TU is the the first scheme for ExNav. It uses traditional uncertainty sampling [Settles, 2009] to select examples.
- ExNav_RARU is the second scheme for ExNav. It uses RARU as the query strategy for example selection.
- ExNav_TU_P is the third scheme for ExNav that denote ExNav_TU with MIS pruning.
- ExNav_RARU_P is the forth scheme for ExNav that denote ExNav_RARU with MIS pruning.

Target Interest Regions The exploration task characterizes user interests and eventually predicts the interest regions by iteratively gathering user-labeled tuples. As mentioned before, we focus on predicting the user interest regions. Particularly, in our experiments, we generate each target interest region (i.e., relevant region) with random range queries and experimented with three different interest region amounts $\{1, 2, 3\}$. Further, we vary the single region complexity based on the data space coverage of the relevant regions. Specifically, we categorize relevant regions as

small, medium, and large. Small regions have cardinality with an average of 0.5% of the entire experimental dataset, medium regions have a cardinality of 1.0%, and large regions have a cardinality of 2.0%.

User Simulation Given the target relevant regions, we simulate the user by collecting the exact target set of relevant tuples in the region. We rely on this "oracle" set to simulate user feedback for the multi-instance space pruning questions and the example relevance questions. For a pruning question, the user will give positive feedback if the bag covers at least one tuple in the "oracle" set, and a negative feedback if the bag does not cover any tuple in the "oracle" set. For a relevance question, the user will give positive feedback if the example is in the "oracle" set, and negative feedback is the example if not in the "oracle" set. This "oracle" set is also used as the ground truth set to evaluate the accuracy (F-measure) of our final predicted relevant regions.

Environment We implemented all algorithms with Python 3.8 and all the experiments were run on an Intel Core-i9 7980XE processor with 128GB RAM. All experiments reported are averages of 10 complete runs.

Parameters Table 3 shows a list of all settings and schemes of the experiment. In order to assess the impact of different data types accurately, by default, we extract 6786 distinct data objects for each data set, which is the size of the text data. Note using the same data object size across all three data sets allows us to show clearly the impact of different data types, as having inconsistent dataset size can influence the number of examples needed to achieve a certain accuracy level. In addition, the default batch size is 1. Note that the default size of the target relevant region cardinality is 1.0%, and the default target relevant region size is 1 unless otherwise specified. The words "F-measure" and "accuracy" are used interchangeably in the text below.

5.3.2 Experimental Results

Accuracy Comparison Figures 35-51 show the number of examples needed to reach an accuracy (F-measure) of 60%, 70%, and 80% of all participating schemes with different target region numbers and three different datasets (i.e., text, image, and graph). Here we also vary the target

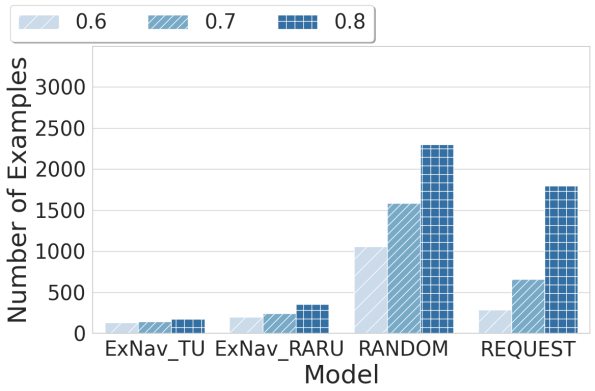

Figure 35: Accuracy 1 Large Region (Text)

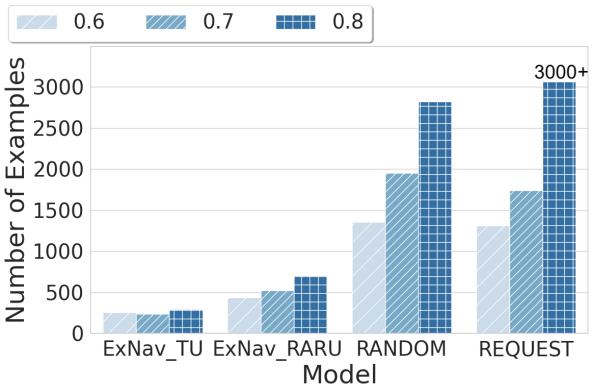

Figure 36: Accuracy 1 Small Region (Text)

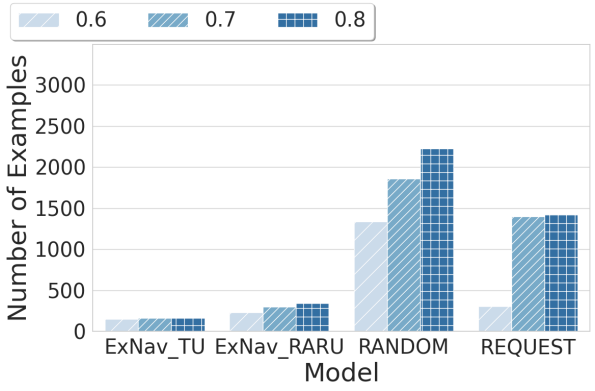

Figure 37: Accuracy 1 Large Region (Image)

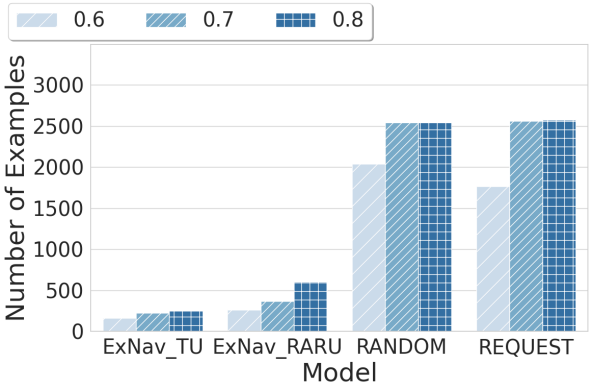

Figure 38: Accuracy 1 Small Region (Image)

region size from large to small. From these figures, we observe that both ExNav schemes have consistently demonstrated significantly higher effectiveness compared to REQUEST and RANDOM.

In particular, for the text data set, 1 relevant region, and ExNav_TU, for example, to reach an accuracy of 60%, ExNav_TU only requires around 125 examples for the large region, around 125 examples for the medium region, and around 250 examples for the small region on average. To achieve the same level of accuracy for the three target region sizes, REQUEST requires 2x, 3x, and

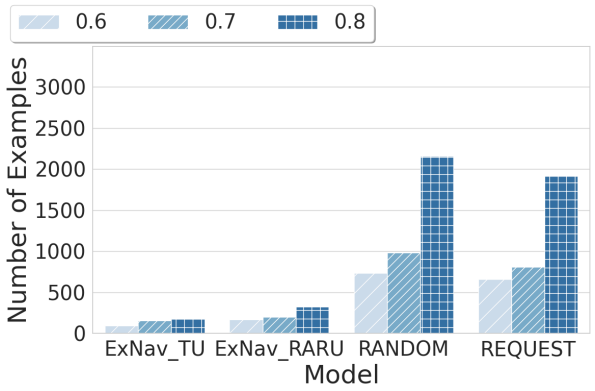

Figure 39: Accuracy 1 Large Region (Graph)

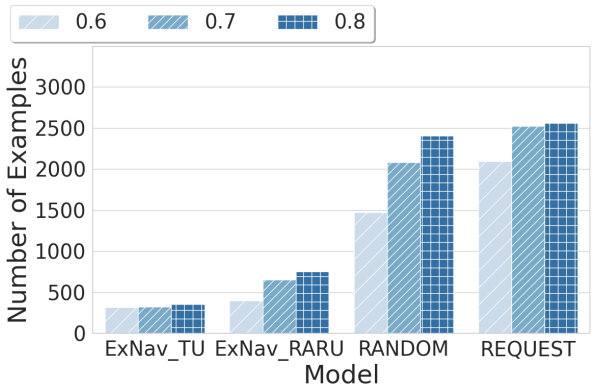

Figure 40: Accuracy 1 Small Region (Graph)

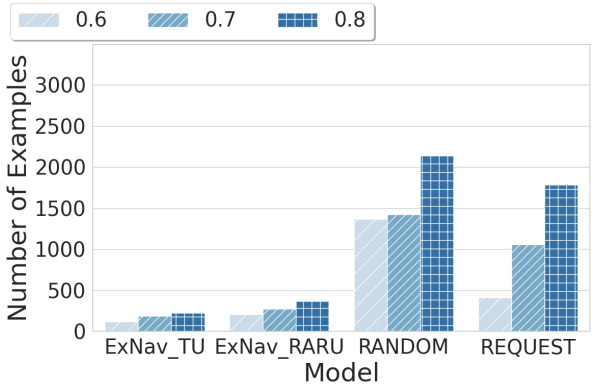

Figure 41: Accuracy 1 Medium Region (Text)

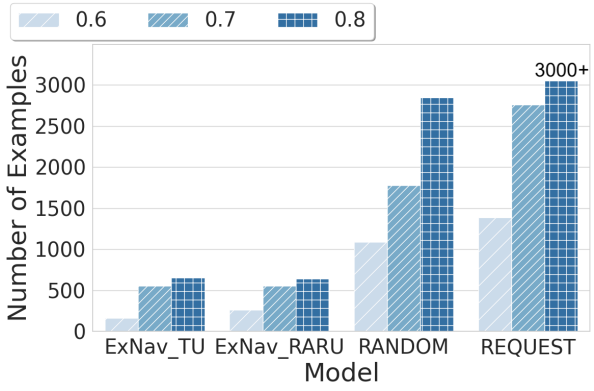

Figure 42: Accuracy 2 Medium Region (Text)

5x more examples, and RANDOM requires 8x, 11x, and 5x more examples, respectively. To reach an accuracy of 80%, ExNav_TU only requires around 180 examples for the large region, around 225 examples for the medium region, and around 325 examples for the small region on average. To achieve the same level of accuracy for the three target region sizes, REQUEST requires 10x, 8x, and >9x more examples, and RANDOM requires 13x, 9x, and 9x more examples, respectively. When we compare the performance for one relevant region, the largest deviation we see between ExNav and REQUEST is for image data set with one medium-sized relevant region, such that

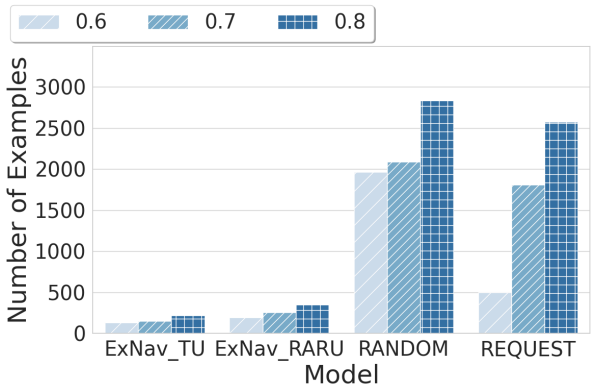

Figure 43: Accuracy 1 Medium Region (Image)

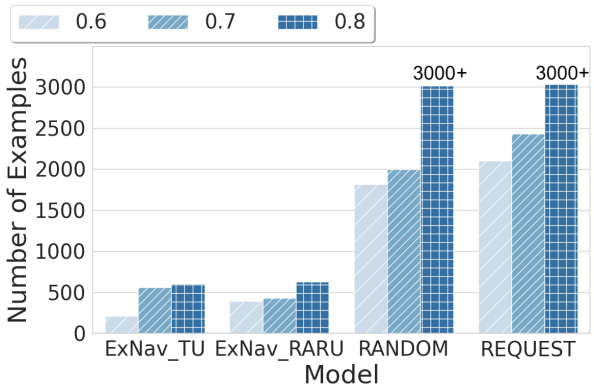

Figure 44: Accuracy 2 Medium Region (Image)

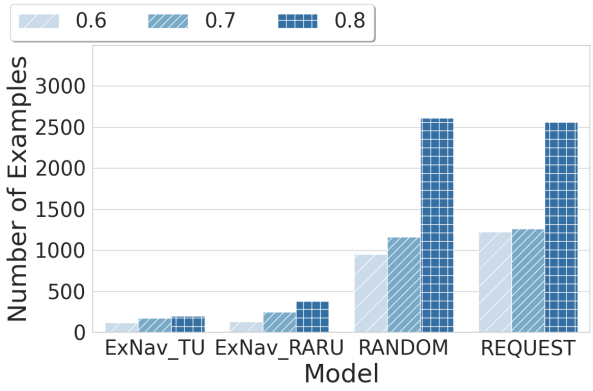

Figure 45: Accuracy 1 Mediums Region (Graph)

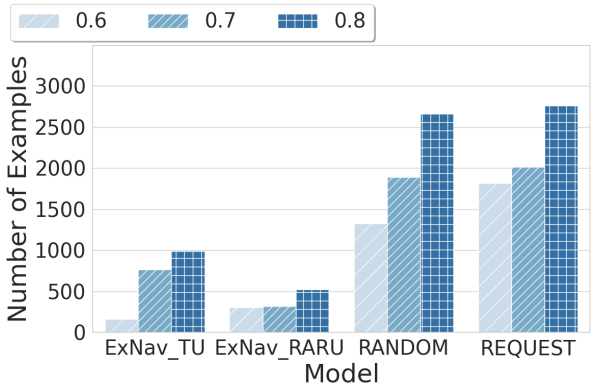

Figure 46: Accuracy 2 Mediums Region (Graph)

REQUEST requires 13x more examples than ExNav in order to reach an accuracy of 60%. Here, we would like to point out that 13x more examples required by REQUEST can be translated to a 92.3% reduction in users' effort when switching from REQUEST to ExNav. Note that in our experiment, we set the example limit to be 3000 as we believe it is not meaningful to label more examples beyond this point. We use the $>$ sign to indicate that the scheme is not able to reach the

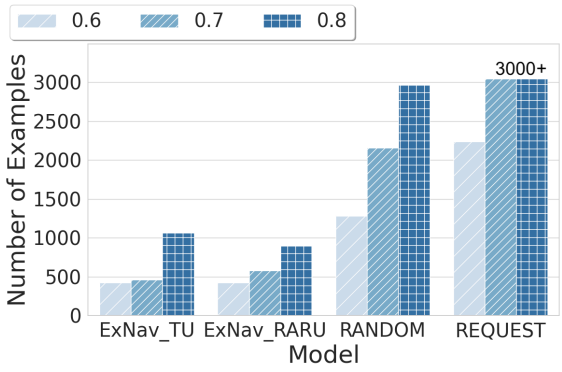

Figure 47: Accuracy 3 Medium Region (Text)

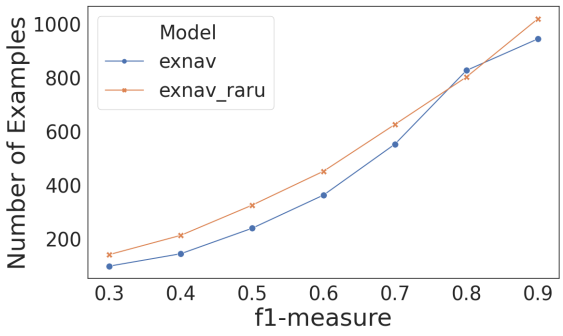

Figure 48: Accuracy 3 Medium Regions (Text)

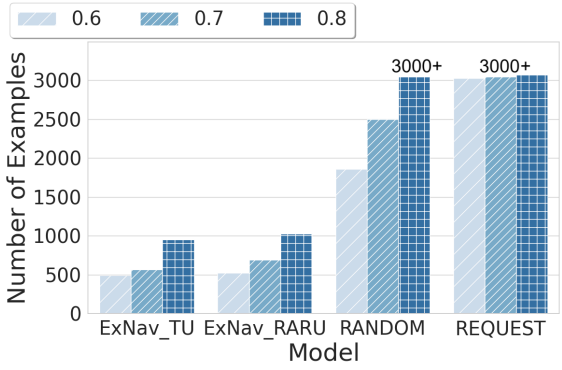

Figure 49: Accuracy 3 Medium Region (Image)

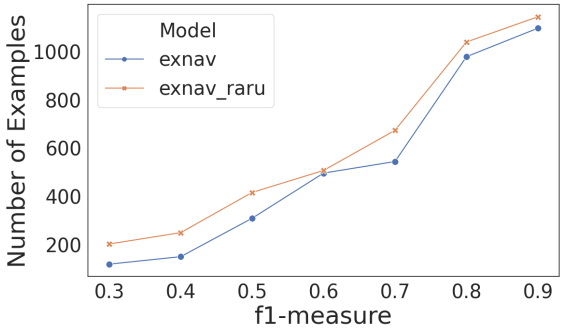

Figure 50: Accuracy 3 Medium Regions (Image)

required accuracy level at 3000 examples.

