KNOWLEDGE DISCOVERY WITH RECOMMENDERS FOR BIG DATA MANAGEMENT IN SCIENCE AND ENGINEERING COMMUNITIES

A Thesis presented to

the Faculty of the Graduate School

at the University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

YUANXUN ZHANG

Dr. Calyam Prasad, Thesis Supervisor

December, 2020

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

KNOWLEDGE DISCOVERY WITH RECOMMENDERS FOR BIG DATA MANAGEMENT IN SCIENCE AND ENGINEERING COMMUNITIES

presented by Yuanxun Zhang,
a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Prasad Calyam

Dr. Dong Xu

Dr. Satish Nair

Dr. Trupti Joshi

ACKNOWLEDGMENTS

During my over seven-year Ph.D. study at the University of Missouri - Columbia, I have gotten help from a lot of amazing people to achieve today's results.

First and foremost, I would like to thank my advisor Prof. Prasad Calyam for his great guidance and timely support my studies over the past seven years. I am greatly thankful that he would hire me from China, and open a gate for me to explore computer science problem. He teaches me the methods for doing research, and encourages me to discover the fields I am interested. I could not achieve today's achievements without him.

I would also like to thank my committee members, Prof. Dong Xu, Prof. Satish Nair, and Prof. Trupti Joshi, for their invaluable assistance, and feedback at all stages of this thesis.

In addition, a special thanks to my awesome labmates, Ronny Bazan Antequera, Dmitrii Chemodanov, Saptarshi Debroy, Samaikya Valluripally, Longhai Cui, Sai Swathi, Songjie Wang, and Chengyi Qu. Besides, I also would like to thank other collaborators form other laboratories, Yang Liu, Shuai Zeng, and Feng Feng. They are not only good partners but also good friends. I have very good memories of studying and working with you.

Last, but definitely not least, I would also like to thank my family for their love and support without which I would not have survived the Ph.D. study.

TABLE OF CONTENTS

List of Tables

List of Figures

Figure Page

ABSTRACT

Recent science and engineering research tasks are increasingly becoming data-intensive and use workflows to automate integration and analysis of voluminous data to test hypotheses. Particularly, bold scientific advances in areas of neuroscience and bioinformatics necessitate access to multiple data archives, heterogeneous software and computing resources, and multi-site interdisciplinary expertise. Datasets are evolving, and new tools are continuously invented for achieving new state-of-the-art performance. Principled cyber and software automation approaches to data-intensive analytics using systematic integration of cyberinfrastructure (CI) technologies and knowledge discovery driven algorithms will significantly enhance research and interdisciplinary collaborations in science and engineering. In this thesis, we demonstrate a novel recommender approach to discover latent knowledge patterns from both the infrastructure perspective (i.e., measurement recommender) and the applications perspective (i.e., topic recommender and scholar recommender).

In the infrastructure perspective, we identify and diagnose network-wide anomaly events to address performance bottleneck by proposing a novel measurement recommender scheme. In cases where there is a lack of ground truth in networking performance monitoring (e.g., perfSONAR deployments), it is hard to pinpoint the root-cause analysis in a multi-domain context. To solve this problem, we define a "social plane" concept that relies on recommendation schemes to share diagnosis knowledge or work collaboratively. Our solution makes it easier for network operators and application users to quickly and effectively troubleshoot performance bottlenecks on wide-area network backbones. To evaluate our "measurement recommender", we use both real and synthetic datasets. The results show our measurement recommender scheme has high performance in terms of precision, recall, and accuracy, as well as efficiency in terms of the time taken for large volume measurement trace analysis.

In the application perspective, our goal is to shorten time to knowledge discovery and adapt prior domain knowledge for computational and data-intensive communities. To achieve this goal, we design a novel topic recommender that leverages a domain-specific topic model (DSTM) algorithm to help scientists find the relevant tools or datasets for their applications. The DSTM is a probabilistic graphical model that extends the Latent Dirichlet Allocation (LDA) and uses the Markov chain Monte Carlo (MCMC) algorithm to infer latent patterns within a specific domain in an unsupervised manner. We evaluate our scheme based on large collections of the dataset (i.e., publications, tools, datasets) from bioinformatics and neuroscience domains. Our experiments result using the perplexity metric show that our model has better generalization performance within a domain for discovering highly-specific latent topics. Lastly, to enhance the collaborations among scholars to generate new knowledge, it is necessary to identify scholars with their specific research interests or cross-domain expertise. We propose a "ScholarFinder" model to quantify expert knowledge based on publications and funding records using a deep generative model. Our model embeds scholars' knowledge in order to recommend suitable scholars to perform multi-disciplinary tasks. We evaluate our model with state-of-the-art baseline models (e.g., XGBoost, DNN), and experiment results show that our ScholarFinder model outperforms state-of-the-art models in terms of precision, recall, F1-score, and accuracy.

Chapter 1

Introduction

1.1 Science Drivers for Bold Research Problems

Neuroscience drivers. Need for such cyber and software automation in the neuroscience community is evident from the National Research Council report titled - Research at the Intersection of the Physical and Life Sciences that identified 'Understanding the Brain' as one of the top five grand challenges for research that will significantly benefit society [1]. Similarly the National Academy of Engineers has identified 'Reverse Engineer the Brain' as one of the 14 Grand Challenges for Engineering in the 21st century [2]. The importance of these research challenges is also evident from the substantial increase in funding for multi-disciplinary neuroscience research by federal agencies (e.g., BRAIN initiative). At the same time, neuroscience as an area has seen a 592% increase in PSAT major selections among 9-11th graders (2007-2013) and a 100% increase in PhD degrees awarded (2003-13; [3]). This surge in interest has resulted in the initiation of undergraduate majors in neuroscience at 4-year institutions and in universities such as MIT, Harvard, UCLA, and University of Chicago [4, 5]. All this represents a propitious convergence of interests and a tremendous opportunity to boost interdisciplinary training and research interactions among engineers and neuroscientists to tackle

these challenges at the scientific and cyberinfrastructure (CI) levels.

Bioinformatics drivers. Similarly, the need for cyber and software automation in the bioinformatics community has been driven by the technological advances in next-generation sequencing that have completely revolutionized biological research relating to the world's crop breeding and production. Data-intensive workflows are being developed to empower scientists to conduct experiments on a whole genome scale and to generate a snapshot of all changes happening within an organism. Big Data are been generated in biology including transcriptomics (RNA-Seq), single cell RNA-Seq (scRNA-Seq), proteomics, metabolomics, microRNA (miRNA), small interfering RNA (siRNA), microbiome, metagenome, and genomic variations including Genome Wide Association Studies (GWAS) and Single Nucleotide Polymorphisms (SNP) as well as phenotypic data. To meet the data-intensive multi-omics application analytics needs, users in research and education communities accessing organism-specific databases (e.g., SoyKB [6, 7], SoyBase [8], MaizeGDB [9], Arabidopsis Information Portal [10], LegumeIP [11], and Medicago Gene Atlas database [12, 13]) need access to new tools and infrastructure capabilities. All of these data (ranging from sizes of a few GB to several TB) processed through CI-based workflows can be mined in an innovative and integrative manner through composition of 'multi-omics' analysis pipelines. Such workflows thus can provide valuable insights and comprehensive understanding of systems biology in all organisms including plants, animal, humans, microbes and viruses.

1.2 Knowledge Discovery Challenges

1.2.1 Infrastructure Perspective

Research and training in neural science and engineering increasingly deals with diverse (across levels) and voluminous multi-parameter data [14], posing unique challenges outlined in an NSF iNeuro report [7] as limited access to: (i) multi-

omics data archives [8], (ii) heterogeneous software [9] and computing resources (Neuroscience Gateway [10], Amazon Web Services (AWS)), and (iii) to multi-site interdisciplinary expertise (e.g., engineering, biology and psychology). Existing distributed high-performance computing (HPC) and CI resources enable access to analyze and visualize such data. However, to fully utilize their capabilities, neuroscientists (often with limited CI skills) are required to take valuable time away from the focus of knowledge discovery in neuroscience, to learn how to use such technology. There is a consensus that neuroscience research is also hampered by limited exchange of ideas, sharing of data, and collaboration within the community. Labs pursue their individual research independently, distribute their research via journal articles, and reinvent the computing pipelines used in their analysis [11] time and again. Such practices often results in redundant analysis scripts from these independent labs, which are often difficult to reproduce, due to poor coding skills, lack of version control and manual implementation of certain tasks [12]. Moreover, additional data at all levels of neuroscience continues to accumulate at rapid rates and volumes [13]. However, the community lacks effective CI tools and knowledge discovery methods that can guide novice and expert users to harness such data sets and to also foster effective interdisciplinary interactions to advance the 'team science' research needed in neuroscience, in a scalable, reproducible, sustainable, and efficient manner.

1.2.2 Application Perspective

Many plant biology users and database communities (SoyKB, KBCommons, SoyBase, MaizeGDB, Araport, LegumeIP and others) now perform multi-omics genome scale experiments using NGS genome sequencing, epigenomics, miRNA and sRNA, transcriptomics, proteomics, metabolomics and others regularly. Many experiments involve large set of comparisons such as hundreds of genome resequencing datasets or transcriptomics RNA-Seq studies for various conditions that cannot be performed at desktop-scale systems. These datasets are diverse and are generated

from multi-site, multi-campus collaborative projects, with the ultimate goal to share it publicly with the entire community through publications and open-source tools. Most existing tools typically handle analysis with single omics data-types within distributed data repositories. Moreover, users are increasingly looking to leverage shared computing resources available locally or remotely to cope with the data-intensive analytics needs and answer research questions that can only be answered with multi-omics data analysis in an integrated fashion. Particularly, they are drawn to integrate their local computers and storage with public cloud infrastructures such as NSF-supported CyVerse or Amazon Web Services that can provide the 'scale out' analytics capabilities. However, they do not have easy-access to web frameworks and computational infrastructures that support CI templates that help with multi-omics data integration, hypothesis generation and testing with massive data available in the public domain for various biological organisms. Having access to effective CI tools and knowledge discovery methods can foster a user base that can leverage CI capabilities in a 'self-service' manner so that they can continue to add and use valuable information in public databases and related tools for enabling new collaborations and discoveries.

1.3 Recommender Solution Approach

To spur research productivity and interdisciplinary collaborations within neuroscience and bioinformatics (i.e., our target communities), we propose a novel recommender approach to discover latent knowledge pattern. Our recommender approach will feature a multi-layer recommender architecture (i.e., Infrastructure layer and Applications layer) as shown in Figure 1.1 to enable novice/expert user interfaces for knowledge discovery within next-generation science gateways, as well as template-driven control of federated CI resources.

The goal of our recommender approach is to increase the effectiveness of researchers and educators using CI resources in workflow management (currently

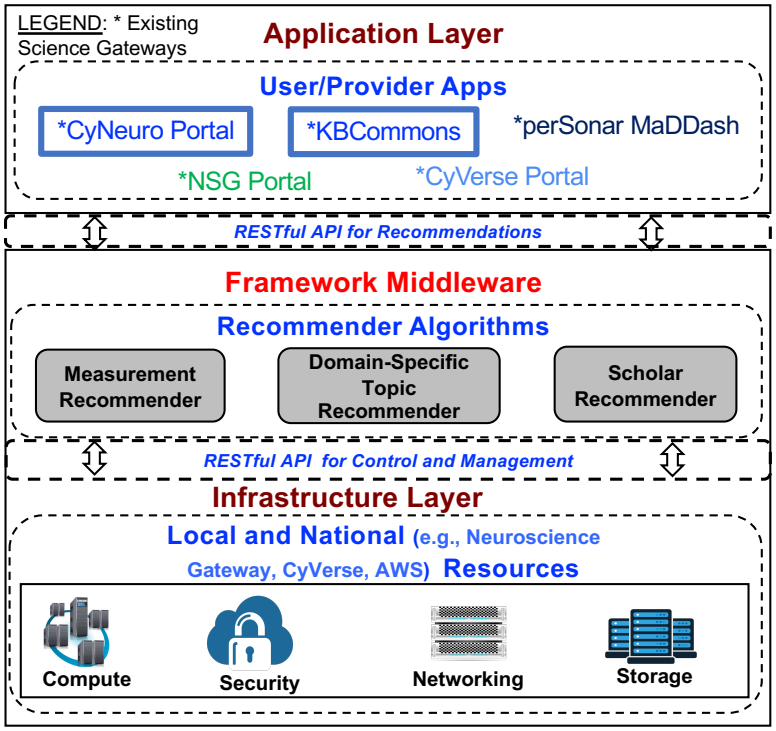


Figure 1.1: A novel recommender approach architecture

a nascent phenomenon) within science gateways, as well as template-driven control of federated CI resources. The recommend approach requirements are motivated by science application drivers within a CyNeuro science gateway for neuroscience, and the KBCommons science gateway for informatics capabilities for genes/proteins, metabolites and other organisms (including plants, animals, humans, microbes and viruses). The CyNeuro science gateway is being developed by the project team at the University of Missouri (MU) for neuroscience as part of an existing NSF CyberTraining project. CyNeuro leverages local institution and national CI resources (e.g., Neuroscience Gateway at UCSD, JetStream, XSEDE, AWS) in order to integrate data, tools for data analytics, computing, and visualization with cyber and software automation. The KBCommons science gateway [15] is being built by the project team based on the success from our Soybean Knowledge Base (SoyKB) efforts [14, 7]. KBCommons portal provides many informatics tools and analytic capabilities, and has already established several building blocks for soybean with its current user base. To meet the analytics needs, it is hosted and closely integrated with NSF-supported CyVerse infrastructure that provides

infrastructure to life science researchers and scientists to access public datasets, manage and store their own data, access high-performance computing, and share results.