Moreover, for different medium target region numbers, to reach an accuracy of 60%, ExNav_TU requires around 125 examples for 1 region, around 150 examples for 2 regions, and around 450 examples for 3 regions on average. To achieve the same level of accuracy for the three target region numbers, REQUEST requires 3x, 9x, and 5x more examples and RANDOM would require 11x, 7x, and 3x more examples. To reach an accuracy of 80%, ExNav_TU only requires around

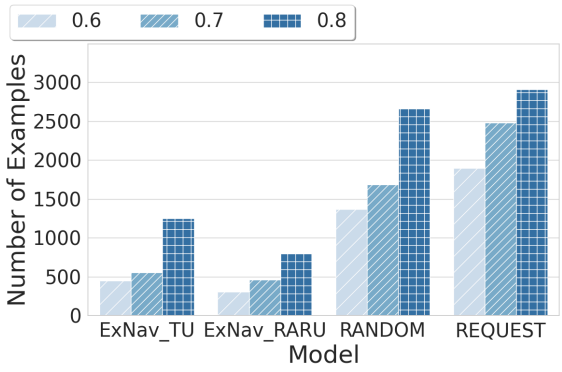

Figure 51: Accuracy 3 Mediums Region (Graph)

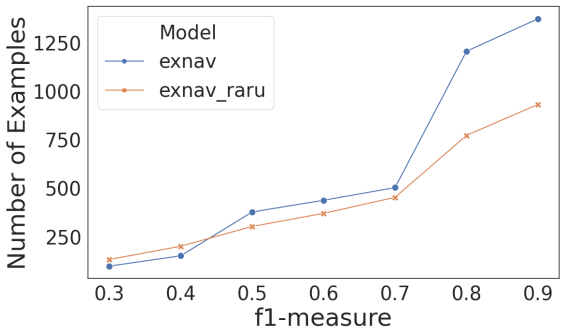

Figure 52: Accuracy 3 Medium Regions (Graph)

225 examples for 1 region, around 670 examples for 2 regions, and around 1050 examples for 3 regions on average. To achieve the same level of accuracy for the three target region numbers, REQUEST requires 8x, >4x, and >3x more examples and RANDOM would require 9x, 4x, and 3x more examples, respectively. During the experiment, we also observed that one of the main reasons for REQUEST to perform poorly is due to its predictive model *Naive Bayes*, which is not able to overfit well with respect to these high dimension embeddings.

Furthermore, when looking at the two ExNav schemes, we observed that ExNav_TU on average performs better than ExNav_RARU in the case of only 1 relevant region. This is to be expected as ExNav_TU performs an exhaustive search over the entire exploration space in each iteration to find the most uncertain object for labeling. However, such benefit fades when more relevant regions exist in the space, such that we see ExNav_RARU on average outperforms ExNav_TU with 2 and 3 relevant regions by a noticeable margin. This is again as expected, since its limitation of shortsightedness, as discussed in Section 5.2, will hinder its capability to discover multiple discrete relevant regions.

Lastly, as shown in the Figures 35-51, the results for the other two datasets also show similar trends for the accuracy comparison. To summarize, all ExNav schemes have similar performance and consistently and significantly outperform REQUEST and RANDOM with respect to accuracy.

Table 4: ExNav Runtime with Different Query Strategies

Query Strategy	10,000 instances	100,000 instances	1,000,000 instances
Traditional Uncertainty	5.88 ms	61.02 ms	675.04 ms
RARU	0.51 ms	5.44 ms	52.79 ms

Scalability Table 4, illustrates the runtime per iteration of ExNav under different query strategies. In particular, we extracted 10000 images from the Caltech-256 dataset and then duplicated the data in order to assess the runtime. The runtime for other data types is similar, as it is independent of a particular data type. The result showed that RARU helps to improve the scalability and the efficiency of the exploration by up to an order of magnitude. This is due to the fact that RARU does not require an exhaustive search over the entire exploration space, and thus, can deliver examples much quicker than traditional uncertainty sampling. These results, combined with the results of accuracy comparison, confirmed RARU's claim that addresses both the shortsightedness and low scalability drawback of the traditional uncertainty sampling as discuss in Section 3.2.2.3. Furthermore, as Explore-by-Examples systems are often used as a post enhancement to the traditional keyword, faceted, or query search results, in these scenarios, the efficiency of ExNav can be further improved due to fewer data involved in the exploration.

Zoom-in into ExNav Schemes In order to better illustrate the comparison between our proposed schemes, in Figures 48-52, we zoom into the two ExNav schemes and show the number of examples needed to reach different levels of accuracy from 30% to 90%. For this set of comparisons, we consider a relatively complex scenario with three medium relevant regions. Compared to ExNav_TU, we have noticed that ExNav_RARU typically requires more labeled examples in the early stages of the experiment (e.g., below 70% of accuracy). This is due to the fact that in the early stages, to raise accuracy, it does not require all relevant regions to be identified accurately. However, when improving accuracy beyond 70%, ExNav_TU appears to be more struggled than ExNav_RARU (i.e., has a larger slope), which is expected due to its limitation of shortsightedness.

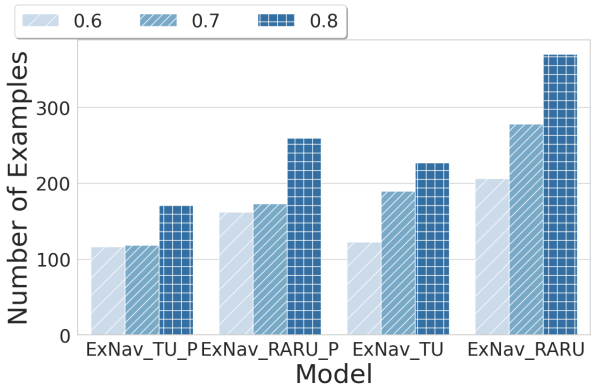

Figure 53: Accuracy 1 Medium Region (Text)

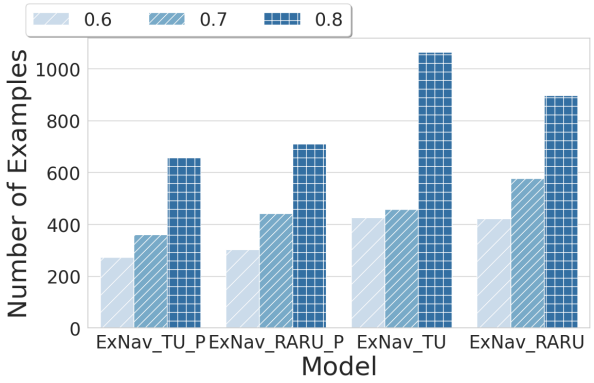

Figure 54: Accuracy 3 Medium Regions (Text)

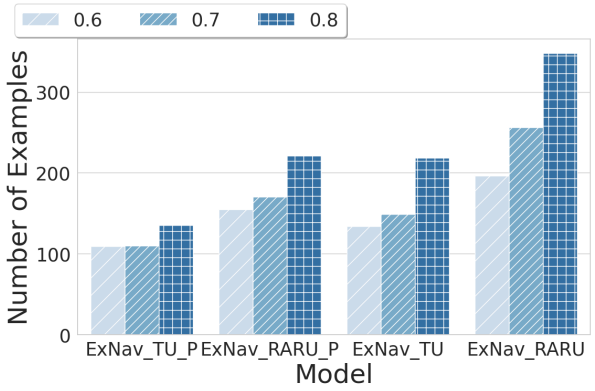

Figure 55: Accuracy 1 Medium Region (Image)

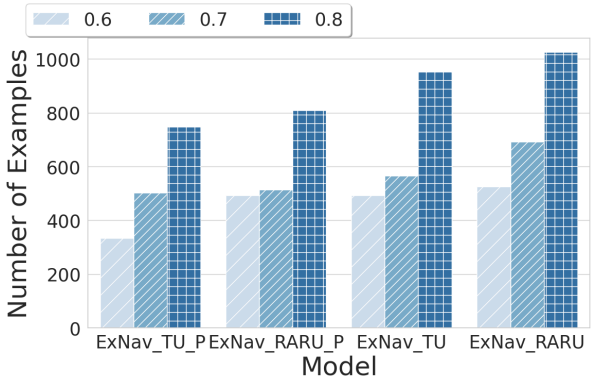

Figure 56: Accuracy 3 Medium Regions (Image)

Impact of Multi-instance Space Pruning As mentioned in the experiment setup, we have evaluated the effectiveness of our MIS pruning strategy. In particular, we created 16 bags by grouping similar objects and asking the user to tell us if any of the 16 bags are far away from the relevant region (i.e., contain no relevant objects). These regions will then be pruned from subsequent exploration. Note that the number of examples reported in the figures for ExNav_TU_P and ExNav_RARU_P does not include the 16 bags question, which has been applied as a fixed preprocess for these two schemes. The accuracy comparison between the ExNav schemes with and

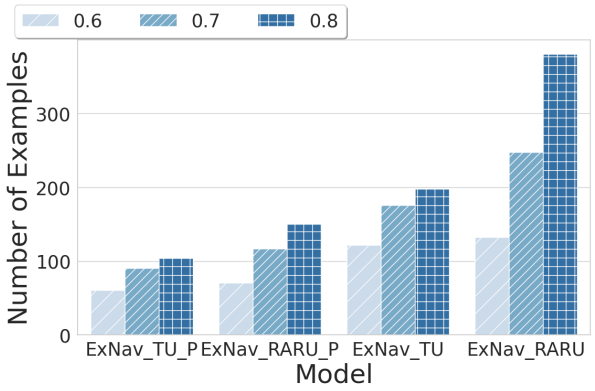

Figure 57: Accuracy 1 Medium Region (Graph)

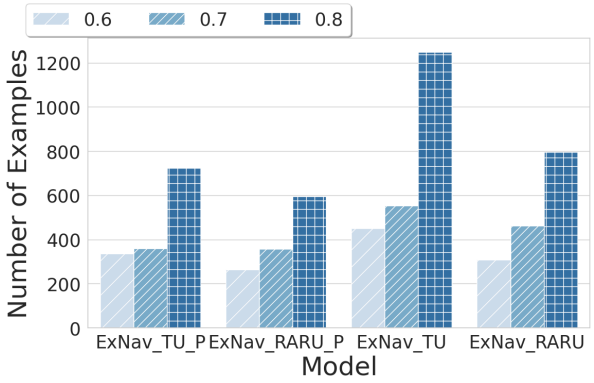

Figure 58: Accuracy 3 Medium Regions (Graph)

without pruning is shown in Figures 53-58. We can see that, for ExNav_TU_P, the pruning can reduce or save around 25%, 27%, and 40% user effort on average for the three datasets, respectively. For ExNav_RARU_P, the pruning can save around 26%, 21%, and 36% user effort on average for the three datasets, respectively. These results indicate that Multi-instance Space Pruning is effective in reducing the exploration space as well as the amount of feedback needed for exploration. Therefore, as discussed early in Section 5.2, when exploring any data domains where group labels are inexpensive to provide, using ExNav with MIS Pruning is recommended.

5.4 Summary

Motivated by the challenge of reducing human effect in exploring large unstructured datasets, in this chapter, we proposed ExNav, a novel generic Example-driven Exploration framework for interactive exploration of unstructured data. ExNav effectively navigates users through the large exploration space to find relevant data items that are often undiscoverable by many traditional exploration or search methods.

Our ExNav enables the exploration of any unstructured data as long as an embedding representation can be obtained. In addition, we described in detail two key components of ExNav, namely, *Multi-instance Space Pruning* and *Query Strategy*, along with a set of optimization techniques that helps to improve the effectiveness of ExNav.

We implemented a prototype of ExNav and experimentally verified its performance with three real-world datasets. The results have shown that our proposed ExNav exhibits significantly better performance when compared to the state-of-the-art while achieving desired interactive performance. Specifically, ExNav can reduce users' effort by up to 92.3% (i.e., 13x less than the state-of-the-art) while still achieving the same accuracy as the state-of-the-art alternative.

6.0 Exploration Result Refinement

The work covered in this chapter was published in the Proceedings of 2020 International Conference on Extending Database Technology (EDBT) [Ge and Chrysanthis, 2020].

Even with the "most precise" set of results discovered by the Example-driven Exploration systems, the volume of the results can still be overwhelming to the users. In this chapter, we present our final contribution, which is a solution to address this challenging problem that is fundamental to the usability of the Example-driven Exploration systems.

6.1 Introduction

As mentioned earlier, the challenge of scalable data exploration can be examined from two viewpoints. Traditionally, scalability has been seen from a systems point of view, where challenges can be attributed to an increasing rate of data on the one hand, and processing power, and storage limitation on the other hand. However, scalability can also be viewed from a human point of view. Given the exponential volume of data, the challenge here is how to avoid overwhelming users with irrelevant results. Existing Example-driven Exploration systems aim to construct a precise query that describes the user's interest, however, the amount of data objets obtained by such query may still be overwhelming, and thus, impractical for the users to comprehend. In this thesis, we devised a novel result refinement algorithm, coined Preferential Diversity (PrefDiv), which addresses the problem of the Top-k result diversification that lies in the center of such challenge.

To begin, we will first discuss the notion of result refinement, which is a well-known solution for dealing with such a challenge, which often happens at two different levels:

- *Ranking* – Ranking techniques utilize user *preferences* with the aim of providing the most relevant results to the users (e.g., [Stefanidis et al., 2011]). These techniques can be distinguished as *quantitative-based*, *qualitative-based*, or *hybrid,* based on the type of user preferences that they can support.

- *Diversification* – Diversification techniques aim to reduce the amount of redundant information in the results. These techniques typically group data in sets that are most ”dissimilar” with each other (e.g., [Santos et al., 2015, Angel and Koudas, 2011, Fraternali et al., 2012]).

Since items ranked highly by ranking algorithms can be similar, thus, Top-k result diversification has recently drawn significant attention as a result refinement technique to facilitate applications such as keyword search, recommendation systems, and online shopping. The key idea of result diversification is to output a subset of representative results from the original in an informative way since the user most probably will not view results beyond a small number. As illustrated in Section 6.2, this requires the representative top-k results to be relevant, diverse, and maintain good coverage of the original answers (i.e., able to cover different underlying aspects of the original results). One thing to note is that the definitions of both relevance and diversity are subjective; thus, they can vary depending on the query and the user.

Our goal in this chapter is to present an approach to efficiently compute the representative result set for arbitrary top-k queries under user-definable relevance and diversity definitions. We name this as the *Diversified Top-k (DT-k)* problem (Section 6.3).

To effectively address the DT-k problem, we present an extremely efficient online algorithm, called *Preferential Diversity* (PrefDiv), for producing representative result sets with sufficient relevance, diversity, and coverage of the original answers (Section 6.4). PrefDiv is a top-k bounded general diversification approach that can be applied to any existing relevance ranking model and datasets to retrieve a diversity-aware top-k representative subset of results.

To the best of our knowledge, PrefDiv is the first general approach to deliver representative results that explicitly consider relevance, diversity, and coverage with an interactive speed that is independent of the underlying database and data set.

To optimize multiple conflicting objectives such as relevance and diversity, a common approach taken by most diversification algorithms utilize a number of tunable parameters. This could be a major drawback for an algorithm, because with the increase of the number of required parameters, the complexity of the algorithm increases as well, making it more difficult to use in real-world

scenarios. Consequently, we extended PrefDiv into a family of algorithms. These include two novel algorithms that automatically determine the two main tunable parameters of PrefDiv: 1) the corresponding diversity thresholds $DIV = \{div_1, div_2, ..., div_n\}$ given the set of diversity constraints Ψ, and 2) the tunable parameters A that balance the trade-off between the relevance and diversity, respectively.

Furthermore, we experimentally evaluated our PrefDiv algorithms in terms of normalized relevance, coverage, and execution time. Our evaluation indicates a speedup of up to 159x, and outperforms the state-of-the-art algorithms on multiple fronts (Section 6.5).

6.2 Problem Challenges

Before presenting our solutions, we will first illustrate the challenges to our proposed approach by means of three examples.

Example 1. Assume a tourist who is currently visiting Athens wants to find an affordable restaurant with great taste. So she visits a publicly available database that contains the relation RESTAURANT (Name, Food Type, Cost, Score), where *Name* indicates the official name of the restaurant; *Food Type* indicates the type of food (e.g., Greek, Japanese, Chinese); *Cost* is the average expense per person, and *Score* is a numeric number between 1 and 10 that indicates the quality of the food and services offered at the restaurant. To find the ideal place for dinner, she, therefore, obtains the following SQL-like query that describes her initial desire through either the help of data exploration systems (e.g., Example-driven Exploration) or by handcraft it manually:

*SELECT * FROM RESTAURANT*

WHERE $Score \geq 6\ AND\ Cost \leq 20$

ORDER BY Cost ASC;

However, these kinds of queries may produce thousands of results, among which the top 5 and

Table 5: Top-5 and Bottom-5 Tuples With Respect to the Cost

Name	Food Type	Cost	Score
McDonald	Fast Food	8	7
KFC	Fast Food	8	7
Burger King	Fast Food	8	7
Arby's	Fast Food	8	7
Oinomageireio H Epirus	Greek	8	9
.....			
Scala Vinoteca	Greek	20	9
Ta Karamanlidika tou Fani	Greek	20	10
A Little Taste of Home	Greek	20	9
Liondi Traditional Greek	Greek	20	9
Dio Dekares i Oka	Greek	20	9

bottom 5 results are listed in Table 5. The problem is that users are typically only interested in seeing a small portion of these results, not to mention many of these results are, in fact, redundant (e.g., differ only in the name). Simply fetching a certain top number (e.g., top 5) of results does not help improve their usefulness. Instead, the user might be better served with the right amount of diverse (i.e., dissimilar) items from the original answer with good coverage of different aspects (e.g., Food Type, Cost, Score). Furthermore, among those representative subsets with good diversity and coverage, the one that is most relevant to the user's interest should be preferred, such that the relevance refers to criteria that can be used to rank the answers. These may be obtained by interoperating the SQL-like query itself (e.g., through the "Order By" predicates), or derived from external user profiles (e.g., query histories, crowdsourcing).