Our recommender approach features will supplement these science gateways to leverage local institution and national CI resources (e.g., Neuroscience Gateway, CyVerse) and online databases (e.g., publications, Jupyter Notebooks, open data sets) in order to integrate data, tools for data analytics, computing, and visualization with cyber and software automation components. The framework in Figure 1.1 features application-facing modules to that include: domain-specific topic recommender, scholar recommender, as well as infrastructure-facing modules that include: measurement recommender.

In this thesis, we mainly focus on three major recommender algorithms to discover the latent knowledge pattern from both infrastructure and application perspective, not on the science gateway framework. The major contributions are shown in Table 1.1,

Table 1.1: List of major contributions in this thesis

Perspective	**Recommender**	**Description**
Infrastructure	Measurement Recommender	Troubleshooting network wide anomalies in multi-domain context based on recommendation schemes to share diagnosis knowledge or work collaboratively
Application	Topic Recommender	Proposing a domain-specific topic model (DSTM) algorithm to help scientists find the relevant tools or datasets for building their applications.
	Scholar Recommender	Demonstrating a "ScholarFinder" model to quantify expert knowledge based on publications and funding records using a deep generative model.

1.3.1 Measurement Recommender

Distributed computing applications are increasingly being developed in scientific communities in areas such as biology, geography and high-energy physics. These communities transfer data on a regular basis between computing and collaborator sites at high-speeds on multi-domain networks that span across continents. Given the real-time data consumption demands of users that require ensuring high data throughput through effective network monitoring, there is a rapidly increasing trend to deploy multi-domain, open measurement frameworks such as perfSONAR [16]. The perfSONAR framework has been developed over the span of several years by worldwide-teams and has over 1400 measurement points all over the world. Typical perfSONAR deployments use both passive and active measurements tools such as Ping, Traceroute, OWAMP (for one-way delay measurements) and BWCTL (for TCP/UDP throughput measurements) to create vast data archives of current and historic measurements on national and international backbones (e.g., ESnet, Internet2, GEANT, SWITCH, RNP). The consumers of these measurements could use these archives to understand the network performance changes due to network fault events (e.g., misconfigurations, outages) and cross-traffic congestion that impact end-to-end data intensive application performance across domains. In multi-site Big Data collaborations, the application traffic generated within a network (domain) traverses several different domains or autonomous systems before reaching a destination. As a result, end-to-end performance troubleshooting becomes significantly harder for standalone perfSONAR measurement instances (Measurement Point Appliances or MPAs) installed at the source and destination domains.

Current end-to-end performance monitoring systems for data-intensive applications have many limitations: Firstly, an end-to-end measurement trace normally traverses different domains and different service providers. A single measurement trace is typically insufficient to isolate and diagnose the root-cause of anomaly events. Users usually need enough data to make an accurate and trustworthy

judgment. However, current infrastructure does not provide the necessary tools or services for users to filter out the relevant data from vast archives of measurements. Secondly, multiple measurement traces from different domains may not be calibrated and trustworthy in cases such as invalid data (e.g., negative one-way delay values due to faulty clock synchronization), missing data or too dense/sparse or irregular (i.e., long data collection gaps) measurement data sampling frequency. These measurement data "veracity" issues can result in erroneous features [17], missing events or even exponential anomaly event detection time [18]. Lastly, there are no existing frameworks for users to create social networks and mingle for trustworthy knowledge sharing and collaborative work to efficiently and effectively diagnose anomaly event root-causes.

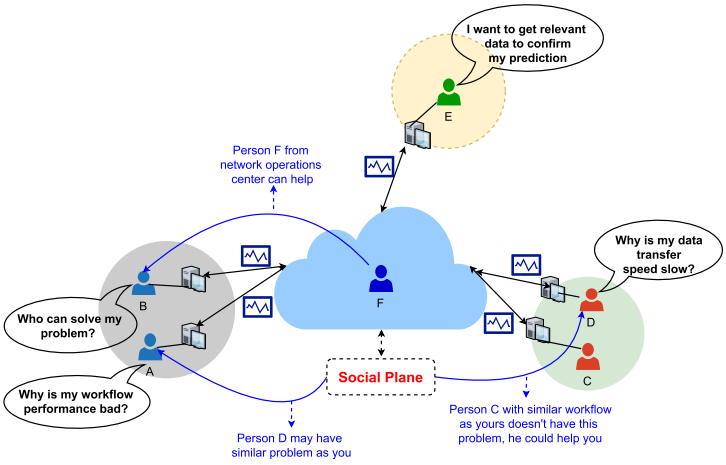


Figure 1.2: Limitations of current end-to-end performance monitoring system due to: challenges in diagnosing the root cause of anomaly events; lack of services to find relevant measurements; lack of frameworks to share knowledge or work collaboratively

Figure 1.2 illustrates the effect of these limitations from an end-user perspective in an example scenario involving users (i.e., A $\sim$ F) from different domains. We assume users A, B and D have to depend on their own knowledge to diagnose their problem of inferior performance due to an anomaly event. Without knowledge sharing, we can see that most of these users fail to diagnose or even recognize their problem, and those users (e.g., user C and F in Figure 1) who may know the cause of anomaly events could not be connected. In addition, many users (e.g., user

E in Figure 1) need more relevant data to confirm their anomaly event detection results. Without pertinent measurements recommendation, users usually collect data randomly, which could result in erroneous detection and ineffective diagnosis.

To overcome such situations that lead to scientists or network operators' dilemma during bottleneck anomaly root-cause diagnosis, we propose a "social plane" approach. The goal of our social plane approach is to provide a framework to allow expert knowledge and other measurement intelligence information to be integrated/- customized into scalable anomaly detection tools. We propose the use of "measurement recommenders" for processing large volumes of measurement traces collected on a daily basis to improve the efficiency of application users and network operators to troubleshoot and work collaboratively on root-cause diagnosis. The novelty of our approach is to combine: (i) content-based filtering which recommends pertinent traces based on user's measurements analysis objective to achieve more accurate detection results, along with (ii) collaborative filtering to recommend the most pertinent traces which may have similar network issues or possibly the same root-cause issues with user's targeted trace.

The content-based filtering is used to rank and recommend the most pertinent traces based on measurements temporal and spatial similarity matching with a target trace/path. The target trace is the one for which the operator or user needs help to perform some specific measurement correlation analysis. In order to strengthen the measurement data "veracity" information, we propose a data sanity checking scheme. The measurement data sanity scores are extended to use a Bayesian Inference-driven scheme for historical domain/community reputation. This scheme acts as a "confidence indicator" to help the operators and users in relevant long-term measurement subscriptions. The data sanity together with domain reputation ultimately provides the essential meta-information on top of content-based filtering recommendation for the operators or users to take informed decisions.

The collaborative filtering scheme is used to seek the measurements which

have similar network issues, or possibly the same root-cause issues with a user's targeted trace. Through relevant measurement analysis, we could find potential users who face similar problems, so that those users could share diagnosis experiences or work collaboratively on the issue. Our algorithm does so by measuring the similarity in anomaly symptoms between users' targeted measurements and candidate measurements. Lastly, we implement a "social network" for measurements, on top of filtering scheme to demonstrate how our collaborative filtering scheme helps users to enhance knowledge sharing and trustworthy collaboration for network performance expectation management.

1.3.2 Topic Recommender

Scientific domains such as bioinformatics and neuroscience have benefited from Big Data analytics, High-performance computing (HPC), and High-throughput computing (HTC) that uses underlying machine learning techniques for solving computational and data-intensive research problems. Bold innovations will increasingly emerge from processing a large number of datasets or recognizing complex knowledge patterns using e.g., text mining. Moreover, the bold innovations will occur from solving multi-disciplinary research problems that require prior knowledge discovery within disciplines and from cross-domain scientist collaborations. To enable the rapid pace of innovation, scientists are continuously seeking to investigate new methods, develop new tools or integrate structured or unstructured data sets.

Finding relevant knowledge patterns featuring tools, methods, and datasets amongst vast information archives to obtain timely guidance to solve multidisciplinary research problems can be challenging for domain scientists. As shown in Figure 1.3, it involves both intra-domain and cross-domain knowledge discovery. An *intra-domain* scenario commonly occurs when scientists are looking to adapt a state-of-the-art solution in their domain. For example, biologists wanting to know if there is a new tool developed for improving the performance of sequence

alignment. Alternately, a *cross-domain* scenario can be seen when scientists are investigating new solutions by extending relevant methods from other domains. A few example scenarios are as follows: biologists applying relevant machine learning and statistical methods for protein structure predictions; machine learning studies may need to extend new algorithms/tools to solve unique problems in personalized medicine; data-intensive neuroscience efforts could adopt cyberinfrastructure integration best practices from bioinformatics [19] for building workflows across distributed computing resources.

On the other hand, Knowledge creation needs access to experts. However, finding an expert in a scientific domain to execute certain research tasks is demanding and critical for knowledge creation in that domain. Moreover, today's research involves working on bold problems that require interdisciplinary experts and cross-domain collaborations. For example, how can we find pertinent researchers to work on building computational bioinformatics/neuroscience infrastructure for flexibly scaling data analysis pipelines? Thus, it is a hard challenge to find the relevant experts from multiple domains, who would be able to handle diverse and interdisciplinary scientific research tasks in a pool of thousands of scholars.

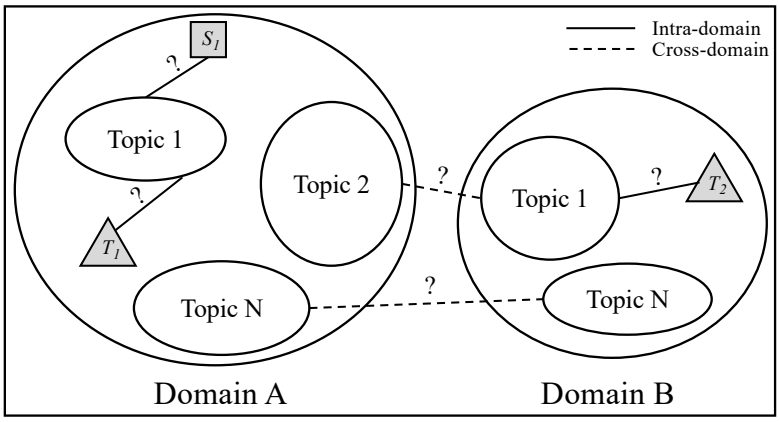


Figure 1.3: Discovery of relationships between research topics, tools (with "T" notation) and datasets (with "S" notation) for scientific domains with intra-domain knowledge and cross-domain knowledge discovery.

For topic recommender, we propose a novel "domain-specific topic model"

(DSTM) to enable the discovery of latent knowledge patterns in scientific domains that rely on prior knowledge discovery and cross-domain collaborations. DSTM is fundamentally a generative model to discover the relationships among research topics, tools, and datasets within intra-domain and cross-domain cases. Our DSTM extends the popular Latent Dirichlet Allocation (LDA) model [20] and the Author-Topic model [21], which is an LDA variant. Our DSTM assumes each topic is represented as a distribution over words, and each tool or dataset is modeled as an individual distribution over topics. Such distributions or parameters can be learned through unsupervised learning from collections of text corpus that reflect the patterns of tools or datasets that are more likely to be used for domain research problems by using the Markov chain Monte Carlo inference algorithm for a specific domain. In order to tangibly apply our DSTM for research activities in scientific communities, we propose three algorithms to apply our DSTM under different knowledge pattern perspectives: (a) *intra-domain knowledge representation* that guides scientists to find existing solutions from their respective domains; (b) *cross-domain knowledge representation* that guides investigation of new solutions by referring to prior solutions in other synergistic domains; (c) *trend representation* that tracks the change in trends or reveals emerging trends over time, in order to guide scientists to make intelligent decisions while choosing tools or datasets to solve a research problem at hand.

1.3.3 Scholar Recommender

For scholar recommender, we propose a novel model viz., "ScholarFinder" to find suitable scholars who can successfully accomplish a given bold/multi-disciplinary research task involving multiple scientific domains. Common existing scholar searching or recommendation approaches involve querying scholars' information (i.e., title, affiliation, or research interests) from their online profiles such as the one shown in Figure 1.4. Using this information, keywords are matched in proposed tasks for judging whether a scholar is a good fit for a given task or not. Such

Figure 1.4: Example of scholar profile that includes tittle, affiliation, research interests, and publications.

solutions might not achieve good performance due to the fact that the research interests tags are commonly fixed, which does not make it possible to capture the changing or expanding research interests of a scholar. Additionally, the tags are always too broad to match specific research tasks. For example, the scholar shown in Figure 1.4 can only match those tasks with keywords such as "computer science", "distributed computing", or "data science". Moreover, when we explore this scholar's publications, we can find that the scholar is also working on other research topics such as "big data", "machine learning". Similarly, if we search for scholars based on these keywords, there will be thousands of candidate scholars to differentiate against.

Based on the above observations, our key problem corresponds to knowledge representation that involves finding a suitable method to quantify scholars' knowledge using contextual information (e.g., scholars' publications). Overcoming this problem can help us solve the issues of using fixed research tags. Given that the publications are text datasets, intuitively, we use a bag-of-words to represent them with a vector. However, a bag-of-words representation will cause the issue of sparsity and high dimensionality. Traditional methods, such as principal component analysis (PCA) can help to reduce dimensions, however these methods need re-computing when new data is added, and the resulting performance is not satis-

factory. Matrix Factorization [22] builds a "user/item" model and performs matrix decomposition to obtain latent users and item features. However, this method also requires re-computing the latent features when new data is added.

Neural language [23] and Skip-gram [24] models are earlier proposed methods for learning good embeddings in language models. These embedding methods are widely used in those datasets with the characteristics of sequence or association. For tabular datasets (e.g., rating matrix), neural network-based methods [25] are popular to learn embeddings. Basically, they build a neural network by randomly initializing latent input vectors (e.g., user/item latent vectors), and performing dot product of latent user/item vectors to make predictions. Using the gradient descent optimization scheme for minimizing the predicting loss, they can learn the good user/item embeddings. Such approaches can solve the issue of re-computing when a new dataset is added but these embeddings can only be used in the same users/items predictions.

Our proposed method involves three learning procedure stages: (a) learn knowledge embedding for scholars based on the contextual information. More specifically, we train the embedding model using the unsupervised learning algorithm Variational Autoencoder (VAE) based on the scholars' publications to capture a reasonable semantic meaning of the context; (b) in a similar manner, we train a different embedding model for representing proposed tasks; (c) we build another deep learning model using knowledge embedding and task embedding as inputs for making predictions and for checking whether a scholar is suitable for any given set of proposed research tasks. We separately train the embedding model (in an unsupervised manner) and the prediction model. Consequently, our pre-trained knowledge embedding can be used for further tasks such as classifications, predictions and visualizations.

1.4 Thesis Outline

In Chapter 2, we firstly introduce our measurement recommender framework. Then, we explain each component of our framework in details. Lastly, we evaluate our framework using real and synthetic datasets; and provide the case studies to demonstrate the possible usage of our measurement recommender.

In Chapter 3, we describe our topic recommender by proposing a domain-specific topic model (DSTM) and related inference algorithm to estimate the latent parameters. We also introduce three applications for discovering the latent patterns from intro-domain, cross-domain and trend perspectives. In Chapter 4, we introduce our scholar recommender by proposing a deep generative model "ScholarFinder".

Finally, we summary this thesis in Chapter 5.

Chapter 2

Measurement Recommender for Performance Management

Multi-domain end-to-end network performance monitoring (NPM) federations such as perfSONAR are increasingly being used in Big Data application management. They rely on trustworthy collaborative measurement intelligence to identify and diagnose network anomaly events that impact application performance. Large volumes of end-to-end measurement traces are generated on a daily basis, and new Big Data analysis techniques are needed to isolate network-wide anomaly event(s) and to diagnose the root-cause(s). In addition, not all network operators and application users have enough knowledge and experience to understand the anomaly events. The lack of a platform for sharing knowledge and working collaboratively makes it difficult to isolate and diagnose network-wide anomaly events quickly and accurately. In this paper, we define a "social plane" that relies on recommended measurements based on "content-based filtering" and "collaborative filtering" approaches to enable network performance expectation management. Based on similarity analysis, the "content-based filtering" facilitates users to subscribe to useful measurements, and the "collaborative filtering" promotes users to share knowledge on anomaly symptoms. Using real perfSONAR measurements and synthetic events, we show the effectiveness of our social plane approach within a SoyKB

Big Data application case study using social network creation and mingling of experts. Our experimental results show that our measurements recommendation scheme has high precision, recall and accuracy, as well as efficiency in terms of the time taken for large volume measurement trace analysis.

2.1 Background and Related Work

2.1.1 Anomaly Detection

To assist network operators in troubleshooting bottlenecks (e.g., prolonged congestion events or device mis-configurations) in high-speed networks, a number of smart and effective network monitoring tools based on statistical measurement data analysis techniques have been developed. In particular, there have been studies on correlated anomaly detection such as [26, 27]. Authors in [26] use PCA technique on passive measurements for network anomaly detection on a network link basis. Authors in [27], the authors address limitations of PCA's failure in detecting strong correlations in distributed network traffic anomalies. Both [26] and [27] do not use topology information and thus propose black-box techniques, in comparison to other topology-aware works such as [28, 29, 30, 31, 32, 33].

In our recent work [28], we used spatial and temporal analysis after combining topology and uncorrelated anomaly events information corresponding to multiple measurement time series for location diagnosis of correlated anomaly events. However, such analysis has a strict requirement for topology information, which is generally not made publicly available by domains that share perfSONAR measurement data. The authors in [29] use Kalman filter for anomaly detection and build a traffic matrix of an enterprise network to overcome link basis limitations. In [34], the authors present a general framework called NICE (Network-wide Information Correlation and Exploration) for analyzing data through correlations and present a qualitative as well as quantitative analysis approach with network related data

such as router logs and topology information. Routing connection relationships are used in [31] for network-wide anomaly detection in backbone networks; relationships are established based on features such as packet sizes, IP addresses and ports. Authors in [35] use perfSONAR measurements for root-cause analysis and localizations of performance problems,

2.1.2 Measurement and Data Reputation

New challenges are emerging for analyzing measurements in large-scale multi-domain infrastructures in order to resolve bottleneck anomaly events and facilitate efficient root-cause diagnosis. In [6], the authors from Google show the benefits of expertise sharing and knowledge management of network failures as vignettes within a complex and high-scale environment. The authors in [35] show that it is possible to have experts identify symptoms of anomaly events based on measurement trace characteristics, which can be shared as measurement intelligence to understand performance bottleneck events in perfSONAR deployments. Our work uses these symptoms knowledge and the idea of expertise sharing within the social plane implementations of Big Data applications such as SoyKB [6].

In addition, our work builds upon [36], where the authors present an automatic network fault detection and diagnosis system for end-users called DYSWIS ("Do You See What I See"). A distributed hash tree network is used to search the collaborative nodes with appropriate properties required to diagnose a failure in large-scale home networks. Contributions of expert knowledge (diagnosis rules and probes) by application developers, vendors, and network administrators are used to enable crowdsourcing of diagnosis strategy. Our work is also comparable to the work in [37], where the authors implement a "social angle" to monitor and detect problems in a large collaborative network environment involving multiple domains. Their solution was limited to a social media platform setup to allow communication of users to share issues, ideas, concerns and problems with other users or network experts and does not feature any recommender algorithms. Our

social plane implementation for measurement intelligence sharing is inspired by the work in [38]. Therein, the authors propose a social-media approach to monitor virtualized environments by creating a "community" that includes various entities along with an administrator.

The risks of using potentially misleading data and the related guidelines to trustworthy measurement best practices were first highlighted in [17], wherein the author explains the methods implemented in handling errors and inaccuracies; the importance of associating meta-data with measurements; the technique of calibrating measurements by examining outliers and testing for consistencies; difficulties that arise with large-scale measurements; among other issues. Our previous work on using sanitized measurement data for anomaly detection [39] is closest to the work by [33] where an anomaly detection system is developed based on prediction of upper and lower dynamic thresholds of various time-varying data trends. In [15], the authors proposed an overlay fault diagnosis framework with a diagnosis uncertainty reasoning analysis based on evidences.

Similarly, reputation-based trust schemes have long been used by the scientific community for decision making in shared environments. In [40], the authors present a reputation-based trust model for peer-to-peer eCommerce communities. Whereas, in [41], the authors describe a similar scheme to use Bayesian Inference to build reputations for agents in the e-business community. Further, works such as [42] extend such reputation models by introducing an age factor in Bayesian Inference as it is desirable to give greater weight to more recent ratings. In [43], the authors propose a tagging and trust mechanism in social networks based on users and their contents that is similar to our research of building a reputation scheme for multi-domain measurement domains within scientific Big Data communities.

2.1.3 Recommendation System

The recommendation system is considered to be one of the most successful approaches for personalized information filtering and searching schemes. Recommen-

dation engines are an interesting alternative to search fields, as recommendation engines help users discover products or content that they may not come across otherwise. They are widely applied in E-commerce ecosystems, such as Amazon and eBay that recommend products to individual customers based on their interests or preferences.

Filter-based (mostly content-based and collaborative) social/community environments have been proposed in different areas of computing. However, in relation to measurement frameworks, such works are limited. Our work is the first to propose a recommender framework to systematically analyze large number of measurement traces to identify bottlenecks and resolve them in multi-domain network monitoring. Collaborative filtering is a technique to filter large sets of data for information and patterns. The authors in [44] propose a novel method that uses social networks and collaborative filtering to identify and prioritize requirements in large-scale software projects. More specifically, they address information overload problems in requirements elicitation of software development activities by using collaborative filtering to recommend relevant requirements to stakeholders and prioritizing the requirements and stakeholders. In [45], the authors use collaborative filtering and a probabilistic topic model to recommend relevant scientific articles to users of online communities. With collaborative filtering, their model can recommend articles for a particular user based on other users who liked similar articles.

Our approach builds upon the above prior works and aims to combine both content-based and collaborative filtering techniques for finding: (a) measurement traces with similar anomaly symptoms, and (b) relevant people who can mingle and resolve any identified bottlenecks. In our previous work [46], we have proposed a content-based recommender to filter relevant measurements based on user's analysis objective, such as temporal analysis and spatial analysis. This work extends the prior content-based filtering in combination with collaborative filtering to find measurements with similar anomaly symptoms such as those identified

in [32]. Furthermore, we show the effectiveness of our social plane approach for a Big Data application case study i.e., SoyKB that involves distributed computing across HPC sites, and Big Data processing within large knowledge bases in bioinformatics.

2.2 Measurement Recommender Framework

The anomaly symptom space can be broad in real network situations, and could pose challenges for effective root-cause analysis. However, it is possible to obtain expert knowledge of common network symptoms that affect e.g., large file transfer applications over wide-area network paths as in [11]. The goal of our social plane approach is to provide a framework to allow such expert knowledge and other measurement intelligence information to be integrated/customized into scalable anomaly detection tools. We propose a subscription of large-scale monitoring of measurement data sets collected on a daily basis and timely identification/notification of critical anomaly events accurately. We suppose that the root-cause diagnosis of the critical anomaly events require mingling of relevant stakeholders for diagnosis. Our social plane approach thus provides a "systematized" method is to: (a) help users to obtain relevant measurements for analysis which may enhance measurements subscription and sharing, and (b) find potential people who may solve a given multi-domain problem which could enhance knowledge sharing and trustworthy collaborations.