One immediate challenge raised is how to define diversity, which clearly changes based on the user and the query being performed. In our work, we associate diversity with the similarity between pairs of answers (i.e., data items). To address this challenge, we propose a tunable definition that can be adjusted with a set of diversity thresholds DIV. Each threshold div in DIV is a real number between $[0, 1]$, which specifies the threshold between "similar" and "dissimilar" data items

Table 6: Top-5 Tuples Based on Cost That Are Diversified With Respect to Attributes "Food Type" and "Score".

Name	Food Type	Cost	Score
McDonald	Fast Food	8	7
Beer Garden Ritterburg	German	8	9
Nolan	Japanese	9	8
Oinomageireio H Epirus	Greek	10	10
Dosirak	Korean	12	6

with respect to the normalized distance given by the specified distance measure (e.g., Euclidean, Manhattan, and Hamming) and attributes. $|DIV| = 0$ results in the traditional top-k query, while more diversity thresholds with higher values increase the diversity of the result set.

Example 2. With the above diversity parameter, the previous sample query in Example 1 could be expanded accordingly:

SELECT * *FROM* RESTAURANT

WHERE $Score \geq 6\ AND\ (Cost \leq 20)$

ORDER BY Cost *DESC*

DIVERSE BY $div = 0.2$ *ON* 'Food Type' (Hamming)

AND $div = 0.3$ *ON* 'Score' (Euclidean) *LIMIT* 5*;*

where Food Type and Score are the attributes on which the diversity is calculated, and Hamming and Euclidean are the corresponding distance measures. The idea here is to generate a set of results that follow the diversity constraints DIV specified within the query [1]. The result of the above query is illustrated in Table 6. Although the above example produces some compelling results with its information representative subset, it could be difficult to see how the coverage is contributing differently to the results than the dissimilarity. To illustrate the importance of the coverage, let us consider a simple example:

[1]Note that our PrefDiv algorithms take the set of diversity constraints DIV as one of their inputs, and it is up to the design of the actual system that integrates the PrefDiv to determine how DIV will be integrated with its user query.

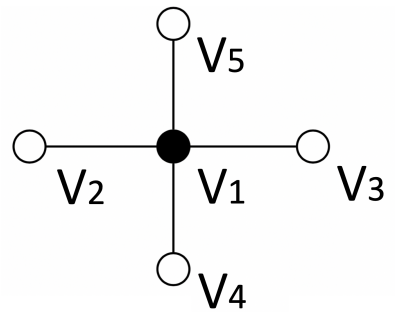

Figure 59: Single Vertex v_1 With 100% Coverage.

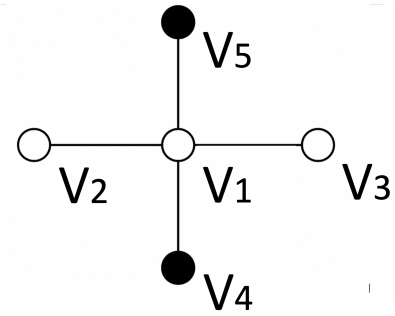

Figure 60: A Set of Vertices $\{v_4, v_5\}$ With 60% Coverage.

Example 3. Consider the nodes in Figures 59 and 60. In these two figures, each node represents an item in the dataset, and an edge exists between a pair of nodes iff the similarity between these two nodes is close enough according to some pre-defined threshold. On the one hand, in Figure 60, a set of dissimilar items $\{v_5, v_4\}$ is selected. However, only $\{v_1, v_5, v_4\}$ are considered to be covered by $\{v_5, v_4\}$, as $\{v_2, v_3\}$ are not connected with either v_5 or v_4. On the other hand, in Figure 59, a single vertex v_1 is connected to all four vertices, hence achieving 100% coverage. In this case, one can see that vertex v_1 better represents the entire graph when compared with $\{v_5, v_4\}$, thus indicating coverage is another valuable aspect to the quality of the representative results.

The above two examples (i.e., Example 2 and 3) illustrate the key advantages and desired features of an effective approach that provides a meaningful and representative subset of the original query results. First, the representative subset is relevant to the intention of the query and contains items that would be ranked highly in the original results. Second, the chosen representative items are diverse, each contributing additional novelty to the answer. Third, the representative items are selected in a way that most items in the original answers are reachable with a small distance (i.e., change) from one of the representative answers. Clearly, simply applying ranking, diversification, or clustering on the original result sets could not achieve the above properties. Thus, techniques that consider multiple aspects of the representative results are needed to address this challenge.

Unfortunately, as we will discuss in more detail in Section 6.3.2.3, finding the optimal solution that maximizes both the "relevance" and "diversity" is an NP-Hard problem by itself, let alone with the addition of the other aspect "coverage" that should also be considered when producing the representative results.

6.3 Problem Formulation

In our work, we assume that the database DB is composed of N items over a D-dimensional space $(d_1, d_2, ..., d_D)$, where each dimension $d \in D$ can be either numerical or categorical attributes. Note that the above assumption enables us to handle any data type (e.g., structured, semi-structured, unstructured) as long as they are vectorized. The user specifies a query Q that aims to retrieve a set of k representative items from DB over a subset of dimensions S, such that $S \leq D$. The goal here is to produce a set of k items that maximizes the *relevance* while ensuring the *diversity* (i.e., ensuring each item is diverse with respect to one another). Below, we will first provide the necessary background and basic concepts of our problem and then present our problem definition and analyze its complexity. The list of symbols used in the following sections of the chapter is shown in Table 7.

6.3.1 Background

6.3.1.1 Relevance The relevance of R represents the degree of the relevancy of each data item $x \in R$ and is typically represented with a utility function $U(x)$ that measures the "goodness" of each data item with respect to certain metrics.

Definition 3. Relevance – *Given a database DB and a utility function $U(x)$, the relevance is measured as the outcome of $U(x)$, which is an* intensity value *$I_x \in (0, 1) \subset \mathbb{R}$ for item $x \in$ DB that is used to express the degree of benefit from retrieving x.*

Table 7: List of Notations Used in Chapter 6

Symbol	Explanation
R_Q	a set of initial query results
R	a set of representative results
k	the number of item in the result-set
$\mathcal{L}$	the number of iterations to obtain R with PrefDiv
Ψ	a set of diversity constraints
ψ	the diversity constraint
div	a diversity threshold
div_{opt}	an optimal diversity threshold
Δ	a set of dimensions
A	a relevance parameter
υ	a self-adjustable relevance parameter
I_x	the intensity value of item x
$U(x)$	a utility function that produces the I_x
$sim_\Psi(x_i, x_j)$	x_i and x_j are similar w.r.t Ψ
$dissim_\Psi(x_i, x_j)$	x_i and x_j are dissimilar w.r.t Ψ

A higher intensity value indicates that a data item is more desired than those items with a lower intensity value.

The intensity value (i.e., relevance score) enables database systems to produce a total order of each data item in a given data set and thus allow the extraction of top-k data items. This is simply achieved by retrieving k data items with the highest intensity value.

6.3.1.2 Diversity In our work, the diversity of a set of data items R is achieved by enforcing each pair of data items in R to be *dissimilar* with respect to each other, such that two data items x_i,

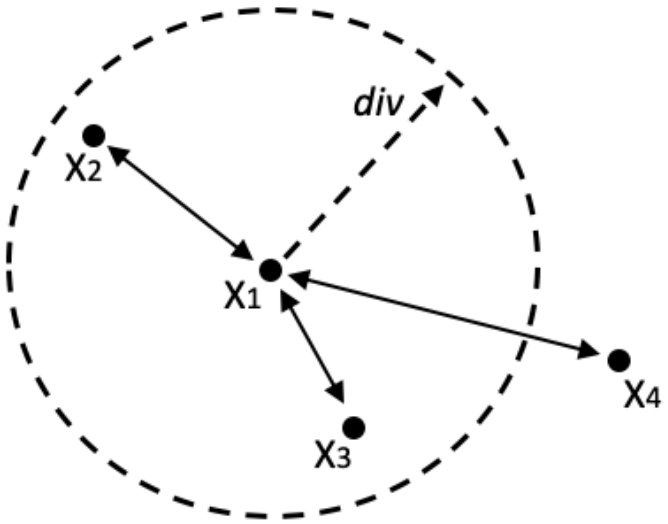

Figure 61: Illustration of Similarity and Dissimilarity.

and x_j are said to be dissimilar if for a given set of user-specified diversity constraints Ψ, x_i and x_j satisfy all constraints $\psi \in \Psi$. Formally, we can define a diversity constraint as follows:

Definition 4. Diversity Constraint – *For a given pair of items x_i and x_j, a set of attributes Δ, a distance threshold div, and a distance function $dist(x_i, x_j, \Delta)$ that measures the distance between x_i and x_j with respect to the set of dimensions specified in Δ. A diversity constraint ψ is satisfied iff $dist(x_i, x_j, \Delta) > div$.*

Based on the above definition of diversity constraints, we now define dissimilarity as:

Definition 5. Dissimilarity – *Let X be a set of data items. For a given set of diversity constraints Ψ, two items x_i and $x_j \in X$ are dissimilar to each other, denoted as $dissim_\Psi(x_i, x_j)$, if they satisfy each diversity constraint ψ in Ψ.*

Consequently, the similarity can simply be defined as the opposite of the dissimilarity, such that:

Definition 6. Similarity – *Let X be a set of data items. For a given set of diversity constraints Ψ,*

two items x_i and $x_j \in X$ are similar to each other, denoted as $sim_\Psi(x_i, x_j)$, if they fail to satisfy at least one diversity constraint in Ψ.

Figure 61 illustrates the concept of similarity and dissimilarity with four 2-dimensional data items $x_1, .., x_4$, and a single diversity constraint that requires the Euclidean distance between each data object with respect to both dimensions (i.e., $\Delta = d_1, d_2$) to be at least div apart. Let us take point x_1 as an example. According to Definition 6, points $\{x_2, x_3\}$ are similar to x_1, since $dist(x_2, x_1, \Delta) \leq div$ and $dist(x_3, x_1, \Delta) \leq div$. In contrast, x_4 is dissimilar with respect to x_1, as $dist(x_4, x_1, \Delta) > div$.

6.3.1.3 Coverage As pointed out in the previous literature [Drosou and Pitoura, 2015], the coverage is another aspect that is important to the quality of the representative results. Since the size of the representative results is very restricted compared to the original answers, having a set of representative results with good coverage increases the chance for the user to get meaningful information from the selected representation items. Furthermore, coverage enables the system to organize all the answers in a cluster-like fashion, where each original answer of query Q can still be retrieved by "zoom-in" into one of the representative items. Such that the "zoom-in" operation will reveal all answers that are "similar" to the selected representative item. The actual implementation of this "zoom-in" operation has been well discussed in [Drosou and Pitoura, 2015], thus it is omitted from the discussion of this chapter.

Clearly, the coverage is defined completely based on the definition of the similarity and thus related heavily to the diversity constraints when the number of representative results is fixed to a certain number k. When k is fixed, a set of more relaxed diversity constraints (i.e., with a higher diversity threshold) will help the representative set to include more original answers into its coverage, and a set of stricter diversity constraints will certainly decrease the coverage of the representative set. In particular, given the definition of similarity, if item x_j satisfies $sim_\Psi(x_i, x_j)$, x_j is said to be covered by the item x_i. Consequently, we can define the coverage of a set of items as follows:

Definition 7. Coverage – *Given a set of original answers R_Q and a representative result set R, where $R \subseteq R_Q$, the coverage of R corresponds to the percentage of items in R_Q that satisfies $sim_\Psi(x_i, x_j)$, such that $x_i \in R$ and $x_j \in R_Q$.*

6.3.2 Diversified Top-k (DT-k) Problem

Based on the above discussions and definitions, we name our problem the *Diversified Top-k (DT-k)* problem.

6.3.2.1 Problem Formulation Consider a database DB that consists of N data items distributed over a multi-dimensional space with mixed numeric and categorical dimensions. Given a query Q and its corresponding initial results set R_Q over DB, the desired result cardinality of k, a utility function $U(x)$, and a set of diversity constraints Ψ, the solution of DT-k produces a k-sized representative subset R from the original results R_Q, whose *relevance*, according to $U(x)$ is maximum, while satisfying the set of diversity constraints Ψ.

We name the above k-sized subset of representative results as the Diversified Top-k (DT-k) set.

6.3.2.2 Problem Complexity Finding the optimal DT-k Set for the Diversified Top-k problem is computationally hard, which can be shown by mapping it to the well-known *Maximum-weight Independent Set* problem [Ind, 2021]. We can achieve the mapping by forming a graph of G that corresponds to the original results R_Q. Each data item x_i in R_Q maps to a vertex v_i in G. An edge e is added between two vertices v_i and v_j if the distance between these two vertices is close enough such that not all diversity constraints are satisfied, and the intensity value I_{x_i} of an item x_i represents the weights of the corresponding vertex in G. Some tractable solutions have been proposed in the literature [Keil et al., 2015, Grohe, 1999], but these solutions require either a very specific type of graph (e.g., Outerstring graphs) or have strict restrictions (e.g., sparsity, outcome degree of each vertex). Thus, they are not practical in our environment.

6.3.2.3 Secondary Objective As discussed above, coverage is another important aspect of result diversification, which is dependent completely on the diversity threshold specified inside each diversity constraint. Given that diversity constraints are typically defined by the user, this may lead to sub-optimal results if the user fails to define reasonable constraints. Consequently, our secondary objective is to address this challenge by automatically adapting the diversity constraints based on the type of query being performed and the initial result set. Later, in Section 6.4.4, we will present a general optimization that helps determine the most suitable diversity constraints for different user queries.

6.4 PrefDiv Algorithms

In this section, we introduce our solution to the Diversified Top-k problem. First, we start with the discussion of a naive approach to the problem and then propose our solution to this problem, namely, *Preferential Diversity* (PrefDiv) algorithm. Finally, we discuss some optimizations that improve the effectiveness of our proposed PrefDiv algorithm and reduce its number of tunable parameters.

6.4.1 Naive Solution

Before we discuss our solutions, one naive solution to the Diversified Top-k problem work as follows: given a new user query Q, a k, a set of initial results $R_Q = \{x_1, ..., x_t\}$, a utility function $U(x)$ and a set of diversity constraints Ψ, for each item in R_Q of q, we first compute and sort each item in R_Q according to the intensity value computed by the $U(x)$. We pick the item $x_i \in R_Q$ with the highest intensity value; for each remaining items x_j in R_Q, we mark them as "Eliminated" if they are similar to the x_i (i.e., $sim_\Psi(x_i, x_j)$). We then add x_i into the final result set R and remove x_i from R_Q. Afterward, a new unmarked item with the highest intensity value will be picked from R_Q, and the previous steps will be repeated until either $|R| = k$ or all remaining items in $|R_Q|$ are marked as "Eliminated".

This naive solution is a greedy approach that will eventually produce a set of items that satisfy all diversity constraints with relatively high-intensity values. Clearly, the naive solution is computationally expensive, especially when the size of R_Q is large. Furthermore, it does not guarantee the resulting set to contain at least k items. However, we use this naive solution as a foundation and propose an efficient online solution that achieves better performance with much less computational cost.

6.4.2 Preferential Diversity

Our Preferential Diversity algorithm is an online solution for the DT-k problem. As discussed in the previous section, finding the optimal solution to the DT-k problem is computationally expensive. Thus we chose a greedy approach in the PrefDiv design. To maximize the efficiency of PrefDiv, we need to develop it as an online algorithm that accesses database tuples (i.e., items) incrementally. The main idea underlying PrefDiv is minimizing as much as possible the number of data items being examined.

PrefDiv builds the DT-k set R by starting with a set of k highest ranked data item (with respect to the relevance score/intensity value), and then gradually replacing items that fail the diversity constraints with slightly less relevant but diverse items outside of R that satisfy the diversity constraints. This process continues until all items in R satisfy the specified diversity constraints.