As depicted in Figure 2.1, our measurement recommender combines the content-based and collaborative filtering techniques to accomplish the purpose:

Content-based Filtering. Content-based filtering is applied when users do some correlation analysis and want to find pertinent measurements according to their temporal or spatial correlation analysis objectives. We propose the *measurements similarity analysis* algorithm to calculate measurement temporal and spatial attributes similarity score between users' target measurement trace and candidate

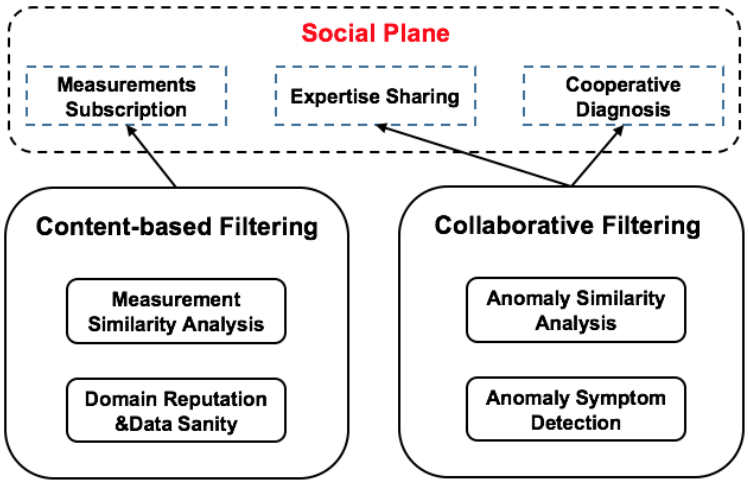


Figure 2.1: Framework of Measurement Recommender

measurement traces from the pool of measurements archives. Using a decision tree, we can rank relevant measurements based on user's temporal or spatial analysis objectives. Because "data veracity" may affect accuracy of analysis results, we provide *data sanity checking* for users to indicate the trustworthiness of measurements and analysis results. With *domain reputation estimation*, users could decide if they should subscribe their measurements from long-term perspective. Hence, content-based filtering scheme could encourage measurements sharing and subscription bases on user's interests.

Collaborative Filtering. Collaborative filtering is applied when users need helps from others who may solve their problem or have similar problem which could involve collaborative work. In order to find similar problem, we classify it with different types of anomaly symptoms (e.g., "delay level shift", "high delay utilization") by using an *anomaly symptom detection* algorithm, and then rank relevant measurement traces with most similar anomaly symptom with target trace using *anomaly similarity analysis*. Through filtering out the measurement traces with similar anomaly events, we connect most relevant persons who face a similar problem. This in turn allows those persons to share diagnosis knowledge and work collaboratively. Consequently, a collaborative filtering scheme encourages diagnosis knowledge sharing and trustworthy collaboration.

With the help of "content-base filtering" and "collaborative filtering" schemes, the "social plane" could be formed to encourage measurements sharing/subscription, expertise sharing and cooperative diagnosis. In the following sections, we explain the details of each of these schemes.

2.3 Content-based Filtering Measurement Recommendation

Content-based filtering is applied when users perform correlation analysis and want to find pertinent measurements according to their temporal or spatial correlation analysis objectives. In this section, we propose our content-based filtering measurements recommendation scheme. Our filtering approach is derived from the concepts of content-based filtering techniques used in many recommendation systems, especially by online retail enterprises, such as Amazon, eBay. Content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the contents/features of the items and users' profiles or interests.

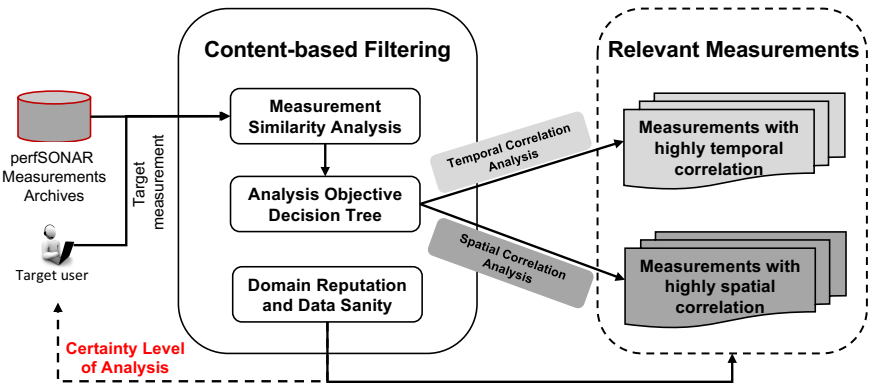


Figure 2.2: Major components of *content-based filtering* scheme, which filters out the relevant measurements according to users' analysis objective

As shown in Figure 2.2, our proposed content-based filtering recommendation scheme filters and ranks most relevant measurement traces from a pool of traces based on *Measurement similarity analysis* and *Analysis objective decision*

tree. In addition, due to current qualities of measurements, “data veracity" is an important factor to affect the accuracy of measurements analysis, especially in correlation analysis. Thus, we provide *Data sanity checking and Domain reputation estimation* in content-based filtering scheme for users to have a level of certainty regarding their measurements and analysis results.

2.3.1 Measurements Similarity Analysis

As proposed in our previous work [46], we argue that for any broad type of correlation analysis objective, i.e., spatial or temporal, there are four important factors that define time-series measurement traces' attributes. They are: 1) *Topology*: the path taken by the measurement probe packets; 2) *Metric*: the measurement tool (such as one-way delay, throughput) used to collect the samples; 3) *Time Range*: the time range of the time-series measurements; 4) *Alignment*: the relative positions of the measurement sampling time stamp instances.

For each of these factors, we will quantify the relative similarity between target trace and candidate traces and then create an overall similarity score from individual factor similarities. The overall similarity score will help the network operators to rank the candidate traces in the order of their relevance.

Measurements Topology Similarity. The topology attribute of any measurement trace is the traceroute information between two sites, which is made of intermediate nodes or hops.

This topology information is very important for correlating measurements traces because - higher the similarity between traces' topologies, higher the probability of common network events of interest. Therefore, we express the topology similarity $topo\_simi_{i,j}$ between target trace i and candidate trace j as:

$$topo\_simi_{i,j} = \frac{topo_i \cap topo_j}{topo_i \cup topo_j} \tag{2.1}$$

Measurements Metric Similarity. The measurement *metric* indicates the net-

work performance measurement tool used for monitoring, such as Ping, OWAMP, BWCTL, etc. The measurement metrics similarity between traces is of importance based on the type of measurement analysis sought. We consider measurement metric similarity for recommendation of measurement traces of two different tools that use different configurations/methods to provide similar metrics e.g., latency or loss. We express measurement metric similarity $metric\_simi_{i,j}$ between traces i and j as a boolean representation:

$$metric\_simi_{i,j} = \begin{cases} 1, (m_i = m_j) \\ 0, (m_i != m_j) \end{cases} \tag{2.2}$$

Measurements Time Range Similarity. The measurement *time range* of a trace is one of the most important measurement attributes for temporal analysis, when the duration of the traces becomes critical to detect a time-specific network event. Thus, if two traces' duration are not aligned temporally, their time range similarity should be equal to zero. Therefore, we express the time range similarity $range\_simi_{i,j}$ between traces i and j as:

$$range\_simi_{i,j} = \frac{r_i \cap r_j}{r_i} \tag{2.3}$$

Measurements Alignment Similarity. Measurement alignment, i.e., the relative positions of measurement sample instances is also significant for correlation analysis with multiple dimension time-series measurements. Samples that are closely aligned are easier to correlate and have better chances of accurate detection of network events upon analysis. Relative alignment of sample instances between two traces is a better metric to quantify their relative similarity. In an illustrative example shown in Figure 2.3, we show one target and three candidate traces with different periods and sampling patterns. As far as the similarity is concerned, candidate trace 1 is best aligned to the target trace as the mean relative displacement between the trace 1 and the target trace is minimum.

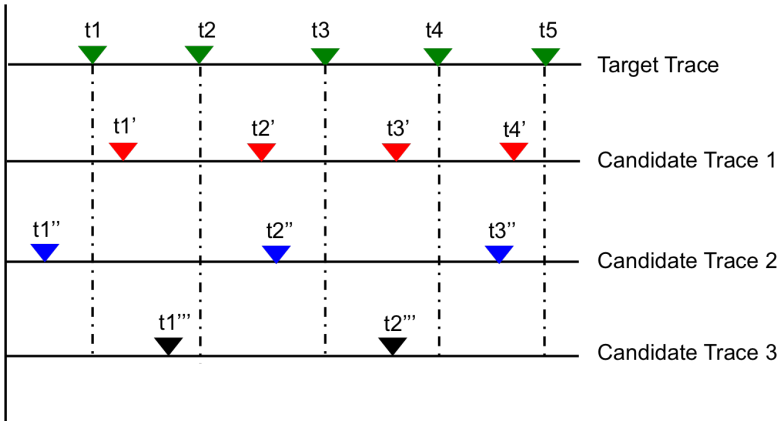


Figure 2.3: Measurements alignment illustration between traces with varied periodicity and non-aligned sample time stamp instances.

Thus, for generic quantification of alignment between measurement traces, we define an alignment displacement metric d that denotes the mean relative distance between sample instances of target trace i and candidate trace j. The metric d is expressed as:

$$d_{i,j} = \sum_{T} |ts_i - \hat{ts_j}| \tag{2.4}$$

where ts_i denotes target trace time stamp, $\hat{ts_j}$ denotes the candidate time stamp closest to the target trace time stamp ts_i, and T denotes the number of such time stamps in the target trace. The range of metric d varies between $[0, +\infty)$ with smaller value indicating better alignment between traces. In order to normalize d and to be consistent with other similarity factors, we transform the range of metric d between $[0, 1]$ using min-max normalization method, which can be expressed as:

$$align\_simi_{i,j} = \frac{max_{d_{i,j}} - d_{i,j}}{max_{d_{i,j}} - min_{d_{i,j}}} \tag{2.5}$$

where $max_{d_{i,j}}$ and $min_{d_{i,j}}$ are the maximum and minimum values of $d_{i,j}$.

2.3.2 Overall Similarity Scores based on Analysis Objective Decision Tree

The overall measurements similarity is expressed as a weighted product of the aforementioned four similarity factors:

$$overall\_simi_{i,j} = \sum_{k} w_k * factor\_simi^k_{i,j} \tag{2.6}$$

where $factor\_simi^k_{i,j}$ denotes each of the aforementioned similarity factors and w_k denotes their respective weights for the overall measurements similarity score. The values of the weights depend on the relative importance of these attributes to achieve different *measurement analysis objectives*. In order to simplify the weights but at the same time to have a more comprehensive list of analysis objective scenarios, we design a decision tree of different generic measurement analysis objectives and corresponding relative importance of attributes' weights (w_t, w_m, w_r, and w_a respectively for each of the similarity factors), as shown in Figure 2.4.

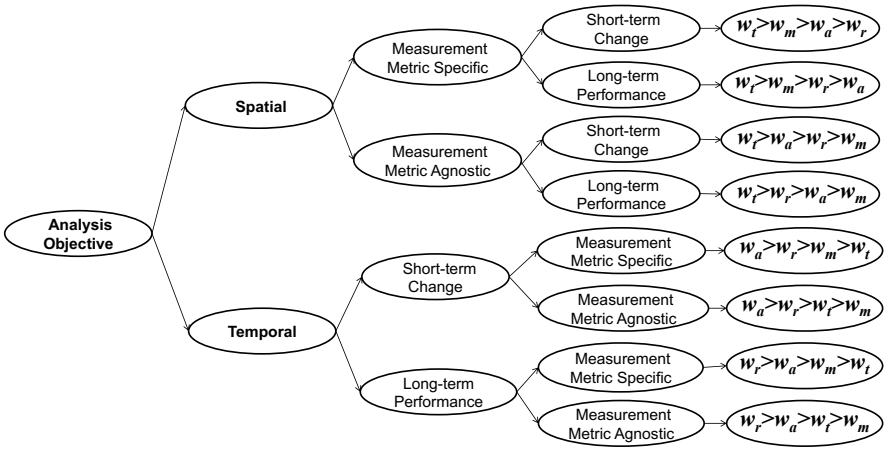


Figure 2.4: Decision tree for different measurement analysis objectives and the corresponding relative measurement attributes' weights

Broadly, we divide the entire analysis objective space into temporal and spatial analysis. In temporal analysis, the weights for temporal factors, such as *time range* and *alignment* are more important than spatial factors such as *topology*. For example, one of the most common measurement analysis for end-to-end data-intensive application management is the correlated anomaly event detection [47], which in-

volves a spatial analysis and falls under 'Analysis Objective'→'Spatial'→'Measurement Metric Specific'→'Short-term Change'. So, the relative weights should be $w_t > w_m > w_a > w_r$. Hence, using a content-based filtering scheme, users can easily obtain the relevant measurements according to their analysis interests.