One potential issue is that relevant items in the DT-k set tend to be similar to each other. Thus strictly enforcing diversity constraints may eliminate too many items that are highly beneficial to the user. To address this issue, we propose a *relevance parameter* A that allows PrefDiv to produce representative results with *partial* diversity. When $A = 1$, R would simply be the top k items from the initial set, i.e., the items with the k highest intensity values. When $A = 0$, R contains k dissimilar items from the initial set. When A is between 0 and 1 and given that PrefDiv is an iterative algorithm, considering k objectives each iteration, the final result will have at least $A * k$ items from every iteration, and in each iteration A will be divided by half. For example, when $A = 0.5$ and $k = 20$, the first iteration will select at least $20 * 0.5$ items for the final result set, the

Algorithm 5 PrefDiv

Require:

Initial result set R_Q, target result cardinality k, relevance parameter A, a set of diversity constraints Ψ

Ensure:

One subset R of R_Q

$T \leftarrow \emptyset$
while exists unexamined items in R_Q and $|R| < k$ **do**
  $T \leftarrow$ Pick k items with highest intensity from R_Q
  for all $x_i \in T$ **do**
    if $Dissim_\Psi(x_i, x_j) : \forall x_j \in R$ **then**
      $R \leftarrow R \cup x_i$
    else
      Mark x_i as "redundant"
    end if
  end for
  while number of promoted items in R from T ¡ $A * k$ **do**
    $R \leftarrow R \cup x_{max}$, s.t., x_{max} is marked & $\forall x_j \in T, I_{x_{max}} \geq I_{x_j}$
    $T \leftarrow T - x_{max}$
  end while
  $A \leftarrow A/2$
  $R_Q = R_Q - T$
end while
Return R

second iteration will select at least $20 * (0.5 * 0.5)$ items, and so on. Consequently, the user is able to control the trade-off between relevance *vs.* diversity by enabling partial diversity on demand.

As illustrated in Algorithm 5, the basic logic of PrefDiv is as follows: PrefDiv takes as input a set of initial results R_Q sorted according to the descending of their intensity value, the desired result cardinality of k, partial diversity parameter A, and a set of diversity constraints Ψ. It outputs a DT-k set that represents the original answers R_Q. In each iteration, PrefDiv fetches and removes k items with the highest intensity value from R_Q and places them in a temporary set T. Each of the items in T is then compared with items currently in R, such that any item in T that fails to satisfy all diversity constraints with respect to all items in R will be marked as "Redundant"; else, it will be added into R immediately. This process will continue until all items in T are either moved into R or marked as "Redundant". Once all k items fetched in the current iteration have been examined, PrefDiv will check if a sufficient number of items were moved into R according to parameter A. In case the number is not sufficient, the difference will be covered by the highest-ranked items (with respect to intensity value) that are marked as "Redundant" in the current iteration. Afterward, the above iteration will be repeated until k representative items are produced ($|R| = k$).

Time Complexity According to the above discussion, we can observe that the worst-case complexity of PrefDiv is $O(kN)$, since each of the N unlabeled items will be compared at most $k-1$ times with the items that are currently in the result set before being included or discarded from the final result. Fortunately, as the size of k is usually a small number, PrefDiv should typically behave as a linear algorithm. Furthermore, as we will show in our empirical studies (Section 6.5), depending on the diversity constraints, PrefDiv typically does not need to examine all original items in R_Q. That is, a very small set of the item would be sufficient enough to produce R if Ψ are appropriately defined.

6.4.3 Relevance Proportionality

From the above discussions, it should be clear that having a good balance between relevance and diversity is important to the quality of the representative result set. In PrefDiv, we have introduced the relevance parameter A to enable the partial diversity, which helps preserve the relevance of the representative results. Our empirical study shows that such a parameter does help improve

Algorithm 6 PrefDiv-PR

Require:

Initial result set R_Q, target result cardinality k, a set of diversity constraints Ψ

Ensure:

One subset R of R_Q

$\mathcal{L} \leftarrow 0$

$T \leftarrow \emptyset$

while exists unexamined items in R_Q and $|R| < k$ **do**

  $T \leftarrow$ Pick k items with highest intensity from R_Q

  for all $x_j \in T$ **do**

    if $Dissim_\Psi(x_j, x_t) : \forall x_t \in R$ **then**

      $R \leftarrow R \cup x_j$

    end if

  end for

  $R_Q = R_Q - T$

  $B_\mathcal{L} \leftarrow T - R$

  Increase $\mathcal{L}$ by one

end while

for $\ell = 1 \rightarrow \mathcal{L}$ **do**

  $\upsilon_\ell \leftarrow$ Compute υ_ℓ according to Equation 6.1

  while the number of items in R from B_ℓ ¡ $\upsilon_\ell * |R|$ **do**

    $R \leftarrow R - x_j$, s.t. $x_j \in R$, $I_{x_j} < I_{x_k} : \forall x_k \in R$

    $R \leftarrow R \cup x_i$, s.t. $x_i \in B_\ell$, $I_{x_i} \geq I_{x_t} : \forall x_t \in B_\ell$

    $B_\ell \leftarrow B_\ell - x_i$

  end while

end for

Return R

the quality of the result set R. However, it is up to the user to define A for any query, and this may increase user efforts when using our algorithm. This motivated us to introduce a new self-adjusted parameter υ to replace the manual relevance parameter A, which led to a new variation of PrefDiv called *Prefernеral Diversity with Proportional Relevance* (PrefDiv-PR). As illustrated in Algorithm 6, the idea here is to automatically compute the right amount of items that should be promoted into the final result set based on the proportion of the relevance of each iteration.

In particular, υ adapts to the aggregated intensity value of all items in each iteration and can be computed as follows: Assume a given set of original results R_Q, a DT-k subset $R \subseteq R_Q$ and a number of iterations $\mathcal{L}$ needed for PrefDiv to obtain the result set R. For each iteration ℓ., s.t. $\ell < \mathcal{L}$, a set of items with the highest intensity value from the remaining items of R_Q are reserved into a separated set B_ℓ, s.t. $B_\ell \subseteq R_Q$ and $|B_\ell| = k$. The υ_ℓ of an iteration ℓ is calculated through the following equation:

$$\upsilon_\ell = \frac{\sum_{x \in B_\ell} I_x}{\sum_{j=1}^{\mathcal{L}} \sum_{x_c \in B_j} I_{x_c}} \tag{6.1}$$

Recall from Section 6.3.1.1, I_x is the intensity value of data item x. The idea here is that at least υ_ℓ proportion (i.e., percent) of the item in the final representative result set should be extracted from iteration ℓ, as early iterations would always have a higher aggregated intensity value, and thus, would occupy a bigger portion of the final representative set R. Our empirical results show that by employing υ to compensate for the loss of relevance, we can prevent too many relevant results from being dropped.

To actually generate the final representative results according to υ_ℓ, PrefDiv-PR needs to first run PrefDiv to obtain the initial representative set R, as well as records the number of iterations $\mathcal{L}$ taken to obtained R. During each iteration of PrefDiv, the items that are initially extracted from R_Q (before applying the diversity constraints) will also be recorded into a separate set B_ℓ. Afterward, PrefDiv-PR will examine the set of items in R that are extracted from each B_ℓ with the corresponding υ_ℓ to determine if additional items need to be extracted from B_ℓ and added into R. Note that if such extraction is necessary, depending on the number of items needed to be extracted, the set of items with the highest intensity value in B_ℓ that have not been included in R will be

chosen from B_ℓ and placed in R. Finally, once all v are satisfied for each iteration, the set of k items with the highest intensity value in R will be retrieved as the final results.

6.4.4 Optimize Diversity Constraints for Coverage

As discussed previously in Section 6.3.1.3, coverage is yet another important property of the representative set. It gives us two major benefits. First, it helps to ensure that the underlying data space (i.e., the original set) has been well represented by the selected representative items. Second, it enables the possibility for the user to retrieve items that are not in the representative result set by performing "zoom-in" operations—each representative item can be seen as the leader of a set of similar items, and by "zooming-in" to one of the leaders, similar items around the leader can be revealed.

Since the definition of coverage depends on the similarity between data objects, it is defined by the set of diversity constraints. In order to boost coverage, a set of appropriate diversity constraints must be defined. Below, we will discuss a general approach to determine such diversity constraints through an example.

Example 4. Consider a set of initial results R_Q that contains 100 items, each with two dimensions, $k = 30$, a single diversity constraint ψ that considers both dimensions, and the Euclidean distance as the diversity measure. Furthermore, assume that no partial diversity is allowed, meaning the final representative set produced must be a strict DT-k set. Obviously, whether it is possible to produce a DT-k set with 30 items is dependent on the definition of diversity constraints, such that if the diversity constraint consists of a diversity threshold that is beyond the maximum pair-wise distance between any pair of items in the original result set, then only a single item can be included in R, as the rest of the data items would be discarded due to the violation of the diversity constraint. Clearly, returning a result set with a single item when $k = 30$ is not ideal, and thus, the diversity threshold should be adjusted lower. In contrast, a minimum possible diversity threshold (i.e., 0) would lead to an arbitrary set of k items, which gives no guarantee of either diversity or coverage.

Clearly, from the example, an *optimal diversity constraint* should include a diversity threshold

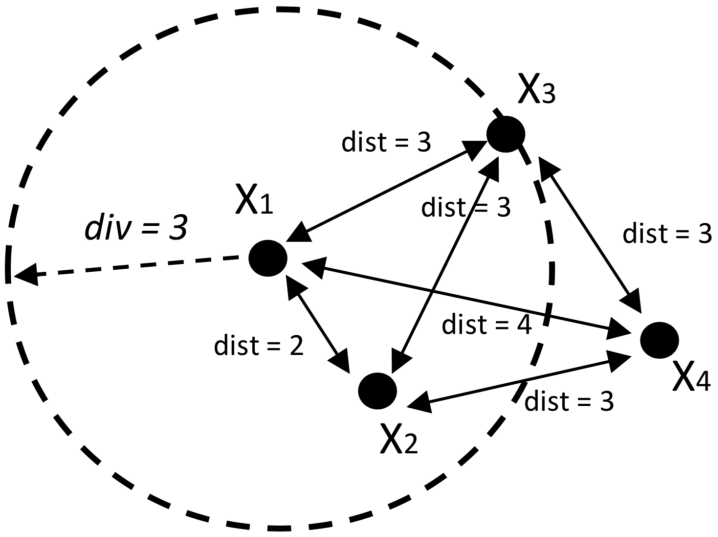

Figure 62: Illustration of the Optimal Radius, When $k = 2$.

that exhibits the following properties: (1) be as large as possible to improve the coverage; and (2) be small enough to allow a strictly diverse representative set (i.e., DT-k set) with k mutually dissimilar items being formed. With the these observations, we define the optimal diversity constraint as:

Definition 8. Optimal Diversity Constraint. *For a given set of item R, an integer number k, and a distance function $dist(x_1, x_2)$, an optimal diversity constraint must contain the largest possible distance threshold, denoted as div_{opt}, that exists between a pair of items in R that can be used to generate a DT-k subset R_Q from R, such that $|R_Q| >= k$ and no two item in R_Q are similar according to $dist(x_1, x_2)$ and div_{opt}.*

As illustrated in Figure 62, assume we have a set of points $P = \{x_1, ..., x_4\}$, such that each point consists of a 2-D coordinate and a diversity constraint ψ that consist of both dimensions and uses Euclidean distance. In such case, if $k = 2$, then $div = 3$ will be the div_{opt} (i.e., optimal diversity threshold) for ψ. If any value less than 3 is chosen to be the div_{opt}, then at least three points will remain after removing all similar points because only p_2 and p_3 are considered to be

Algorithm 7 SearchOptimalDiversityThreshold

Require:

A set of items R_Q, a size k, a set of attribute Δ, and a distance function $dist(x_1, x_2, \Delta)$

Ensure:

A diversity threshold div_{opt}

1: $S \leftarrow$ an initial item $x \in R_Q$

2: $div_{opt} \leftarrow \emptyset$

3: **while** $|S|$ ¡ k **do**

4: $x^* \leftarrow \text{argmax}_{x \in (R_Q - S)}(\min(dist(x, x_j, \Delta) : \forall x_j \in S))$

5: $S \leftarrow S \cup x^*$

6: **end while**

7: $\Theta \leftarrow$ minimum distance between any pair of items in S

8: $x^R \leftarrow \text{argmax}_{x \in (R_Q - S)}(\min(dist(x, x_j, \Delta) : \forall x_j \in S, s.t. dist(x, x_j, \Delta) < \Theta))$

9: $div_{opt} \leftarrow \min(dist(x^R, x_j, \Delta) : \forall x_j \in R_Q)$

10: **Return** div_{opt}

similar with respect to a diversity threshold of 2. If any value larger than 3 is chosen to be the div_{opt}, then only one point will remain in P after removing all similar points. Thus, 3 is the only option for div_{opt}, as no other distance threshold would be able to produce a result with three items. Unfortunately, finding the optimal diversity threshold for a given distance measure and a set of attributes is NP-hard.

According to Definition 5, two items x_i, x_j are dissimilar iff they fail to satisfy a diversity constant Ψ with a diversity threshold div and distance function $dist(x_1, x_2)$. Since an item, x_i can be included in a DT-k subset R *if and only if* x_i satisfies all diversity constraints with respect to other items in R, the maximum distance d between any pair of items in R increases along with the diversity thresholds inside each diversity constraints. Based on the Definition 8 of the optimal diversity constraint, the problem of finding the optimal diversity thresholds (i.e., div_{opt})

for a given diversity constraint can be mapped to the MaxMin Diversity Problem, which aims to select a representative subset $R \subseteq R_Q$, such that $|R| = k$, and the minimum distance between any pair of items in R is maximized. As the MaxMin Diversity problem has been previously proven to be an NP-hard problem [Carbonell and Goldstein, 1990], finding the optimal diversity threshold is also an NP-hard problem.

Inspired by the MaxMin Diversity problem, we adopt a greedy heuristic (Algorithm 7), which automatically computes an approximation of the optimal diversity thresholds for a given set of diversity constraints. As illustrated in Algorithm 7, we first find a subset $S \subseteq R_Q$ that maximize the minimum distance between items in R (Lines 5 - 7). Then, we generate the optimal diversity threshold by comparing the pair-wise distance of all items that are in R_Q but not in S (Lines 8 - 11). The optimal diversity threshold is defined as the largest distance between a pair of items in R_Q that is smaller than the minimum distance between any pair of items in R. As proven in [Ravi et al., 1991], the result produced by this greedy heuristic has a $\frac{1}{2}$ approximation of the optimal solution and a quadratic complexity, and no other polynomial algorithm can provide a better guarantee.

6.5 Experimental Evaluation

To study the effectiveness of our PrefDiv and PrefDiv-PR algorithms, we compare them to the two most effective diversified top-k algorithms, namely *Swap* [Yu et al., 2009] and *MMR* [Carbonell and Goldstein, 1998], as suggested in [Thang et al., 2015]. We also compare them to four diversity-focused algorithms, *MaxSum, MaxMin, K-Medoids*, and *DisC Diversity*, to assess how well diversity has been preserved when relevance is taken into account. All the algorithms in our evaluation are discussed in Section 2.6.

6.5.1 Experimental Testbed

Environment We implemented all of the algorithms with JDK 8.0 on an Intel machine with Core i7 2.5Ghz CPUs, 16GB RAM, and 512GB SSD.

Algorithms We implemented MMR and Swap based on their published descriptions [Carbonell and Goldstein, 1998] and [Yu et al., 2009], respectively. The MaxMin and MaxSum algorithms used in our experiments are based on Definitions 1 and 2 (in Section 2.6), respectively, and DisC Diversity is taken directly from the original author [Drosou and Pitoura, 2012b]. However, DisC Diversity is not a top-k bounded algorithm, and the size of the result set that DisC Diversity produces is heavily dependent on the radius. To allow for a comparison, we modified the DisC Diversity to stop when the size of the result set equals k. We also included one well-known clustering algorithm, K-Medoids [Park and Jun, 2009], which aims to group a set of data objects into clusters through some distance measure, so objects within a cluster are close to each other, and objects outside of the cluster are unrelated to the objects inside the cluster.

In our experiment, we implemented K-Medoids based on [Park and Jun, 2009]. Since K-Medoids does not capture the relevance in any regard, we improved the performance of K-Medoids in balancing the relevance *vs.* diversity trade-off by choosing the object with the highest intensity value as the final recommendation from each of its k clusters. This improvement significantly enhances the performance of the K-Medoids with respect to relevance while exhibiting the minimum decrease in diversity.

Parameters Most of the diversification techniques involved in our experiments require some parameters since finding the best parameters for each technique that are optimum under all situations would be too difficult. For the purpose of comparison, in all of our experiments, there is only one diversity constraint ψ for any given set of experiments, which is used by all algorithms that require a diversity constraint during its execution. We fixed the diversity threshold div used in ψ for each set of the experiments, which is computed with the optimal radius by Algorithm 7. All other parameters except ψ and r are fixed for all runs and adjusted according to the suggestion of the original authors, or based on the best overall performance. For MMR, we set $\lambda = 0.3$, for

Swap, we set the $UB = 0.1$, and for PrefDiv, we set $A = 0.6$.