2.3.3 Data Sanity Checking and Domain Reputation Estimation

"Data veracity" factor is important to determine the accuracy of any measurement analysis results. Hence, we design a "data sanity checking" scheme to check the quality of data, and use it with a "domain reputation" algorithm to verify whether a measurements from a certain domain should be subscribed from a long-term perspective or ignored for critical analysis.

Data Sanity Checking. We implement the two-pronged approach from [39] to sanitize measurement data: a reputation analysis scheme for collected samples, and a filter framework to intelligently prune the potentially misleading samples.

In the context of detecting and diagnosing potential correlated anomaly events within time-series measurements, it is important that the sample data has a desired nature expected by the monitoring objective in terms of two aspects: (i) Sampling Pattern, and (ii) Data Validity. To identify potentially misleading features of measurements, we use the reputation-based data sanity checking scheme which analyzes the measurement samples for sampling pattern, and collected sample validity.

Domain Reputation Estimation. The concept of reputation of data is closely linked to its trustworthiness; an entity's reputation is generally a subjective proof of its historical actions and in most cases, a measure of expectations of future behavior. The main objective of proposing a domain reputation scheme is to generate domain-centric expectations for network operators when they subscribe to measurement data from different domains. Such a domain reputation scheme can encourage trustworthy measurement practices (e.g., sharing of calibrated measure-

ment tool data) among multiple domains supporting Big Data applications.

Reputation of any domain is a function of the quality of measurement data generated from that domain. We observed that in any random collection that was publicly accessible, some measurements exhibit non-periodic sampling patterns, i.e., they are either too dense or too sparse, and some were invalid due to faulty clock synchronization between measurement servers or data corruption (e.g., negative one-way delay values). For spatial analysis with OWAMP, the characteristics are: *sampling pattern* and *data validity*. Thus, we [39] have defined the sanity score of any trace with path (source, destination) i as:

$$s_i = \frac{N_i - (N_i - n_i^{majority}) - (N_i - n_i^{valid})}{N_i} \tag{2.7}$$

where N_i denotes the number of measurement samples in path i, n_i^{valid} denotes the number of valid data samples in path i, and $n_i^{majority}$ denotes the number of samples showing consistent sampling pattern.

Through Bayesian Inference, new or an updated reputation score (i.e., posteriori) of an entity can be computed by combining the old/previous reputation score (i.e., priori) with a new belief. In order to translate sanity scores into domain reputation, we first discretize the measurement data sanity scores into data sanity ratings of a particular domain using boolean variables such as 'Good' (variable x) and 'Bad' (variable y), and some sanity threshold ϵ. The value of ϵ is a measure of the degree of conservativeness of the reputation scheme that is kept constant for the entire system. The value of ϵ can be set based on the distribution of measurements' sanity scores in the system. If the average sanity score of measurement data in the system is very high, ϵ value is kept high to differentiate between good and bad measurements, and vice versa. Usually for all practical purposes, ϵ value is around $[\mu + \sigma, \mu + 2 * \sigma]$.

$$x = |i| \ \ \forall \ s_i >= \epsilon; \ \ y = |i| \ \ \forall \ s_i < \epsilon \tag{2.8}$$

Therefore, at any given time t, the measurement data sanity rating of any domain is represented as $\rho^t = [x, y]^t$. Now if there are T such discrete data sanity ratings collected over a period of time, then the overall data sanity rating after T collection is given as $\rho^T = [x, y]^T$, where x^T and y^T are expressed as:

$$x^T = \sum_{t=1}^{T} \lambda^{T-t} x^t \quad \text{and} \quad y^T = \sum_{t=1}^{T} \lambda^{T-t} y^t \tag{2.9}$$

where $0 \leq \lambda \leq 1$ is called the 'forgetting factor' and keeps the recent history of data sanity rating more relevant in reputation calculation than ancient history. The value of λ represents how forgetful a system is, $\lambda = 1$ means the system forgets nothing. Thus, this value depends on the user's opinion such that, if the user thinks the historical reputation is also very important, this value should be close to 1; however, if the user thinks current reputations are more important, the value should be close to 0.

Now after collecting T such discrete data sanity ratings, the reputation of the domain responsible is expressed as a posterior expectation of beta distribution of ρ^T and is represented as:

$$\begin{aligned} R^T &= \mathrm{E}[beta(\rho^T)] \\ &= \frac{x^T + 1}{x^T + y^T + 2} \end{aligned} \tag{2.10}$$

2.4 Collaborative Filtering Measurement Recommendation

Collaborative filtering is applied when users need help from other users to solve a multi-domain performance issue. When network anomaly events are detected in their monitoring system, the application users or network administrators want to diagnose the reasons of anomaly events. However, anomaly event diagnosis is a challenging problem, especially in and end-to-end multi-domain monitoring

system. Due to lack of skills or knowledge in network diagnosis and complicated reasons to cause network issues involving large measurement archives, most users fail to identify many anomaly events, which may affect their performance. If there was a "social" platform for users to share experience and diagnosis skills, it could improve the efficiency of trouble-shooting anomaly events greatly. This is the reason why we leverage the concept of collaborative filtering to add a "social plane" to anomaly event detection in the overall monitoring system.

Collaborative filtering is widely used in recommendation systems, such as eBay, Amazon, to find the possible interesting products for particular users based on finding other users with similar interests, by using all the users' ratings in their database (also called as user-based collaborative filtering). Similarly, when anomaly events are found in a monitoring system, the users want to find those people who faced similar network anomaly events and find the root-cause of the anomaly events. Thus, we leverage the concept of collaborative filtering in our anomaly detection for finding those network measurement traces with similar anomaly symptoms. *Our approach is also based on the reasonable assumption that the people who have encountered similar networking issues previously have a higher probability to know the root-cause of these anomaly events.*

The original collaborative filtering (user-based model) usually involves two procedures: Finding the Top-N users with similar tastes; Using Top-N users' ratings for a particular item to predict a targeted user's rating. The collaborative filtering in our scenario similarly, has two steps:

- Finding those measurement traces having similar anomaly symptoms;
- Using those measurement traces to connect the potential people who may understand the network issues and provide solutions and suggestions, assuming each measurement trace is managed by the network administrators or Big Data application community users.

As shown in Figure 2.5, target users put their suspicious measurements traces into our collaborative filtering, and then our system pulls all measurements traces

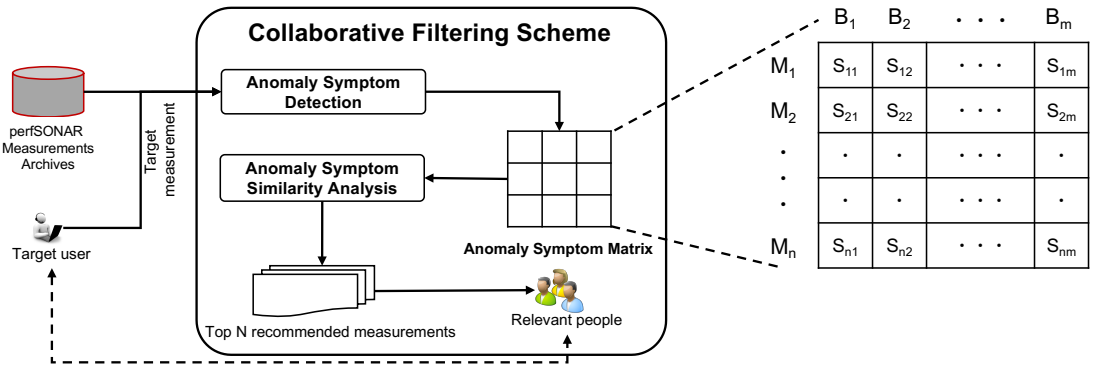


Figure 2.5: Major components of *collaborative filtering* scheme, which filters out relevant measurement traces to connect relevant users to the target user

as candidate measurements from a relevant perfSONAR measurements archive. The anomaly symptom detection module scans target and candidate measurements to detect anomaly symptoms for each measurement trace. After detecting anomaly symptoms, we quantify each anomaly symptom and construct an anomaly symptom matrix. Finally, the anomaly symptom similarity analysis module filters out those measurements traces with most similar anomaly symptoms. Hence, through similar anomaly symptoms, we can connect those users who also have similar network problems such that they can diagnose collaboratively and in a trustworthy manner.

In the following subsections, we describe how our collaborative filtering based recommender detects and classifies anomaly events and similarity between measurement traces.

2.4.1 Anomaly Symptom Detection

The first challenge of our collaborative filtering approach is how to detect anomaly symptoms in measurements i.e., how to classify anomaly events based on their features. Anomaly events could be visualized and analyzed in different approaches such as time-series, clusters or heat maps. In our research, we apply anomaly detection algorithms to network time-series measurements. As described in [32], network malfunctions are caused by different reasons. Moreover, different network

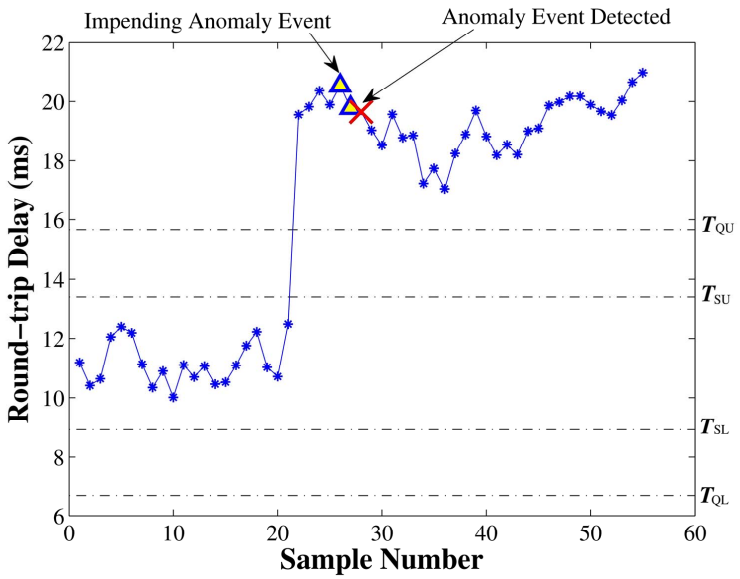


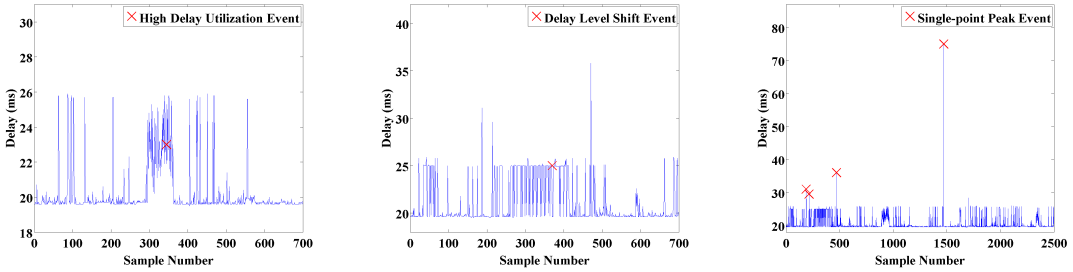
Figure 2.6: Plateau-detector thresholds illustration: normal state threshold is $[T_{SU}, T_{SL}]$, and abnormal state threshold is $> T_{SU}$ or $< T_{SL}$

malfunctions may have different anomaly symptoms (or signatures) in the time-series measurements. In this section, we describe how to detect anomaly symptoms and create an anomaly profile for each anomaly symptom, which can help us quantify anomaly symptoms for collaborative filtering and content-based filtering.