Datasets We ran our tests on two real-world datasets: Cameras [Drosou and Pitoura, 2012b], and Foursquare. We selected these datasets in order to experiment with two different distance functions, *Hamming* with Cameras and *Euclidean* with Foursquare. The Cameras dataset consists of 579 records and seven attributes per record. The Foursquare dataset is collected from the major location-based social network, Foursquare. We obtained real-life user preferences, used Foursquare's public venue API, and queried information for 14,011,045 venues. In order to build realistic user profiles for our evaluations, we used a dataset collected by Cheng et al. [Cheng et al., 2011] that includes geo-tagged user-generated content from a variety of social media between September 2010 and January 2011. This dataset includes 11,726,632 check-ins generated by 188,450 users. Accordingly, each reading in our Foursquare dataset has the following tuple format: ¡ID, latitude, longitude, # check-ins, # unique users¿. In our experiments, we consider only data items (i.e., venues) from New York City (NYC), which consists of 10912 items, and San Francisco (SF), which consists of 7859 items. In our experiments, we will be using NYC and SF to denote Foursquare's NYC data and Foursquare's SF data, respectively.

User Preferences The intensity values (I) for each individual dataset is generated as follows:

For the Cameras dataset, we generated 100 different sets of user preferences, such that in each profile, the preference intensity value for each individual camera is generated based on a uniform distribution, and each individual user preference is represented as one unique query.

For the Foursquare dataset, we obtained the *real-life* user preferences based on the hierarchy of the Foursquare dataset, such that every individual venue v in the dataset is associated with a type T_v. For example, an Italian restaurant belongs to the category "Italian restaurant", which can belong to the higher level category "Restaurants", which can itself belong to the category "Food", and so on. In order to build highly personalized and specific profiles, we use the bottom layer of the hierarchy, as well as the specific venues visited. In particular, given the set of check-ins $\mathcal{C}_u$ of user u, we build a hierarchical profile $\mathcal{P}$ where at the top level, the preferences of the user are expressed in terms of the (normalized) frequencies of this user's visitations with respect to the types of venues. The

second layer of the user profiles further provides the normalized frequencies of venues for the different types of locations visited by u. Since our user profile is sparsely gathered during a short period of time, to resemble a real-world user profile, we merged the 1000 sparse Foursquare user profiles to create one superuser profile. We performed our experiments by randomly selecting 50 query points from each city (100 query points in total). For each query point, we considered all venues located within a 1.5 kilometer radius of the query location.

6.5.2 Evaluation Metrics

In our experimental evaluation, we evaluate the performance of all models based on three well-known and commonly used metrics: *Normalized Relevance* [Tong et al., 2011], *Coverage* [Drosou and Pitoura, 2012b] (Definition 7), and *Execution Time*. Note, there are two other commonly used metrics for evaluating ranking algorithms, such as DCG and Spearman rho. However, both metrics focus on measuring the correctness of the order of the results produced by the ranking algorithm. Thus, they are not the ideal evaluation metrics for evaluating the effectiveness of result diversification algorithms. As stated in previous sections, our proposed PrefDiv algorithms are post-processing steps of initial query results, which do not impact the relative order of the original result set. In other words, the produced representative result set of PrefDiv algorithms essentially follows the original order of the initial results. Thus, metrics that focus on the correctness of the ranking order do not fit the context of this evaluation.

Definition 9. Normalized Relevance. *Let S be a set of items and $S_k^* \subseteq S$ such that $|S_k^*| = k$. The Normalized Relevance of a subset S_k^* is defined as the sum of the intensity value of items in S_k^* over the sum of k items with highest intensity value in S.*

$$nRev(S_k^*) = \frac{\sum_{x \in S_k^*} I_x}{\max_{S_k \subseteq S, |S_k|=k} \sum_{x \in S_k} I_x} \tag{6.2}$$

In order to calculate the coverage for the Foursquare dataset, given that it is difficult to find a fixed radius that would work with any query location, we calculate the coverage with respect to

the optimal radius generated by Algorithms 7 for every output size k. In the case of a Camera dataset, where the hamming distance is employed, the coverage is calculated with a fixed diversity threshold/radius $div = 3$ (which is the mid-point of the maximum distance allowed). For a fair comparison, all algorithms are evaluated with respect to the same diversity threshold/radius. Note that Normalized Relevance and Normalized Intensity Value would be used interchangeably in the following sections.

6.5.3 Experimental Results

6.5.3.1 Normalized Relevance As demonstrated in Figures 63, 64, and 69, we can see a clear separation between two groups of algorithms for all datasets, where PrefDiv, PrefDiv-PR, Swap, MMR, and K-Medoids tend to group together, and MaxMin, MaxSum, and DisC Diversity form another group. The reason for this is that the second group does not take relevance into account; hence, it would be unlikely for them to retrieve a representative subset that has a high total intensity value. In contrast, the first group of algorithms takes relevance into account, and, as such, it achieves a significantly higher performance in terms of retrieving relevant items.

6.5.3.2 Coverage Figures 65, 66, and 71 show that our PrefDiv and PrefDiv-PR exhibit better coverage on average when compared with MMR and Swap by 20% and 42%, respectively. The reason could be because both MMR and Swap optimize dissimilarity as their definition of diversity. In contrast, both PrefDiv and PrefDiv-PR are coverage-aware algorithms, which seek an optimal radius that directly improves the coverage of the representative result set. Therefore, both PrefDiv and PrefDiv-PR can perform much better than Swap and MMR. We also observed that on average PrefDiv is able to outperform MaxSum in terms of coverage by 160%, which could be explained because MaxSum as pure dissimilarity-based algorithm fails to cover the entire space of the dataset. K-Medoids demonstrates good coverages for both datasets. However, both PrefDiv and PrefDiv-PR still exhibit slightly better coverage in general.

The MaxMin algorithm performs well in terms of the Foursquare dataset, although the per-

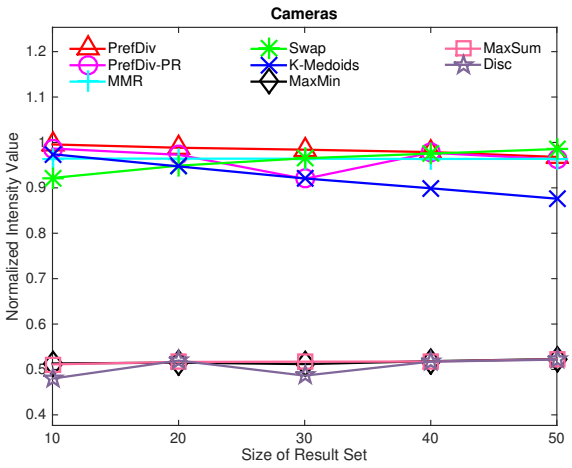

Figure 63: Normalized Intensive Value of Cameras

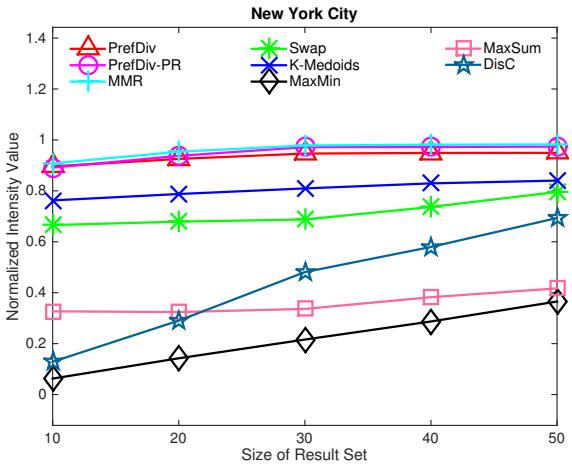

Figure 64: Normalized Intensive Value of NYC

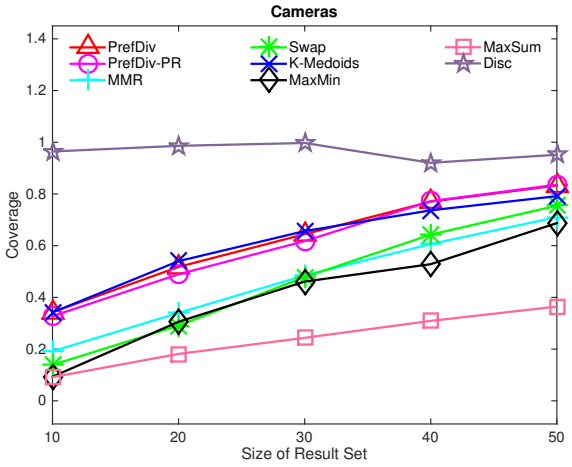

Figure 65: Coverage of Cameras.

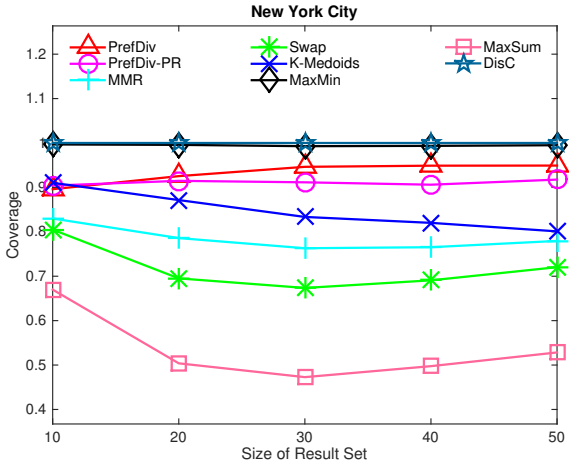

Figure 66: Coverage of NYC

formance dropped significantly for the cameras dataset. The reason could be because, in the Foursquare dataset, the average number of venues around each query point is about 90 venues. In contrast, the Cameras dataset consists of 579 tuples. This shows that MaxMin is able to obtain good coverage with a relatively small dataset and Euclidean distance that takes a wide range of values as distance, but fails to cover the space with large datasets and hamming distance that

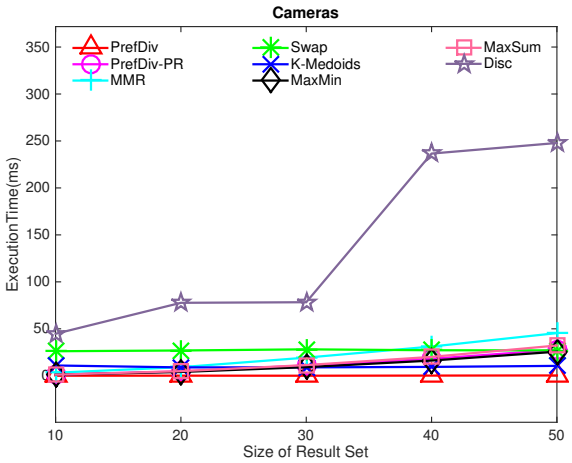

Figure 67: Execution Time of Cameras

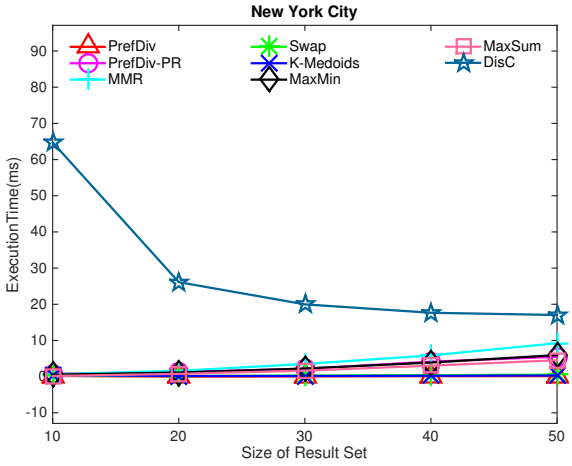

Figure 68: Execution Time of NYC

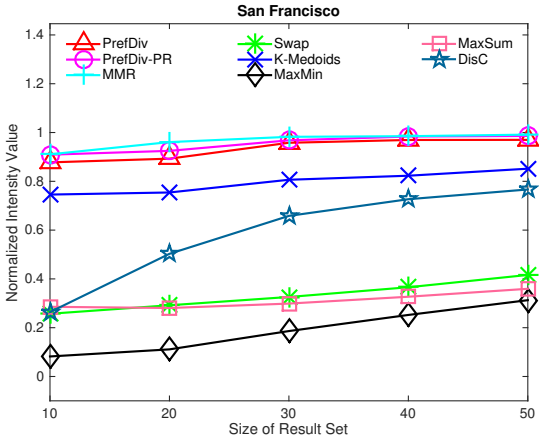

Figure 69: Normalized Intensive Value of Foursquare's San Francisco Data

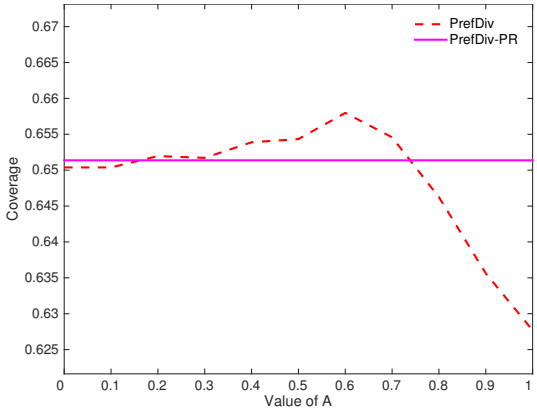

Figure 70: Result Set Coverage, With the Optimal Radius and K = 30.

only takes the number of attributes + 1 distinct values as distance. In both datasets, DisC Diversity demonstrated the highest coverage, which is to be expected since DisC Diversity is the only algorithm in the experiment that directly optimizes coverage as the only objective.

Another interesting observation is that, in the Foursquare dataset, except PrefDiv, PrefDiv-

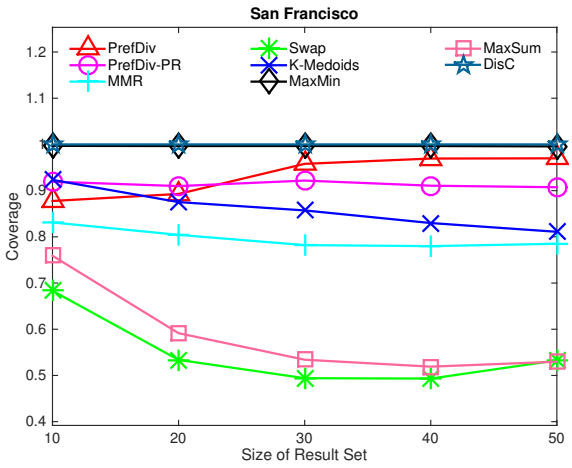

Figure 71: Coverage of Foursquare's San Francisco Data

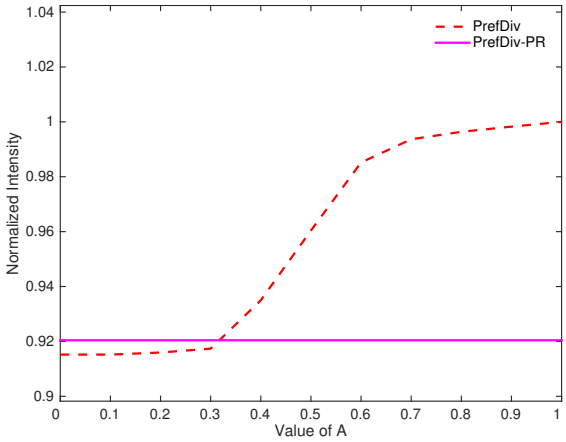

Figure 72: Result Set Normalized Relevance, With the Optimal Radius and K = 30.

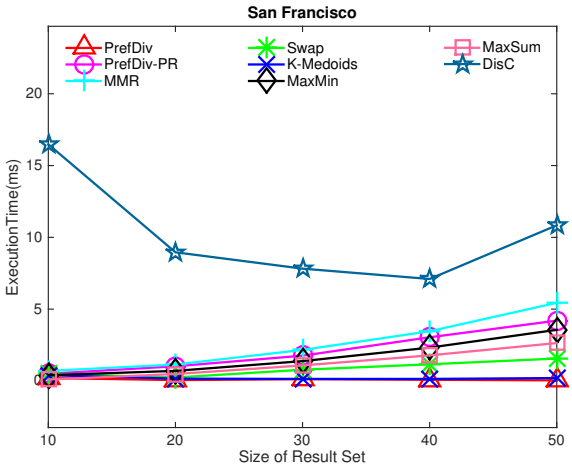

Figure 73: Execution Time of Foursquare's San Francisco Data

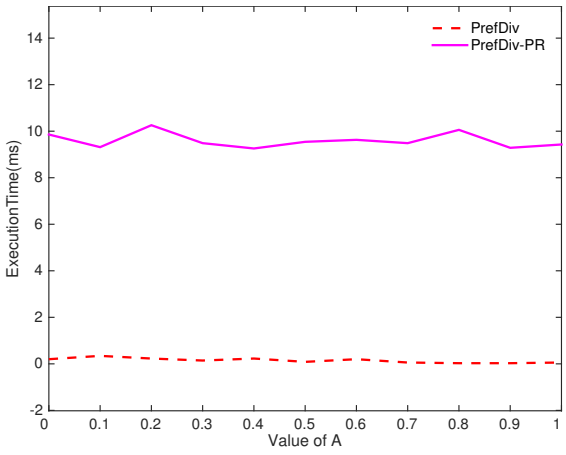

Figure 74: Result Set Execution Time, With the Optimal Radius and K = 30.

PR, and DisC Diversity, other algorithms appear to have a drop in coverage when the value of k increases, although, in general, with the increase in result size, the coverage should increase as well. The reason for such behavior is that in the Foursquare dataset, we employed the optimal radius as the criterion for determining the similarity between items. Therefore, with the increase

in result size, the optimal radius becomes smaller, thus leading to a decrease of coverage for some algorithms.

6.5.3.3 Execution Time We have measured the execution time required by all algorithms. As shown in Figures 67, 68, and 73, our PrefDiv and PrefDiv-PR appears to be the overall fastest algorithm when compared to all other alternatives. In general, PrefDiv and PrefDiv-PR perform near identically in terms of runtime, which is expected as the additional computation overhead introduced in PrefDiv-PR is negligible. To illustrate the efficiency of our proposed greedy heuristic for searching the optimal diversity threshold, we have included its runtime in the PrefDiv-PR, so the runtime difference between PrefDiv and PrefDiv-PR reflects the runtime of the search optimal diversity threshold algorithm. With that in mind, PrefDiv-PR is still on average faster than both MMR and Swap by up to 72% and 116%, respectively. Specifically, in the Camera dataset, PrefDiv is able to execute 57 times faster than K-Medoid, 127 times faster than MMR, and 159 times faster than Swap. In the Foursquare dataset, most algorithms tend to be faster when compared with Cameras, because the number of venues near each query point in Foursquare is much smaller than in the Cameras dataset. However, we still observed that, on average, PrefDiv is able to outperform MMR and Swap by 30 and 36 times, respectively. When compared to K-Medoid, which is also very efficient when dealing with this type of dataset, PrefDiv still appears to be 2.7 times faster than K-Medoid on average. As mentioned previously, if the optimal radius for most frequent queries is stored, PrefDiv-PR would not need to calculate the optimal radius for these queries again. Furthermore, for the fairness of the comparison, all of the algorithms run in a single-threaded mode. Since the optimal radius computation method that we adopted is fully parallelizable, it can take advantage of the modern multi-threaded CPU architecture to speed up the computations. In fact, we have observed linear speed up with respect to the number of CPU cores in the system up to 16 cores (the highest we have experimented). One interesting remark here is that, in the Foursquare dataset, the execution time of DisC Diversity drops when k increases from 10 to 20. The reason is that the runtime of DisC Diversity is also affected by the length of the radius, therefore, with

smaller output sizes, the optimal radius will become larger, which leads to the relativity longer execution time of DisC.

6.5.3.4 Parameter A of PrefDiv As illustrated in Figures 65 to 69, a performance difference between PrefDiv and PrefDiv-PR exists due to the existence of the accuracy parameter A in the PrefDiv algorithm. In this section, we conducted an experiment to study the effect of parameter A in PrefDiv with the Camera dataset and $k = 30$. As shown in Figures 70, 72, and 74, when A increases from 0 to 1, we observed an improvement of normalized relevance, albeit with a decrease in coverage. This is as expected since, with higher values of A, PrefDiv will select more relevant items, and with lower values of A, more diverse items will be selected that lead to an increase in coverage. However, this would be at the expense of lower relevance. The execution time appears to be stable regardless of the value of A. This is because, for each iteration, PrefDiv only requires a tiny amount of execution time. Therefore, the additional iterations introduced by the low value of A would not have a large impact on the overall runtime.

6.5.3.5 Relevancy vs. Diversity Lastly, as a summary, we present three scatterplots that capture the trade-off between relevance and diversity. Each point in Figures 75 and 76 are corresponding to the average of over 50 different query locations with one value of k, and each point in Figures 77 are corresponding to the average of over 100 different user profiles with one value of k. As shown in the figures, we have Normalized Intensity Value as the y-axis and Coverage as the x-axis. Algorithms located in the upper left corner of the figure exhibit the best coverage result, while those in the lower right corner have the highest relevance scores. As we can observe, both PrefDiv and PrefDiv-PR are located towards the upper right corner (circled) for all three scatter plots, which indicates that both PrefDiv and PrefDiv-PR exhibit better ability to handle the trade-off between relevance and diversity with respect to both datasets and distance measures. One may notice that in the Camera dataset, the advantage of PrefDiv and PrefDiv-PR with respect to other alternatives is relatively smaller compared to that in the Foursquare dataset. This is because the

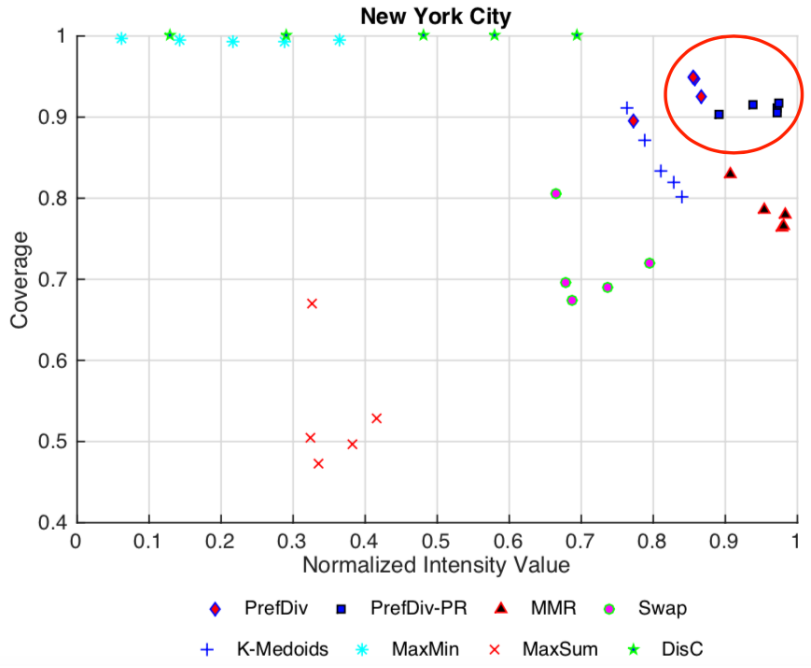

Figure 75: Relevance VS. Diversity (NYC).

Cameras dataset uses the Hamming distance as the distance measure, which has a much smaller domain than the Euclidean distance used in the Foursquare dataset, thus weakening the benefit of the optimal diversity threshold. These results also indicate that the relational proportionality introduced in PrefDiv-PR does effectively improve the quality of the result, since PrefDiv-PR is able to outperform (although slightly) the PrefDiv with manually configured relevance parameter A.

6.5.3.6 Additional Observations Despite the fact that both PrefDiv and PrefDiv-PR run up to 159 times faster than other alternatives, the greedy heuristic (Algorithm 7) that we proposed for finding the optimal diversity threshold/radius runs at a quadratic time complexity, and thus, it is much slower. Although it is not required to run this heuristic before each execution of the PrefDiv/PrefDiv-PR, it certainly helps improve its performance. However, this is not an issue with our PrefDiv/PrefDiv-PR algorithm because other algorithms (e.g., DisC Diversity) also benefit

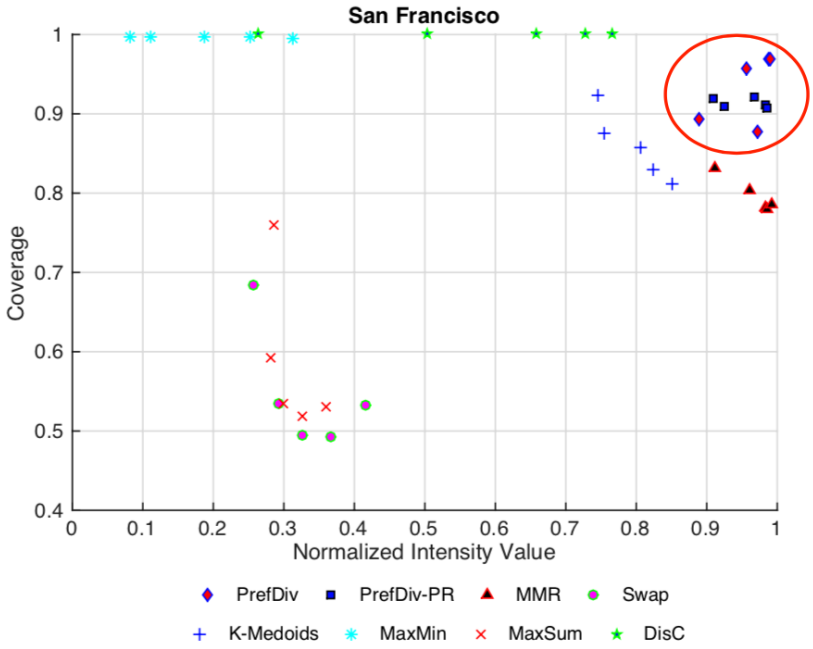

Figure 76: Relevance VS. Diversity (SF).

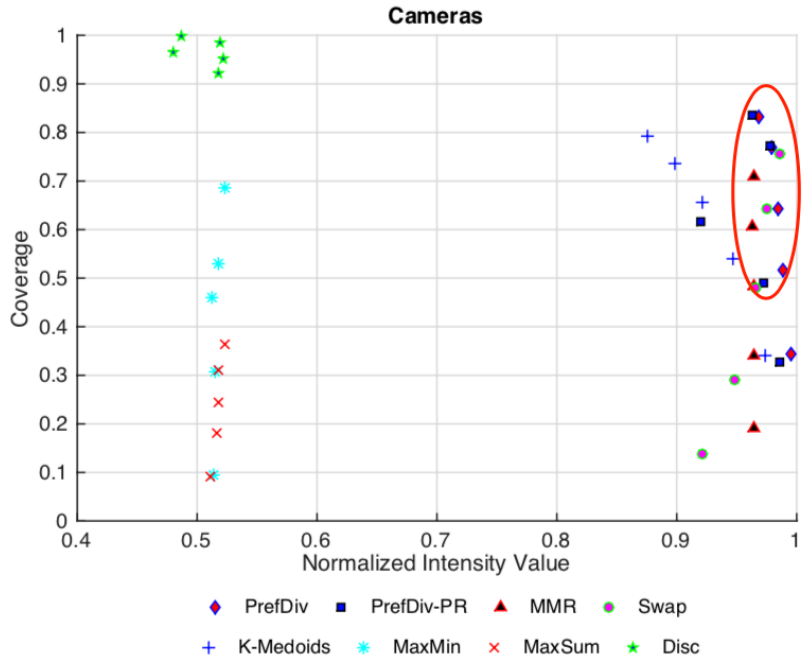

Figure 77: Relevance VS. Diversity (Cameras).

from the optimal diversity threshold as much as PrefDiv/PrefDiv-PR.

Fortunately, this greedy heuristic only needs to run once for each query, and thus, it can simply be cached to boost the runtime of frequent queries significantly.

6.6 Impacts of PrefDiv

The ability of PrefDiv to produce an informative and diversified representative subset of results significantly benefits the Example-driven Exploration as it helps to overcome the scalability issue from a human point of view and thus, enables users to effectively interpret and consume the exploration results. However, the efficiency of PrefDiv and its ability to balance the trade-offs between relevance, diversity, and coverage can also benefit the design of other real-world systems that need to produce highly informative representative subsets or require interactive efficiency in producing the representative results (e.g., [Raghu et al., 2017, Ge et al., 2016a, Ge et al., 2017b, Ge et al., 2017a, Raghu et al., 2019]). One example is a novel mobile recommendation service that provides a set of diverse points-of-interest (POI's) recommendations [Ge et al., 2016a, Ge et al., 2017b, Ge et al., 2017a], where the interactive efficiency has been weighted equally important as the quality of the produced recommendations. Other examples are in the scientific domain, and dimensionality reduction, in which PrefDiv has been employ as a novel way to select subsets of highly informative dimensions for high-dimensional gene expression datasets [Raghu et al., 2017, Raghu et al., 2019]. Those selected dimensions will then be used to enable effective downstream analysis in a variety of medical and bioinformatics research.

6.7 Summary

Scalability from a human point of view is a very challenging problem for Example-driven Exploration as it consists of finding the perfect balance between the conflicting objectives of rele-

vance and diversity. To produce a set of representative objects, traditional top-k result diversification approaches focus on producing a subset of results that balance the trade-off between selecting highly relevant items and items that are dissimilar with respect to each other. In order to achieve the above-mentioned objectives, most algorithms rely on a number of tunable control parameters, making them harder to configure (and be adopted). Coverage is another important factor of diversity, which has been mostly ignored in previous top-k result diversification algorithms.

In this chapter, we addressed these problems and proposed an efficient online solution called *Preferential Diversity* (PrefDiv). PrefDiv produces a set of high-quality representative items from a large set of initial answers, where each representative item is chosen to optimize both the *relevance* and *diversity* (i.e., dissimilarity, and coverage). We also proposed a number of optimizations that further improve PrefDiv's usability, efficiency, and effectiveness.

We theoretically analyzed and experimentally compared our algorithms to the state-of-the-art, top-k diversification algorithms. Our evaluation showed that our algorithms achieve similar performance in terms of normalized relevance but outperform the state-of-the-art algorithms in terms of coverage by a noticeable margin while achieving a speedup of the runtime up to two orders of magnitude.

7.0 Conclusions

7.1 Our Contributions

Exploration over large datasets is a key first step in data analysis, as users may be unfamiliar with the underlying database schema and unable to construct precise queries that represent their interests. Such data exploration task usually involves forming and executing numerous ad-hoc queries, which requires a considerable amount of time and human effort. In this thesis, we focused on a novel interactive data exploration approach, called Example-driven Exploration, which has generated a lot of attention for its capabilities to identify effectively high-value content from the data that are often hidden using the traditional search methods without the need to specify any form of queries.

Our hypothesis has been that in order to efficiently and effectively guide users towards unearthing valuable insights from large datasets, data exploration must provide: interactive efficiency, analysis-based effectiveness, and user-in-the-loop engagement. This led us to study and propose solutions to enable *effective, scalable,* and *generic Example-driven Exploration*, where the objective is to minimize user effort in exploration and searching desired information from both big structured and unstructured data. The main contributions of this thesis as illustrated in Figure 78 are summarized as follows:

- We proposed REQUEST, a novel framework that is designed to minimize the human effort and enable both effective and efficient data exploration. REQUEST supports the Example-driven Exploration style of data exploration by integrating two key components: 1) Data Reduction and 2) Query Selection. As instances of the REQUEST framework, we developed several highly scalable schemes, which employ active learning techniques and provide different levels of efficiency and effectiveness as guided by the user's preferences. Our results, on real-world datasets from Sloan Digital Sky Survey, show that our schemes on average require 1-2 orders of

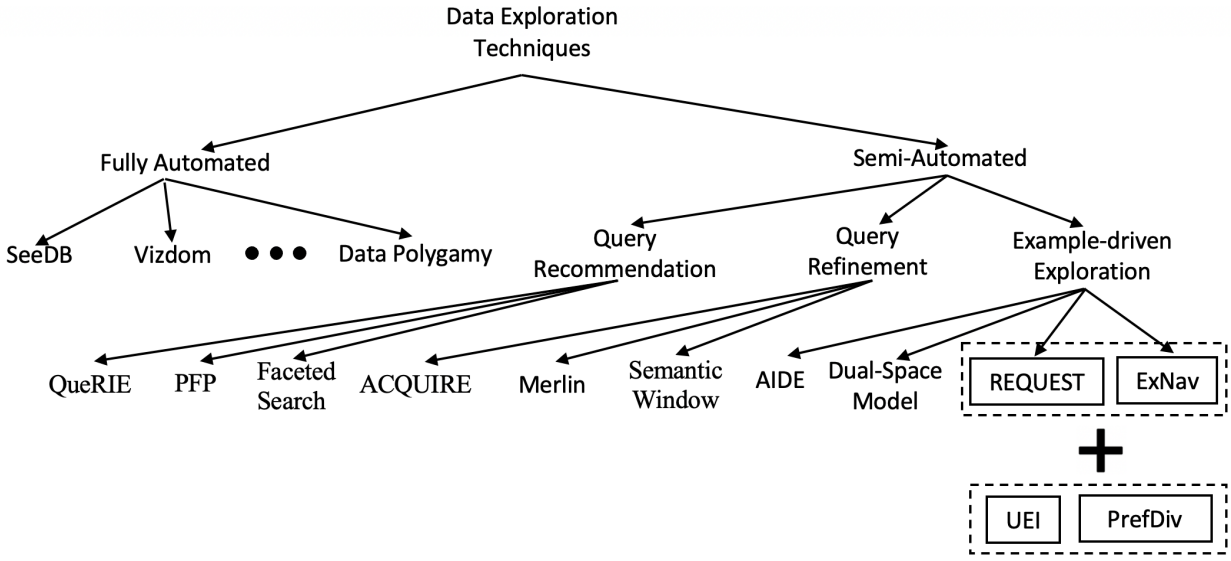

Figure 78: Illustration of the Taxonomy of Data Exploration Techniques and Our Contributions. Each Left Node Refers to One Example Data Exploration System. Our Contributions Are Marked With Squares.

magnitude fewer feedback questions than the random baseline and 3-16× fewer questions than the state-of-the-art while maintaining interactive response time. Moreover, our schemes are able to construct, with high accuracy, queries that are often undetectable by current techniques.