In addition, our anomaly symptoms detection algorithm is based on our previous work on the Adaptive Plateau Detector algorithm (APD) [48]. Before describing the anomaly symptoms detector, we describe the working of the APD algorithm briefly. Plateau events are used to detect performance change events in network environment, which look like a "plateau" in the measurements plots, as shown in Figure 2.6. A plateau trigger event is detected if the most recent measurement sample value crosses the upper or lower thresholds of the summary (i.e., T_{SU}, T_{SL}) and quarantine (i.e., T_{QU}, T_{QL}) buffers as determined by the settings of sensitivity and trigger elevation parameters. The summary buffer is used to maintain sample history that indicates the normal state values and a quarantine buffer is used to store outlier data samples that are more than twice the normal state sample values. A plateau anomaly is triggered when the count of trigger events reaches a value of pre-configured trigger duration, which is indicated by

the cross mark in Figure 2.6.

High delay utilization. "High delay utilization" may be caused by network congestion overload, which is a case of a significant and persistent queue backlog in one or more links along the path. As shown in Figure 2.7(a), the feature of "high delay utilization" has a high delay utilization duration, and we configure the parameter *trigger duration* of APD algorithm as shown in Table 2.1 to detect this symptom.



(a) High delay utilization (b) Delay level shift (c) Single point peak

Figure 2.7: The process of adjusting different parameters in the APD configurations to detect different anomaly symptoms

Delay level shift. The "delay level shift" may be caused by route changes or clock synchronization issue. It will be a significant change above normal baseline shown in Figure 2.7(b) and related configuration shown in Table 2.1.

Single point peak. The "Single point peak" may be affected by end-points, such as NIC (Network Interface Card) buffering issue, I/O issue and OS manual operations issues. So, this symptom is usually made of a single (or few) point(s) higher than normal delay as shown in Figure 2.7(c) and a related configuration is shown in Table 2.1.

For each anomaly symptom s, we quantify it as $s = (p, g)$, where the p denotes the type of anomaly symptom (such as "high delay utilization", "delay level shift"), and the value of p will be denoted as enumerative value $(0, 1, 2)$; And the g denotes the magnitude of anomaly symptom, which can be expressed as $(1-\frac{\overline{normal}}{\overline{abnormal}})$ means the difference ratio between average value of abnormal events and average value of normal events during a certain defined window size, so the value of $g \in [0, 1]$.

Hence, after anomaly symptom detection, we obtain a set of anomaly symptoms s for each measurement trace M_i, which could be expressed as $M_i = \{s_1, s_2, ..., s_m\}$.

Parameters	**High delay utilization**	**Delay level shift**	**Single point peak**
summary window count (swc)	100	100	100
trigger duration (td)	50	15	1
trigger threshold (tt)	Same as original APD	1.2	1.5
trigger method (tm)	Persistent and Intermittent	Persistent	Same with original APD

Table 2.1: Parameters configuration of APD to detect anomaly symptoms

2.4.2 Anomaly Symptom Matrix

In Section 2.4.1, we obtain a set of anomaly symptoms for each measurement trace by using anomaly symptom detection. The next step is to construct an anomaly symptom matrix D to compare similarities of measurement anomaly events.

However, constructing an anomaly symptom matrix is a challenging problem. A naive approach may scan all the anomaly symptoms and look for the most similar one to be aligned with target anomaly symptoms. However this approach is hard to operate and requires lots of computation resources.

To solve this problem, we propose using bins to align similar anomaly symptoms. This approach is easier to operate and requires lesser computation resources. The overall computation time of alignment processing could be achieved in *O(n)*. As the value of symptom magnitude g is between 0 and 1, our bin size will be defined within a 0.2 scale. However, a challenge with this approach is the lack of a way to analyze similarity with different anomaly symptom types. The value should be zero, if the types are different. Hence, we adjust them with different ranges. The magnitude of anomaly symptom S_i for particular anomaly symptom could be expressed as,

$$S_i = p_i + g_i \tag{2.11}$$

Where p_i equals to 0, 1, or 2, so the adjustment magnitude of anomaly symptom

with different anomaly type will be shifted to different range.

Hence, our anomaly symptom matrix is defined as: the X-axis has 15 number of bins with 0.2 scale for each bin; and the Y-axis is the measurements traces as shown in Figure 2.5.

2.4.3 Anomaly Symptom Similarity Analysis

After generating anomaly symptom matrix, we apply Pearson correlation coefficient to compute the anomaly symptom similarity score between two measurement trace M_i and M_j, which can be expressed as:

$$M\_sim(i,j) = \frac{\sum(M_i - \overline{M_i})(M_j - \overline{M_j})}{\sqrt{\sum(M_i - \overline{M_i})^2}\sqrt{\sum(M_j - \overline{M_j})^2}} \tag{2.12}$$

Through anomaly symptom similarity analysis, we could filter out the measurements with most similar anomaly symptoms when compared to the target measurement, so as to connect the relevant users to the target user for expertise sharing and cooperative mingling as shown in Figure 2.5.

2.4.4 Social Plane Implementation

In our mesurement recommender, we also implement our "social plane" approach social networking user interface in HumHub (A open source social network tool kit), which is social network open source development kit. When HumHub is deployed, it works as a common social network website as Facebook, Twitter, and it supports basic social network interfaces: users can create their own domains or communities; users can post their messages, which could be shared among their domains and their followers. We address challenges that could be faced by our proposed architecture implementation in a real deployment. The main challenge is to adapt our implementation of our proposed architecture such that it can be used routinely on a regular basis by many stakeholders. One strategy is to allow

users to subscribe and receive anomaly event notifications via email pertaining to various communities' measurement resources and data archives that are of direct interest in terms of performance monitoring objectives. Another strategy is to create relevant user interfaces and visualizations that allow stakeholders to use the social plane tools to collaborate around critical anomaly events in a 'systematic' manner through knowledge sharing and expert mingling for root-cause diagnosis.

In the following, we will describe the basic features of our "social plane" implementation.

Content-based Filtering Features. When users want to perform correlation analysis, the users have the option to find other measurements which are interest to them. The measurements can be searched based mainly on two objectives: Temporal and Spatial correlation analysis objectives. Based on the objectives selected, our recommendation scheme will recommend measurements which match the user's measurements. In our application, users in a particular domain only need to choose their analysis objectives (such as temporal, spatial), thereafter our system will automatically analyze a domain's measurements attributes (refer to Section 2.3) and recommends relevant measurements based on a user's analysis objective.

Collaborative Filtering Features. Users can subscribe to measurements by inputting e.g., the IP address of measurements archive, which are related to their services. After subscribing the measurements, users can observe these measurements, their end-to-end performance. When users find any suspect events, they can post the suspect events on the social web site by providing basic events information. The Collaborative filtering scheme is executed when users post a suspect event as described in the procedure in Section 2.4. After collaborative filtering, the users with similar anomaly will be filtered out as shown in Figure 2.8. Further, users can directly send messages to those for asking for help. Consequently, those users can build up connections and share diagnosis knowledge or work collaboratively.

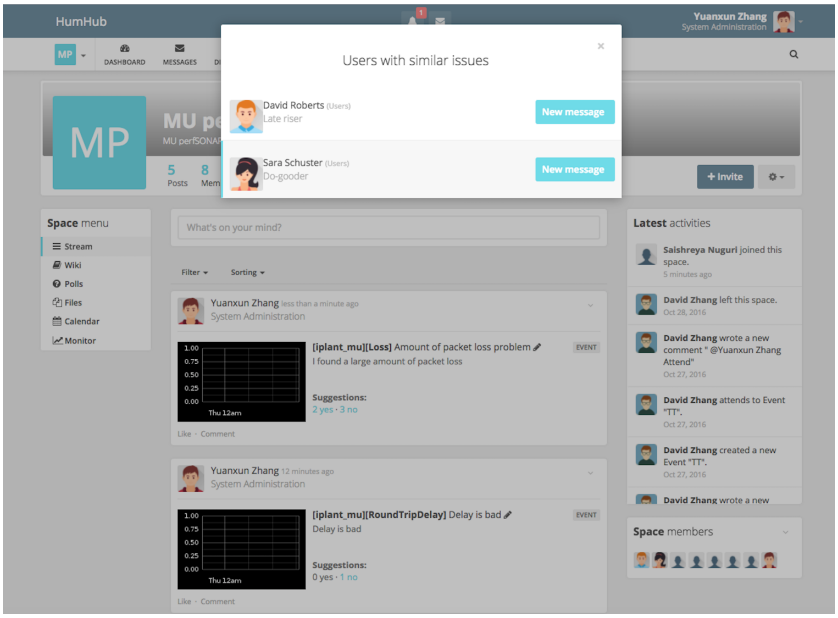


Figure 2.8: Screenshot of social plane implementation in Humhub

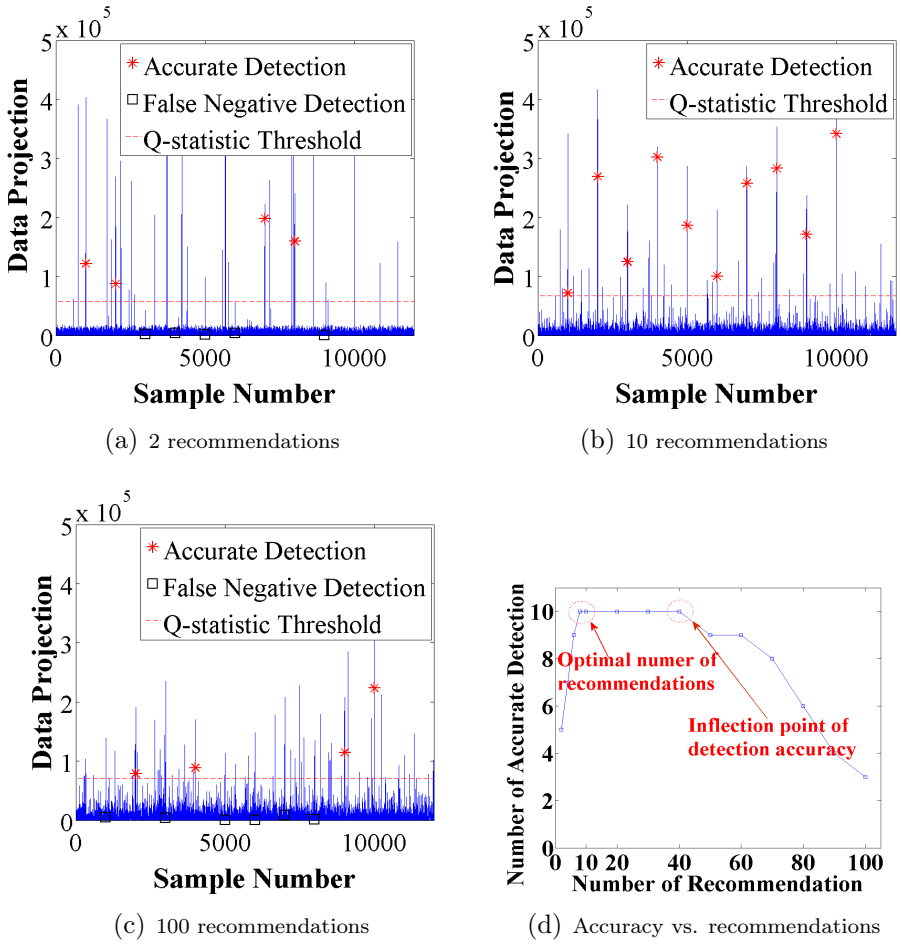
2.5 Evaluation

In the evaluation section, we separately evaluate the effectiveness of *content-based filtering* and *collaborative filtering* to show the effectiveness of our measurements recommendation scheme in accurately identifying anomaly events for a user's measurements analysis objective, and accurately identifying those anomaly events with similar symptoms. Because of the challenges in finding the ground-truth of measurements in real-world deployments, we create a simulation environment to synthesize measurements and anomaly events to prove our recommendation schemes.

2.5.1 Content-based Filtering Recommendation Performance Results

In order to examine the effectiveness of the proposed recommendation scheme in accurately identifying anomaly events upon analysis, we perform experiments with synthetic perfSONAR data. The synthetic data is carefully generated to closely mimic the actual perfSONAR OWAMP measurement traces. And then, we inject two types of anomaly events: correlated and uncorrelated anomaly events. Correlated anomaly events are temporal correlated anomaly events, which means

anomaly events from different traces happened at same time. In order to inject correlated anomaly events, we generate 100 traces and then inject anomaly events in those traces at the same time. We also inject events at random times as uncorrelated anomaly events. The percentage of anomaly events in each trace varies from 0.1%-1% of the trace sample population. The magnitudes of anomaly events vary from 10% - 60% over normal measurements with higher magnitudes causing sharper spikes.