- We devised a novel indexing mechanism, called *Uncertainty Estimation Index* (`UEI`), which supports the interactivity and scalability of the active learning-based interactive data exploration systems. `UEI` combines hierarchical in-memory indexing with columnar and inverted-indexing based secondary storage mechanism. It achieves scalability and efficiency through a dynamic estimation of the set of data that are most beneficial to the current exploration. By intelligently manage the in-memory cache, `UEI` enables active learning-based interactive data exploration systems to scale beyond the main memory restriction while maintains the desired accuracy and convergency speed. We experimentally evaluated `UEI` using a state-of-the-art interactive data exploration system with two schemes, one incorporating `UEI` and one utilizing a standard DBMS. We measure the efficiency of the proposed solution using a large real-world

dataset placed on the secondary storage. Our experiments show that (1) `UEI` version outperforms the DBMS one by providing more than 50x runtime efficiency when the size of the dataset exceeds the main memory capacity, and (2) is capable of achieving sub-second interactive response time for data that is 100 times larger than the available memory while achieving the desired exploration accuracy and effectiveness.

- We presented a novel data exploration framework, namely ExNav (**Ex**ploration **Nav**igator), which enables the user to effortlessly explore the world of unstructured data for insights that are often unreachable from traditional search and exploration methods. In particular, we exploit the space of advanced machine learning, data embedding, and active learning algorithms to design effective exploration and space pruning approaches tailored for unstructured datasets. Our experimental evaluation using multiple real-world unstructured datasets (i.e., text, image, and graph) show that ExNav can reduce users' effort by up to 9x while still achieving the same accuracy as the state-of-the-art alternative. Moreover, ExNav is also able to identify relevant data items that are often undetectable by current techniques, even when a large number of samples are explored.
- We formulated an extended version of the result diversification problem, considering three objectives—relevance, diversity, and coverage—and present a novel approach named PrefDiv that produces better-diversified results. Our approach PrefDiv takes a large set of possible answers generated from a user query and outputs a representative subset of results that are highly ranked according to the preference of the user. The data items contained in the representative set are diverse, such that each item is different from the rest and provides good coverage of the underlying aspects of the original results. Our approach also suggests a set of appropriate parameters for each user query to achieve a balance between our conflicting objectives and is efficient enough to ensure an interactive experience. We study the complexity of our algorithms and experimentally evaluate them in terms of normalized relevance, coverage, and execution time. Our evaluation indicates a speedup of up to $159x$, and outperforms the state-of-the-art algorithms on multiple fronts.

7.2 Open Questions

We have proposed approaches to perform efficient and scalable Example-driven Exploration that assist the user in effectively finding the desired data objects with the need to specify any form of explicit queries. Our approaches show superior performance compared to existing approaches in multiple regards. However, there are still multiple challenges and open problems that require further investigation. In the following, we briefly summarize some of these interesting new directions.

- As discussed in the Introduction and illustrated in previous works of literature (e.g., [Juslin et al., 1998, Griffin and Tversky, 1992]), human labels are subject to noise (i.e., labeling noise). Consequently, the performance of the data exploration system will degrade greatly without having methods specifically designed to handle labeling noise. However, it is hard to estimate labeling noise because it does not depend on model or training data. To address this challenge, one idea is to use *soft labels* with more than two possible categories of relevance as follows: First, using ensemble methods like *Bagging and Randomization* [Dietterich et al., 2000] to average out the influence of labeling noise. Second, using a labeling system with multiple categories [Xue and Hauskrecht, 2017] to reduce the range of influence of labeling noise. Third, estimating the labeling noise explicitly and acquiring labels multiple times for examples with high noise [Sheng et al., 2008]
- The number of region pruning questions asked by the current approaches grows exponentially with the dimensionality, which renders its usage impractical for datasets with high dimensionality. In addition, a question with too many features involved will also become hard, if not impossible, for the user to perceive. Based on this observation, reducing the number of dimensions involved in the exploration space becomes a promising solution. However, two obstacles still stand in the way of solving the problem. First, the number of dimensions needs to be reduced below the user's perception limit. Second, the retained dimensions should still be meaningful to the user. With these in mind, it will be beneficial to Example-driven Exploration

to design a novel feature selection method, which will help the user to find the combination of features (i.e., dimensions) that match his/her interest.

Inspired by previous work on view recommendation [Ehsan et al., 2016, Ehsan et al., 2017] (Section 4.2), one idea is to create a score function that captures the importance of a combination of features. The more dataset information that is preserved by a combination, the higher its score should be. Consequently, it would be useful to devise an algorithm to obtain the set of k most informative combination of features, out of the huge amount of possible combinations, to be presented to the user. Using this set, the user will be able to effectively define the k attribute exploration space that is most suitable with respect to his/her exploration task. This new functionality will likely to improve the performance of query selection, reducing the time of selecting useful examples while preserving the dataset information to the greatest extent.

- The existing Example-driven Exploration system essentially relies on the user to determine the termination of an exploration task. Thus, the user is required to frequently examine the exploratory query that is currently constructed by the platform to determine whether or not the exploratory query is sufficient (i.e., accurate) enough for his/her exploration task. Such a requirement can dramatically increase the effort of a user during the exploration tasks, thus harming the effectiveness of the exploration platform. Furthermore, as users do not have a clear idea of what they are looking for, it is usually difficult for them to determine the accuracy of the current exploratory query correctly. Therefore, one way to overcome such a challenge is to design methods that can indicate the current exploration progress to reduce the user's effort in determining the termination of the exploration task, and to increase the accuracy of the final exploratory query that is aligned with the user's expectations. There are several approaches (e.g., [El-Yaniv and Wiener, 2012, Hanneke, 2011, Hanneke, 2016]) to trace the convergence of an active learner that can be used as an indication for determining the termination of an active learning task. However, these approaches cannot be adopted to solve the above challenge, as they are focused on error rate as the accuracy measure of a classification model, which is inappropriate for data exploration. Recall that the F-measure used to measure accuracy in data

exploration systems. Furthermore, convergence detection methods that rely on a single measure of the classification model can easily fall into local optimal, and in turn, fail to detect the true convergence. Therefore, an efficient and robust model that combines multiple measures of the predictive model (e.g., model change rate and approximated maximum uncertainty) to capture the trans of the convergence for data exploration tasks will be beneficial to strengthen the usability of Example-driven Exploration approaches.

7.3 Broader Impact

Intelligent and efficient means of data exploration has far-reaching implications for a large variety of data-intensive application areas such as commerce, intelligent transportation, environmental modeling, astronomy, learning analytics, financial risk assessments, health and population planning, and even on daily life activities in the current data-rich society. In this thesis, we presented our novel solutions that aim at addressing a fundamental roadblock in the use of large datasets, namely the ability to efficiently explore and understand the value and relevance of unfamiliar data before it is submitted for intensive analytical processes. Consequently, our solutions on effective Example-driven Exploration can enable more opportunities in developing cutting-edge technologies that have the capacity to make a wide-ranging impact on the way data is utilized and consumed in business, social, and scientific settings.

In addition to facilitating the Example-driven Exploration, some of our proposed techniques can be leveraged to benefit other active learning-based or data-driven applications. For instance, our indexing technique UEI can be leveraged in a wide range of active learning-based human-in-the-loop systems (e.g., [Qian et al., 2019, Bhattacharjee et al., 2017, Qian et al., 2020]) to effectively reduce the memory requirement, and in turn, enable these systems to scale out for larger datasets. Furthermore, our result refinement technique PrefDiv and its ability to balance the trade-offs between relevance, diversity, and coverage can also benefit the design of other real-world systems and scientific applications that need to produce highly informative rep-

resentative subsets or require interactive efficiency in producing the representative results (e.g., [Raghu et al., 2017, Ge et al., 2016a, Ge et al., 2017b, Ge et al., 2017a, Raghu et al., 2019]).

Bibliography

[ima, 2021] (2021). Caltech-256 object category dataset. https://authors.library.caltech.edu/7694/.

[tex, 2021] (2021). CBC coronavirus news dataset. https://www.kaggle.com/ryanxjhan/cbc-news-coronavirus-articles-march-26.

[Ind, 2021] (2021). Independent set (graph theory). https://en.wikipedia.org/wiki/Independent_set_(graph_theory).

[gra, 2021] (2021). Mashup PPI dataset. http://cb.csail.mit.edu/cb/mashup/.

[sds, 2021] (2021). Sdss samples queries. http://cas.sdss.org/dr4/en/help/docs/realquery.asp.

[Abe and Mamitsuka, 1998] Abe, N. and Mamitsuka, H. (1998). Query learning strategies using boosting and bagging. In *International Conference on Machine Learning (ICML)*, pages 1–9.

[Agrawal et al., 2009] Agrawal, R., Gollapudi, S., Halverson, A., and Ieong, S. (2009). Diversifying search results. In *ACM International Web Search and Data Mining*, pages 5–14.

[Anagnostopoulos et al., 2010] Anagnostopoulos, A., Becchetti, L., Castillo, C., and Gionis, A. (2010). An optimization framework for query recommendation. In *ACM International Web Search and Data Mining (WSDM)*, pages 161–170.

[Angel and Koudas, 2011] Angel, A. and Koudas, N. (2011). Efficient diversity-aware search. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 781–792.

[Bhattacharjee et al., 2017] Bhattacharjee, S. D., Talukder, A., and Balantrapu, B. V. (2017). Active learning based news veracity detection with feature weighting and deep-shallow fusion. In *IEEE International Conference on Big Data*, pages 556–565.

[Binnig et al., 2017] Binnig, C., Stefani, L. D., Kraska, T., Upfal, E., Zgraggen, E., and Zhao, Z. (2017). Towards sustainable insights or why polygamy is bad for you. In *Conference on Innovative Data Systems Research (CIDR)*.

[Brachman et al., 2012] Brachman, R. J., Cohen, W. W., and Dietterich, T. (2012). *Synthesis Lectures on Artificial Intelligence and Machine Learning*.

[Briggs et al., 2012] Briggs, F., Fern, X. Z., and Raich, R. (2012). Rank-loss support instance machines for MIML instance annotation. In *ACM SIGKDD Conference on Special Interest Group on Knowledge Discovery and Data Mining*, pages 534–542.

[Cai et al., 2013] Cai, W., Zhang, Y., and Zhou, J. (2013). Maximizing expected model change for active learning in regression. In *IEEE International Conference on Data Engineering (ICDE)*, pages 51–60.

[Carbonell and Goldstein, 1990] Carbonell, J. and Goldstein, J. (1990). The discrete p-dispersion problem. *European Journal of Operational Research*, 46:48–60.

[Carbonell and Goldstein, 1998] Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *ACM SIGIR*, pages 335–336.

[Carbonneau et al., 2019] Carbonneau, M., Granger, E., and Gagnon, G. (2019). Bag-level aggregation for multiple-instance active learning in instance classification problems. *IEEE Transactions on Neural Networks and Learning Systems*, 30(5):1441–1451.

[Çetintemel et al., 2013] Çetintemel, U. et al. (2013). Query steering for interactive data exploration. In *CIDR*.

[Celis et al., 2018] Celis, L. E., Straszak, D., and Vishnoi, N. K. (2018). Ranking with fairness constraints. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 28:1–28:15.

[Chatzigeorgakidis et al., 2019] Chatzigeorgakidis, G., Skoutas, D., Patroumpas, K., Palpanas, T., Athanasiou, S., and Skiadopoulos, S. (2019). Local similarity search on geolocated time series using hybrid indexing. In *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 179–188.

[Cheng et al., 2011] Cheng, Z., Caverlee, J., Lee, K., and Sui, D. (2011). Exploring millions of footprints in location sharing services. In *International AAAI Conference on Web and Social Media (ICWSM)*.

[Chirigati et al., 2016] Chirigati, F., Doraiswamy, H., Damoulas, T., and Freire, J. (2016). Data polygamy: The many-many relationships among urban spatio-temporal data sets. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 1011–1025.

[Cho et al., 2016] Cho, H., Berger, B., and Peng, J. (2016). Compact integration of multi-network topology for functional analysis of genes. *Cell systems*, 3(6):540–548.

[Clarke et al., 2008] Clarke, C. L., Kolla, M., Cormack, G. V., Vechtomova, O., Ashkan, A., Büttche, S., and MacKinnon, I. (2008). Novelty and diversity in information retrieval evaluation. In *ACM SIGIR*, pages 659–666.

[Crotty et al., 2015] Crotty, A., Galakatos, A., Zgraggen, E., Binnig, C., and Kraska, T. (2015). Vizdom: Interactive analytics through pen and touch. In *Proc. PVLDB Endow.*, pages 2024–2027.

[Dash et al., 2008] Dash, D., Rao, J., Megiddo, N., Ailamaki, A., and Lohman, G. (2008). Dynamic faceted search for discovery-driven analysis. In *ACM conference on Information and knowledge management*, pages 3–12.

[Diao et al., 2015] Diao, Y., Dimitriadou, K., Li, Z., Liu, W., Papaemmanouil, O., Peng, K., and Peng, L. (2015). Aide: an automatic user navigation system for interactive data exploration. In *Proc. PVLDB Endow.*, pages 1964–1967.

[Dietterich et al., 2000] Dietterich, T. G. et al. (2000). Ensemble methods in machine learning. *Multiple classifier systems*, 1857:1–15.

[Dimitriadou et al., 2014] Dimitriadou, K., Papaemmanouil, O., and Diao, Y. (2014). Explore-by-example: an automatic query steering framework for interactive data exploration. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 517–528.

[Dimitriadou et al., 2016] Dimitriadou, K., Papaemmanouil, O., and Diao, Y. (2016). Aide: An active learning-based approach for interactive data exploration. *IEEE Transactions on Knowledge and Data Engineering*, 28:2842–2856.

[Drosou and Pitoura, 2012a] Drosou, M. and Pitoura, E. (2012a). Dynamic diversification of continuous data. In *International Conference on Extending Database Technology (EDBT)*, pages 216–227.

[Drosou and Pitoura, 2012b] Drosou, M. and Pitoura, E. (2012b). Result diversification based on dissimilarity and coverage. In *Proc. PVLDB Endow.*, pages 13–24.

[Drosou and Pitoura, 2015] Drosou, M. and Pitoura, E. (2015). Multiple radii disc diversity: Result diversification based on dissimilarity and coverage. *ACM Transactions on Database Systems (TODS)*, 40(1):4:1–4:43.

[Ehsan et al., 2016] Ehsan, H., Sharaf, M. A., and Chrysanthis, P. K. (2016). Muve: Efficient multi-objective view recommendation for visual data exploration. In *IEEE International Conference on Data Engineering (ICDE)*.

[Ehsan et al., 2017] Ehsan, H., Sharaf, M. A., and Chrysanthis, P. K. (2017). Efficient recommendation of aggregate data visualizations. In *Accepted for publish in IEEE Transactions on Knowledge and Data Engineering*, pages 263–277.

[Eirinaki et al., 2014] Eirinaki, M., Abraham, S., Polyzotis, N., and Shaikh, N. (2014). Querie: Collaborative database exploration. *IEEE Trans. Knowl. Data Eng.*, 26(7):1778–1790.

[El-Yaniv and Wiener, 2012] El-Yaniv, R. and Wiener, Y. (2012). Active learning via perfect selective classification. In *The Journal of Machine Learning Research*, volume 13, pages 255–279.

[Feild and Allan, 2013] Feild, H. and Allan, J. (2013). Task-aware query recommendation. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–92.

[Ferecatu and Geman, 2007] Ferecatu, M. and Geman, D. (2007). Interactive search for image categories by mental matching. In *International Conference on Computer Vision (ICCV)*, pages 1–8.

[Fraternali et al., 2012] Fraternali, P., Martinenghi, D., and Tagliasacchi, M. (2012). Top-k bounded diversification. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 421–432.

[Freytag et al., 2014] Freytag, A., Rodner, E., and Denzler, J. (2014). Selecting influential examples: Active learning with expected model output changes. In *European Conference on Computer Vision*, pages 562–577.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

[Ge and Chrysanthis, 2020] Ge, X. and Chrysanthis, P. (2020). Efficient prefdiv algorithms for effective top-k result diversification. In *International Conference on Extending Database Technology (EDBT)*, pages 335–346.

[Ge and Chrysanthis, 2021] Ge, X. and Chrysanthis, P. K. (2021). On supporting scalable active learning-based interactive data exploration with uncertainty estimation index. In *International Conference on Extending Database Technology (EDBT)*, pages 421–426.

[Ge et al., 2016a] Ge, X., Chrysanthis, P. K., and Pelechrinis, K. (2016a). Mpg: Not so random exploration of a city. In *IEEE International Conference on Mobile Data Management*, pages 72–81.

[Ge et al., 2017a] Ge, X., Daphalapurkar, A., Shimpi, M., Kohli, D., Pelechrinis, K., Chrysanthis, P. K., and Zeinalipour-Yazti, D. (2017a). Data-driven serendipity navigation in urban places. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 2501–2504.

[Ge et al., 2017b] Ge, X., Panati, S. R., Pelechrinis, K., Chrysanthis, P. K., and Sharaf, M. A. (2017b). In search for relevant, diverse and crowd-screen points of interests. In *International Conference on Extending Database Technology (EDBT)*, pages 578–581.

[Ge et al., 2016b] Ge, X., Xue, Y., Luo, Z., Sharaf, M. A., and Chrysanthis, P. K. (2016b). Request: A scalable framework for interactive construction of exploratory queries. In *IEEE International Conference on Big Data*, pages 646–655.

[Ge et al., 2020] Ge, X., Zhang, X., and Chrysanthis, P. (2020). Exnav: An interactive big data exploration framework for big unstructured data. In *IEEE International Conference on Big Data*, pages 503–512.

[Gheorghiu et al., 2015] Gheorghiu, R., Chrysanthis, P. K., and Labrinidis, A. (2015). Unifying qualitative and quantitative database preferences to enhance query personalization. In *International Workshop on Exploratory Search in Databases and the Web (ExploreDB)*, pages 6–8.

[Gheorghiu et al., 2014] Gheorghiu, R., Labrinidis, A., and Chrysanthis, P. K. (2014). A user-friendly framework for database preferences. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborativeCom)*, pages 205–214.

[Gou et al., 2012] Gou, J., Du, L., Zhang, Y., Xiong, T., et al. (2012). A new distance-weighted k-nearest neighbor classifier. *J. Inf. Comput. Sci*, 9(6):1429–1436.

[Griffin and Tversky, 1992] Griffin, D. and Tversky, A. (1992). The weighing of evidence and the determinants of confidence. In *Cognitive Psychology*.

[Grohe, 1999] Grohe, M. (1999). Descriptive and parameterized complexity. In *Computer Science Logic, 13th Workshop, number 1683 in Lecture Notes in Computer Science (LNCS)*, pages 14–31.

[Grover and Leskovec, 2016] Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *ACM SIGKDD Conference on Special Interest Group on Knowledge Discovery and Data Mining*, pages 855–864.

[Haas et al., 2015] Haas, D., Wang, J., Wu, E., and Franklin, M. J. (2015). Clamshell: Speeding up crowds for low-latency data labeling. In *Proc. PVLDB Endow.*, pages 372–383.

[Hanneke, 2011] Hanneke, S. (2011). Rates of convergence in active learning. In *Annals of Statistics*, volume 39, pages 333–361.

[Hanneke, 2016] Hanneke, S. (2016). Refined error bounds for several learning algorithms. In *Journal of Machine Learning Research*, volume 17, pages 1–55.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[Hong et al., 2017] Hong, W., Yuan, J., and Bhattacharjee, S. D. (2017). Fried binary embedding for high-dimensional visual features. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6221–6229.

[Huang et al., 2019] Huang, E., Peng, L., Palma, L. D., Abdelkafi, A., Liu, A., and Diao, Y. (2019). Optimization for active learning-based interactive database exploration. In *Proc. PVLDB Endow.*

[Hussain et al., 2015] Hussain, Z., Khan, H. A., and Sharaf, M. A. (2015). Diversifying with few regrets, but too few to mention. In *International Workshop on Exploratory Search in Databases and the Web (ExploreDB)*, pages 27–32.

[Idreos et al., 2015] Idreos, S., Papaemmanouil, O., and Chaudhuri, S. (2015). Overview of data exploration techniques. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*.

[Islam et al., 2013] Islam, S., Liu, C., and Zhou, R. (2013). A framework for query refinement with user feedback. In *J. Syst. Softw., 86(6):1580–1595*.

[Juslin et al., 1998] Juslin, P., Olsson, H., and Winman, A. (1998). The calibration issue: Theoretical comments on suantak, bolger, and ferrell (1996). *Organizational Behavior and Human Decision Processes*, 73(1):3–26.

[Kalinin et al., 2014] Kalinin, A., Cetintemel, U., and Zdonik, S. (2014). Interactive data exploration using semantic windows. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 505–516.

[Ke et al., 2017] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Neural Information Processing Systems*, pages 3146–3154.

[Keil et al., 2015] Keil, J. M., Mitchell, J. S. B., Pradhan, D., and Vatshelle, M. (2015). An algorithm for the maximum weight independent set problem on outerstring graphs. In *Canadian Conference on Computational Geometry*, pages 19–25.

[Kersten et al., 2011] Kersten, M. L. et al. (2011). The researcher's guide to the data deluge. *PVLDB*, 4(12):1474–1477.

[Khan and Sharaf, 2015] Khan, H. A. and Sharaf, M. A. (2015). Progressive diversification for column-based data exploration platforms. In *IEEE ICDE*.

[Kiessling and Kostler, 2002] Kiessling, W. and Kostler, G. (2002). Preference SQL: design, implementation, experiences. In *VLDB*, pages 990–1001.

[Kovashka and Grauman, 2013] Kovashka, A. and Grauman, K. (2013). Attribute pivots for guiding relevance feedback in image search. In *International Conference on Computer Vision (ICCV)*, pages 297–304. IEEE Computer Society.

[Lewis and Gale, 1994] Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12.

[Li et al., 2015] Li, H., Chan, C.-Y., and Maier, D. (2015). Query from examples: An iterative, data-driven approach to query construction. In *Proc. PVLDB Endow.*, pages 2158–2169.

[Li et al., 2008] Li, H., Wang, Y., Zhang, D., Zhang, M., and Chang, E. Y. (2008). Pfp: parallel fp-growth for query recommendation. In *ACM Conference on Recommender Systems (RecSys)*, pages 107–114.

[Liu and Heer, 2014] Liu, Z. and Heer, J. (2014). The effects of interactive latency on exploratory visual analysis. *IEEE Trans. Vis. Comput. Graph.*, 20(12):2122–2131.

[Lowd and Domingos, 2005] Lowd, D. and Domingos, P. (2005). Naive bayes models for probability estimation. In *International Conference on Machine Learning (ICML)*, pages 529–536.

[Manas Joglekar, 2016] Manas Joglekar, Hector Garcia-Molina, A. P. (2016). Interactive data exploration with smart drill-down. In *IEEE International Conference on Data Engineering (ICDE)*, pages 906–917.

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*, pages 3111–3119.

[Mishra and Koudas, 2009] Mishra, C. and Koudas, N. (2009). Interactive query refinement. In *International Conference on Extending Database Technology (EDBT)*, pages 862–873.

[Modi and Kovashka, 2017] Modi, B. and Kovashka, A. (2017). Confidence and diversity for active selection of feedback in image retrieval. In *British Machine Vision Conference (BMVC)*.

[Mohamad et al., 2018] Mohamad, S., Sayed Mouchaweh, M., and Bouchachia, A. (2018). Active learning for classifying data streams with unknown number of classes. *Neural Networks*, 98:1–15.

[Mottin et al., 2017] Mottin, D., Lissandrini, M., Velegrakis, Y., and Palpanas, T. (2017). New trends on exploratory methods for data analytics. *PVLDB*, 10(12):1977–1980.

[Mozafari et al., 2014] Mozafari, B., Sarkar, P., Franklin, M. J., and Jordan, M. I. (2014). Scaling up crowd-sourcing to very large datasets:a case for active learning. In *Proc. PVLDB Endow.*

[Niranjan Kamat, 2014] Niranjan Kamat, Prasanth Jayachandran, K. T. A. N. (2014). Distributed and interactive cube exploration. In *International Conference on Data Engineering*, pages 472–483.

[Panigrahi et al., 2012] Panigrahi, D., Sarma, A. D., Aggarwal, G., and Tomkins, A. (2012). Online selection of diverse results. In *ACM International Web Search and Data Mining*, pages 263–272.

[Park and Jun, 2009] Park, H.-S. and Jun, C.-H. (2009). A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341.

[Peng et al., 2020] Peng, B., Fatourou, P., and Palpanas, T. (2020). MESSI: in-memory data series indexing. In *IEEE International Conference on Data Engineering (ICDE)*, pages 337–348. IEEE.

[Peng et al., 2017] Peng, L., Huang, E., Xing, Y., Liu, A., and Diao, Y. (2017). Uncertainty sampling and optimization for interactive database exploration. In *UMass Technical Report*.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

[Qarabaqi and Riedewald, 2014] Qarabaqi, B. and Riedewald, M. (2014). User-driven refinement of imprecise queries. In *IEEE International Conference on Data Engineering (ICDE)*, pages 916–927.

[Qarabaqi and Riedewald, 2016] Qarabaqi, B. and Riedewald, M. (2016). Merlin: Exploratory analysis with imprecise queries. *IEEE Trans. Knowl. Data Eng.*, 28(2):342–355.

[Qian et al., 2019] Qian, K., Popa, L., and Sen, P. (2019). Systemer: A human-in-the-loop system for explainable entity resolution. *Proc. Proc. PVLDB Endow. Endow.*, 12(12):1794–1797.

[Qian et al., 2020] Qian, K., Raman, P. C., Li, Y., and Popa, L. (2020). Learning structured representations of entity names using active learning and weak supervision. *CoRR*, abs/2011.00105.

[Qin et al., 2012] Qin, L., Yu, J. X., and Chang, L. (2012). Diversifying top-k results. In *Proc. PVLDB Endow.*, pages 1124–1135.

[Quellec et al., 2017] Quellec, G., Cazuguel, G., Cochener, B., and Lamard, M. (2017). Multiple-instance learning for medical image and video analysis. *IEEE Reviews in Biomedical Engineering*, 10:213–234.

[Raghu et al., 2019] Raghu, V. K., Ge, X., Balajee, A., Shirer, D. J., Das, I., Benos, P. V., and Chrysanthis, P. K. (2019). A pipeline for integrated theory and data-driven modeling of genomic and clinical data. In *ACM International Workshop on Data Mining in Bioinformatics (ACM BioKDD)*.

[Raghu et al., 2017] Raghu, V. K., Ge, X., Chrysanthis, P. K., and Benos, P. V. (2017). Integrated theory-and data-driven feature selection in gene expression data analysis. In *IEEE International Conference on Data Engineering (ICDE)*, pages 1525–1532.

[Ravi et al., 1991] Ravi, S. S., Rosenkrantz, D. J., and Tayi, G. K. (1991). Facility dispersion problems: Heuristics and special cases. In *Algorithms and Data Structures Symposium (WADS)*, pages 355–366.

[Ray and Craven, 2005] Ray, S. and Craven, M. (2005). Supervised versus multiple instance learning: an empirical comparison. In *International Conference on Machine Learning (ICML)*, pages 697–704.

[Ren et al., 2016] Ren, W., Huang, K., Tao, D., and Tan, T. (2016). Weakly supervised large scale object localization with multiple instance learning and bag splitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):405–416.

[Roy et al., 2008] Roy, S. B., Wang, H., Das, G., Nambiar, U., and Mohania, M. (2008). Minimum-effort driven dynamic faceted search in structured databases. In *ACM conference on Information and knowledge management*, pages 13–22.

[Roy et al., 2009] Roy, S. B., Wang, H., Nambiar, U., Das, G., and Mohania, M. (2009). Dynacet: Building dynamic faceted search systems over databases. In *International Conference on Data Engineering*, pages 1463–1466.

[Santos et al., 2015] Santos, R. L. T., Macdonald, C., and Ounis, I. (2015). Search result diversification. In *Foundations and Trends in Inf Retrieval*, volume 9, pages 1–90.

[Sarawagi and Bhamidipaty, 2002] Sarawagi, S. and Bhamidipaty, A. (2002). Interactive deduplication using active learning. In *ACM SIGKDD Conference on Special Interest Group on Knowledge Discovery and Data Mining*, pages 269–278.

[Sellam and Kersten, 2013] Sellam, T. and Kersten, M. L. (2013). Meet charles, big data query advisor. In *CIDR*.

[Settles, 2009] Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison.

[Settles and Craven, 2008] Settles, B. and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1070–1079.

[Settles et al., 2008] Settles, B., Craven, M., and Ray, S. (2008). Multipleinstance active learning. In *Neural Information Processing Systems*.

[Seung et al., 1992] Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *ACM Workshop on Computational Learning Theory*, pages 287–294.

[Sheng et al., 2008] Sheng, V. S., Provost, F., and Ipeirotis, P. G. (2008). Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM.

[Stefanidis et al., 2010] Stefanidis, K., Drosou, M., and Pitoura, E. (2010). Perk: personalized keyword search in relational databases through preferences. In *International Conference on Extending Database Technology (EDBT)*, pages 585–596.

[Stefanidis et al., 2011] Stefanidis, K., Koutrika, G., and Pitoura, E. (2011). A survey on representation, composition and application of preferences in database systems. *ACM Transactions on Database Systems (TODS)*, 36(19):19:1–19:45.

[Tang et al., 2017] Tang, B., Han, S., Yiu, M. L., Ding, R., and Zhang, D. (2017). Extracting top-k insights from multi-dimensional data. In *ACM SIGMOD Conference on Special Interest Group on Management of Data*, pages 1509–1524.

[Tang et al., 2015] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). LINE: large-scale information network embedding. In *International World Wide Web Conferences (WWW)*, pages 1067–1077.

[Thang et al., 2015] Thang, D. C., Tam, N. T., Hung, N. Q. V., and Aberer, K. (2015). An evaluation of diversification techniques. *Lecture Notes in Computer Science (LNCS)*, 9262:215–231.

[Tong et al., 2011] Tong, H., He, J., Wen, Z., Konuru, R., and Lin, C.-Y. (2011). Diversified ranking on large graphs: an optimization viewpoint. In *ACM KDD*, pages 1028–1036.

[Vartak et al., 2016] Vartak, M., Raghavan, V., Rundensteiner, E. A., and Madden, S. (2016). Refinement driven processing of aggregation constrained queries. In *International Conference on Extending Database Technology (EDBT)*, pages 101–112.

[Vartak et al., 2015] Vartak, M., Rahman, S., Madden, S., Parameswaran, A., and Polyzotis, N. (2015). Seedb: Efficient data-driven visualization recommendations to support visual analytics. In *Proc. PVLDB Endow.*, pages 2182–2193.

[Wen et al., 2018] Wen, Y., Zhu, X., Roy, S., and Yang, J. (2018). Interactive summarization and exploration of top aggregate query answers. In *Proc. PVLDB Endow.*, pages 2196–2208.

[Witten and et al., 1999] Witten, I. H. and et al. (1999). Weka: Practical machine learning tools and techniques with java implementations. In *Workshop: Emerging Knowledge Engineering and Connectionist-Based Information Systems*.

[Xue and Hauskrecht, 2016] Xue, Y. and Hauskrecht, M. (2016). Robust learning of classification models from noisy soft-label information. In *European Conference on Computer Vision*.

[Xue and Hauskrecht, 2017] Xue, Y. and Hauskrecht, M. (2017). Efficient learning of classification models from soft-label information by binning and ranking. pages 164–169.

[Yang et al., 2020] Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Ábrego, G. H., Yuan, S., Tar, C., Sung, Y., Strope, B., and Kurzweil, R. (2020). Multilingual universal sentence encoder for semantic retrieval. In *ACL*, pages 87–94.

[Yu et al., 2009] Yu, C., Lakshmanan, L., and Amer-Yahia, S. (2009). It takes variety to make a world: Diversification in recommender systems. In *International Conference on Extending Database Technology (EDBT)*, pages 368–378.

[Zhang et al., 2013] Zhang, D., He, J., and Lawrence, R. D. (2013). MI2LS: multi-instance learning from multiple informationsources. In *ACM SIGKDD Conference on Special Interest Group on Knowledge Discovery and Data Mining*, pages 149–157.

[Zhang et al., 2017] Zhang, Y., Wang, Y., Cai, W., Zhou, S., and Zhang, Y. (2017). From theory to practice: Efficient active cost-sensitive classification with expected error reduction. In *International World Wide Web Conferences*, pages 153–161.

[Zhu et al., 2015] Zhu, J., Wu, J., Xu, Y., Chang, E. I., and Tu, Z. (2015). Unsupervised object class discovery via saliency-guided multiple class learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(4):862–875.

[Zhu et al., 2007] Zhu, X., Zhang, P., Lin, X., and Shi, Y. (2007). Active learning from data streams. In *International Conference on Data Mining (ICDM)*, pages 757–762. IEEE Computer Society.

[Ziegler et al., 2005] Ziegler, C.-N., Sean M. McNee, J. A. K., and Lausen, G. (2005). Improving recommendation lists through topic diversification. In *International World Wide Web Conferences (WWW)*, pages 22–32.

[Zliobaite et al., 2011] Zliobaite, I., Bifet, A., Pfahringer, B., and Holmes, G. (2011). Active learning with evolving streaming data. In *Machine Learning and Knowledge Discovery in Databases - European Conference (ECML)*, volume 6913 of *Lecture Notes in Computer Science*, pages 597–612.