(a) 2 recommendations

(b) 10 recommendations

(c) 100 recommendations

(d) Accuracy vs. recommendations

Figure 2.9: Accuracy of correlated anomaly detection in terms of false alarms with varying number of recommended traces

In the first experiment, we use the traces recommended by our scheme to detect network anomaly events using an exemplary temporal correlation analysis. For this experiment, we use different number of recommendations and see whether analysis with such recommendations can successfully identify the correlated/uncorrelated

anomaly events that we injected. In Figure 2.9(a), 2.9(b), and 2.9(c), we show the accuracy of such anomaly event detection with our scheme recommending 2, 10 and all 100 traces, respectively. We observe that in this particular scenario, 10 recommendations accurately detect all the anomaly events with no false alarms. Whereas analysis with only 2 recommendations lack the necessary data to establish correlation, thus causing false alarms. Further, all 100 recommendations suffer from too much noise in anomaly detection caused by undesired traces resulting in false alarms. In Figure 2.9(d), we showed the nature of detection accuracy with the number of recommended traces exhibiting an inflection within the 10 - 40 recommendations range. Thus, we argue that there exists: an optimal number of recommendations (in this case 10) for accurate detection, and an inflection point (in this case 40) beyond which too many traces contribute to high levels of noise resulting in false alarms.

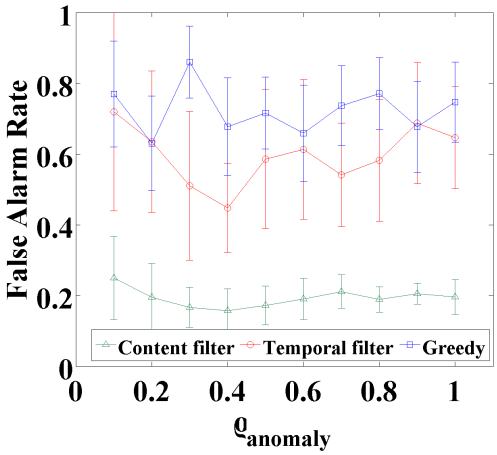


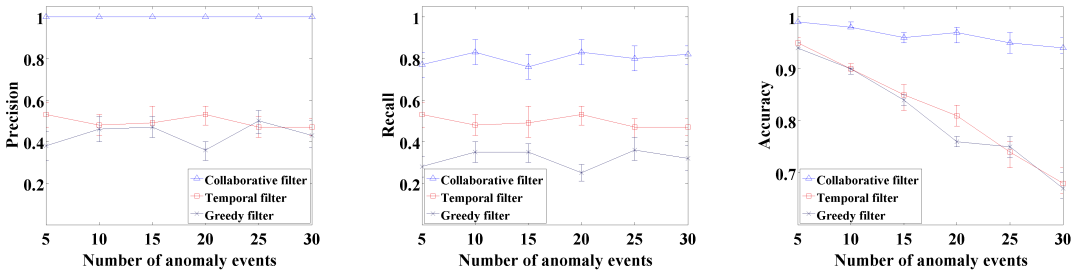
Figure 2.10: False Alarm Rate comparison among the three different schemes evaluated

Figure 2.10 shows the benefits of our content filter based recommendation scheme over the greedy recommendation approach (random selection), and recommendation strategy with filtering based on partial measurement features, such as temporal aspects (time range) of the traces. For this experiment, we vary the density of anomaly events and recommend equal number of traces for each approach being compared. We see that on an average, the content filter performs

Instance Type	100 Traces			1000 Traces			10000 Traces			100000 Traces		
	CF	GF	TF	CF	GF	TF	CF	GF	TF	CF	GF	TF
t2.small	4	3	2	33	20	21	340	209	211	22492	16347	16936
c4.large	3	2	2	28	18	17	277	175	174	3081	1942	1890

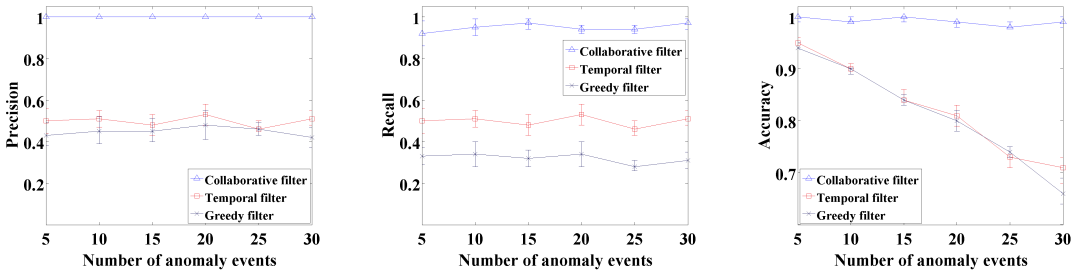
Table 2.2: Computation time (seconds) comparison of collaborative filtering (CF), greedy filtering (GF) and temporal filtering (TF)

consistently better than the greedy and temporal filter based approaches. It is interesting to observe that for a very small density of anomaly events, the false alarm rate is higher for all approaches. This is because, too few anomaly events with minimal anomalous features are difficult to detect and are neither related to the number of recommendations nor the filtering approaches.



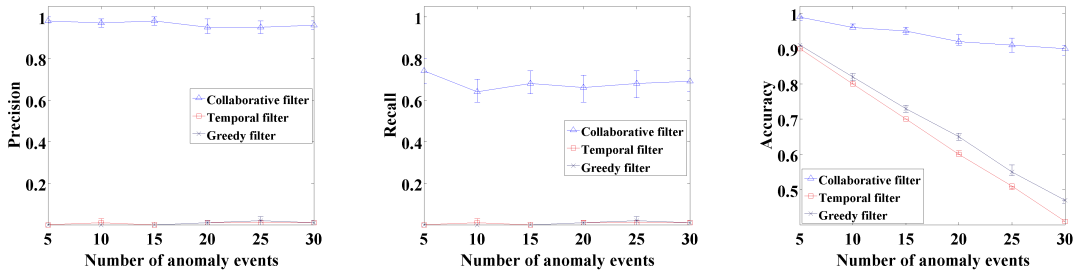
(a) Precision (b) Recall (c) Accuracy

Figure 2.11: Results of recommendation schemes to find anomaly symptom: "high delay utilization" with varying number of anomaly events



(a) Precision (b) Recall (c) Accuracy

Figure 2.12: Results of recommendation schemes to find anomaly symptom: "delay level shift" with varying number of anomaly events



(a) Precision (b) Recall (c) Accuracy

Figure 2.13: Results of recommendation schemes to find anomaly symptom: "single point peak" with varying number of anomaly events

2.5.2 Collaborative Filtering Recommendation Performance Results

To evaluate our collaborative filtering scheme for finding the most similar symptoms, we perform experiments with perfSONAR data as detailed in Section 2.5.1. We also evaluate the efficiency of our scheme to show the overload for accuracy.

- *Effectiveness*: We use three different metrics (Precision, Recall, Accuracy) in the effectiveness evaluation. These metrics are widely used in pattern recognition and information retrieval fields.
  - *Precision* is to indicate the fraction of recommended measurement traces that have similar anomaly symptom as the target measurement trace.
  - *Recall* is to indicate the fraction of measurement traces that have similar anomaly symptom as the target measurement trace that are recommended.
  - *Accuracy* is to indicate the fraction of measurement traces that have and don't have similar measurement traces that are accurately estimated.
- *Efficiency*: We use the average running time to evaluate efficiency of our scheme.

Effectiveness evaluation. We compare our collaborative filtering scheme with other two schemes: temporal filtering and greedy filtering schemes. temporal

filtering scheme recommends measurements with similar anomaly symptoms by checking temporal correlation level; and greedy filtering scheme recommends measurements by random selection.

In order to show the effectiveness of our scheme in dealing with each anomaly symptom, we separately evaluate our scheme in dealing with each anomaly symptom in terms of different number of anomaly events injected in candidate measurement traces. First, we select one measurement trace as a target measurement trace and 100 measurement traces as candidate measurement traces. Second, for each evaluation scenario, we inject one anomaly symptom into the target measurement trace, and then we randomly select number of measurement traces from candidate measurement traces to inject different types of anomaly symptoms.

For example, as shown in Figure 2.11, we inject the "high delay utilization" anomaly event into target measurement trace and inject different types anomaly symptoms to candidate measurement traces. The goal of these schemes is to find those measurements in candidate measurement traces that have similar anomaly symptoms as the target measurements. In Figure 2.11, we show that our collaborative filtering scheme has much better performance than the other two schemes with respect to the *Precision* and *Recall* metrics. We see that the results won't change as the number anomaly events are increased. However, in Figure , the collaborative filtering scheme also has higher performance, but it is not obvious as the previous two metrics. The reason for why temporal filtering and greedy filtering also have good accuracy is because of the number of true negatives, which means without any positive predication it can achieve 0.95 accuracy if there are 5 measurement traces with similar anomaly symptom. Also, the accuracy result decreases as the number of anomaly events increase. The other two evaluation scenarios for "delay level shift" in Figure 2.12 and "single peak point" in Figure 2.13 show the similar results. We also find that the "Temporal filter" and "Greedy filter" schemes are unable to make correct positive predictions, so the "precision" and "recall" results are almost close to zero.

Efficiency evaluation. In the efficiency evaluation studies, we compare the computation time for finding similar anomaly symptoms among these three approaches in terms of different number of measurements traces. Table 2.2 shows scalability results of our filtering algorithms when used on a large number of traces in a production scale environment. Specifically, we show computation time performance comparison of our filtering algorithm with greedy filtering and temporal filtering from 100 to 100,000 measurement traces. For this, we use different Amazon Web Services instances (see descriptions provided in Table 2.3) for analyzing scalability. We can observe that the computation time increases linearly as the number of measurements traces increases. Also, the overhead of our algorithm is minimal in production scale environments and is on the order of a few seconds even for several thousands of measurement traces. The computation time results of greedy filtering and temporal filtering are similar, and the computation time of collaborative filtering is ≈ 1.5 times larger than the other two schemes. This is an expected result, because the greedy and temporal filtering schemes need to run the APD algorithm only once. Whereas, our APD algorithm runs $1 \sim 3$ times in the case of the collaborative filtering scheme depending on how we randomly inject three different anomaly symptoms in the target measurement trace for efficiency evaluation.

Instance Type	Descriptions
t2.small	Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 2 GiB memory, EBS only
m3.medium	3 ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon E5-2670v2, 3.75 GiB memory, 1 x 4 GiB Storage Capacity
c4.large	8 ECUs, 2 vCPUs, 2.9 GHz, Intel Xeon E5-2666v3, 3.75 GiB memory, EBS only

Table 2.3: Descriptions of Amazon EC2 instances; *Legend:* ECU (EC2 compute unit), GiB (gibibite), EBS (elastic block store)

2.6 Case Studies

In this application testbed case studies section, we will show how our collaborative filtering scheme creates a "social plane" for helping application users or network administrators to share experience and diagnose collaboratively. We also show how our content-based filtering helps users find best matched measurements based on their measurement analysis objectives.

2.6.1 Case Studies of Collaborative Filtering

SoyKB [14] is a comprehensive web resource developed at University of Missouri (MU) for soybean translational genomics and breeding, which handles the integration of soybean genomics and multi-omics data along with gene function annotations, biological pathway and trait information. The SoyKB has been featured as a model distributed computing use case in the systems biology area within the Open Science Grid community. It represents a Big Data application in bioinformatics due to the large volume of distributed data sets that need to be integrated and analyzed.

In order to simulate various network issues in a Big data application, we use the GENI Cloud Testbed [49] to develop our case studies. First, we simulate the SoyKB Big Data application [19] related network environment in GENI. As shown in Figure 2.14, SoyKB Application imports raw data from other counties (such as China, Brazil). Researchers at MU use iPlant and TACC resources to store and process. They may also choose public cloud (AWS, Azure) to achieve better performance. Same storage (iPlant) and computing (TACC) resources could also be used by researchers from Iowa and Arkansas. Second, we deploy an end-to-end monitoring infrastructure to collect measurements using our OnTimeMeasure software for GENI [50]. The measurements are collected by a 'Measurements engine' shown in Figure 2.14. Third, we use the "netem" command [51] to control delay values and duration time to simulate different anomaly symptoms, as shown

in Table 2.4. Note that the arrows in Table 2.4 denote the increase in the value of “delay" to simulate different types of anomaly symptoms. Lastly, we use our collaborative filtering scheme to create a social plane for sharing knowledge or collaborative diagnosis in a trustworthy manner.

Anomaly symptom type	variation	duration(s)
High delay utilization	$\uparrow (20 \sim 40)\%$	3600s
Delay level shift	$\uparrow (55 \sim 60)\%$	Persistence
Single point peak	$\uparrow 60\%$	10s

Table 2.4: Simulation anomaly symptoms with “netem” commands

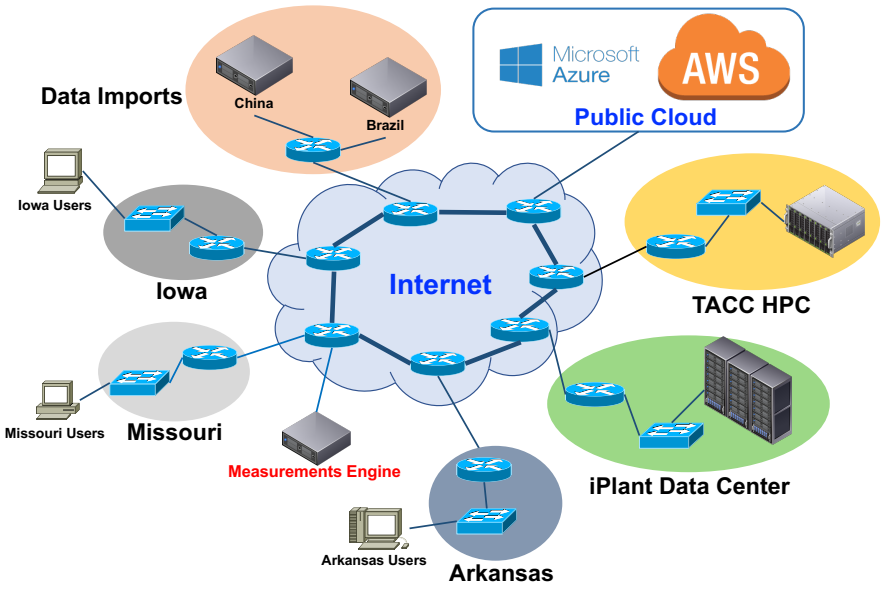


Figure 2.14: Physical simulation of SoyKB application testbed in GENI Cloud Infrastructure

We will now present a few scenarios which are most common cases in scientific Big Data applications, such as different user experiences with same applications; issues located at sources (e.g., users’ side) which may impact on those users’ application performance; and issues located at sinks (e.g., data centers or computation centers) which may impact on most users’ application performance.

User Experience Issue. User experience issue suggests that a user has bad or unsatisfactory experiences of application performance, such as long data transfer times, excessive execution times, or slow SSH login speed. The cause of those issues

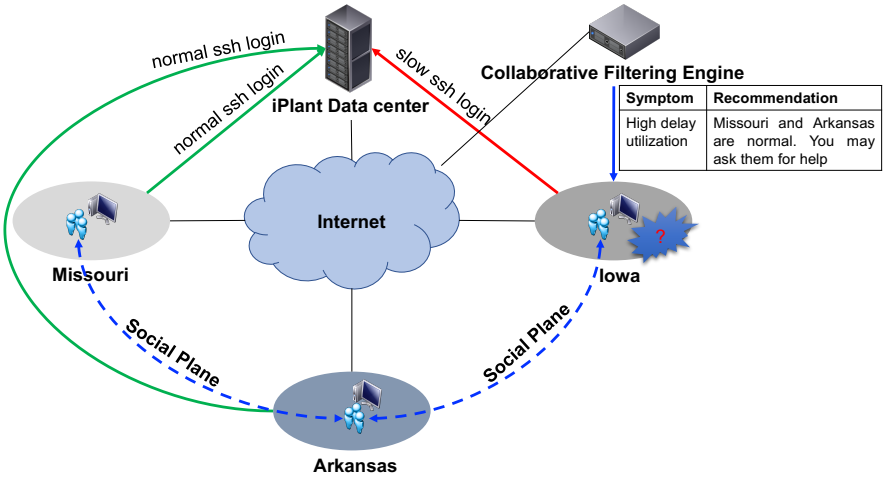


Figure 2.15: Logical illustration of User Experience bottleneck case: users in "Iowa" feel the high latency SSH operation on iPlant server

are usually complicated and irregular, which may be caused by mis-configurations, network bottlenecks, or firewall security issues.

As shown in Figure 2.15, the users in Iowa find the speed of using SSH to login into the iPlant data center for SoyKB related computations is very slow. After analysis using our collaborative filtering engine (labeled in Figure 2.15), the engine detects a "high delay utilization" symptom in the measurement trace from "Iowa" to "iPlant", and the engine also finds measurements traces from "Missouri" or "Arkansas" to iPlant without this kind of issues. So, the engine may recommend users in "Iowa" ask users in "Missouri" or "Arkansas" for help, so that a "social plane" is created among those users for sharing knowledge or collaborative troubleshooting. This example also could be the scenario of issues located at sources, because only certain users face them.

Data Center Issue. Data center issue (or called issues located at data sink) means that the data center faces some issues which are caused by data centers' device failures or link failures, as a result of which, many users or entities will be affected. Those issues may not directly impact on user's experiences, however, those issues may be manifest as notable changes in measurements, which catch users' attention.

As shown in Figure 2.16, the users in Missouri observe the performance change

Trace Name	Metric	Periodicity	Time Range
bnl↔fnal	**One-way delay**	**[58, 62]**	**09-01 00:57 ↔ 09-09 23:58**
lbl↔ornl	One-way delay	[51, 68]	09-03 05:39 ↔ 09-03 22:19
aofa↔bost	One-way delay	[51, 161]	09-01 00:00 ↔ 09-01 05:35
bnl↔bois	One-way delay	[57, 63]	09-01 00:00 ↔ 09-09 23:59
sacr↔bois	One-way delay	[53, 150]	09-01 00:00 ↔ 09-01 16:42
bnl ↔bost	One-way delay	[53, 67]	09-01 00:00: ↔ 09-01 05:35
hous ↔srs	One-way delay	[53, 69]	09-05 05:31 ↔ 09-05 22:08
bnl ↔lbl	One-way delay	[61, 66]	09-01 00:00 ↔ 09-09 23:59
newy ↔sacr	Throughput	[54, 67]	09-01 00:00 ↔ 09-09 23:59
anl ↔newy	Throughput	[52, 64]	09-01 00:01 ↔ 09-09 23:59
bnl ↔nash	Throughput	[57, 71]	09-03 05:58 ↔ 09-03 22:38
denv ↔fnal	Throughput	[58, 67]	09-01 00:00 ↔ 09-09 23:59

Table 2.5: Real perfSONAR measurements traces' attributes description

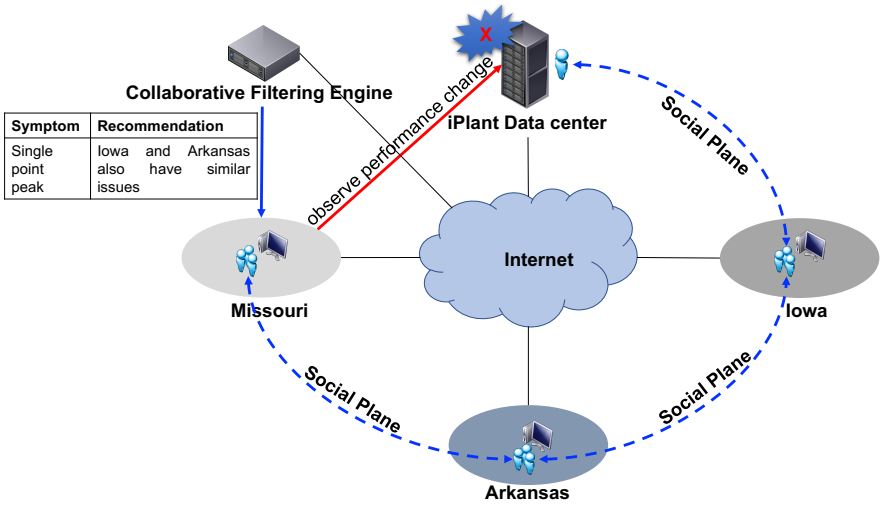


Figure 2.16: Logical illustration of data center issue case: data center iPlant has some issues, which would affect all the iPlant users

in their measurements. After analysis with our collaborative filtering engine, we detect a "Single point peak" symptom in the measurement trace from "Missouri" to "iPlant", and also detect similar anomaly symptoms in measurements traces from "Arkansas" or "iPlant" and from "Iowa" to "iPlant". Consequently, the engine may suggest diagnosing collaboratively work with users in "Arkansas" and "Iowa". And because they have same sink (iPlant), they may also check this issues with administrators of iPlant. Consequently, a "social plane" is created among those users and service provider for sharing knowledge or collaborative troubleshooting of the root-cause issue.

2.6.2 Case Studies of Content-based Filtering

We collect hundreds of perfSONAR traces from National labs and ESnet sites with perfSONAR end-points with different measurement attributes as inputs to our proposed measurement recommendation scheme. In Table 2.5, we show the attributes for only a small subset of collected samples. In this subset, we have kept the trace BNL↔FNAL as the target trace (in bold font) and the rest as candidate traces. Applying our proposed recommendation scheme, we seek to find the most relevant traces for two separate measurement analysis objectives: a) Temporal analysis for correlated anomaly event detection [47] where complete topology information is not available; and b) Spatial analysis for topology-aware correlated anomaly event detection [28]. We remark that recommendations in a spatial analysis context are as useful as the amount of topology details available between the various measurement end-points. In reality, it is not always possible to obtain topology information for the measurement traces in the publicly available perfSONAR data archives. Consequently, recommendations might not be useful for root-cause diagnosis in cases where there is an absence of topology information.

Table 2.6 shows the measurement attributes similarity scores for the candidate measurement traces with the target trace described in Table 2.5. The score evaluations follow the scheme described in Section 2.3 and due to the varied attributes of the collected traces, we observe that the similarity scores of candidate traces based on different attributes can vary by a considerable margin.

In Figure 2.17, we show the historical reputation characteristic comparison of 3 exemplar National Lab sites based on one year (Oct 2014 - Oct 2015) traces' data sanity scores using the scheme discussed earlier. We observe that although these are well known and seemingly reputed perfSONAR end-points within the Big Data application communities deploying perfSONAR, not all the sites produce trustworthy data at all times. Thus, we establish the need for a domain's reputation as a key factor in subscribing to that domain's measurement data for accurate detection and effective troubleshooting. For example, according to Fig-

Trace Name	Topology Similarity	Metric Similarity	Alignment Similarity	Time Range Similarity
lbl↔ornl	0	1	0.456	0.077
aofa↔bost	0.077	1	0	0.026
bnl↔bois	0.385	1	0.999	1
sacr↔bois	0	1	0.103	0.077
bnl↔bost	0.4	1	0	0.026
hous↔srs	0	1	0.767	0.53
bnl↔lbl	0.333	1	0.999	1
newy↔sacr	0.125	0	0.999	1
anl↔newy	0	0	1	0.999
bnl↔nash	0.143	0	0.979	0.862
denv↔fnal	0.154	0	0.999	1

Table 2.6: Measurement attributes similarity score description

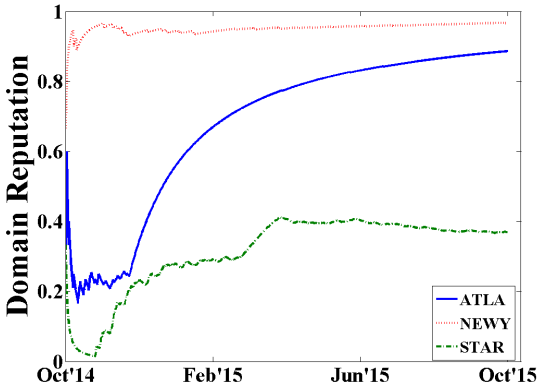


Figure 2.17: Historical reputation characteristics comparison of 3 exemplar DOE lab with real perfSONAR traces over one year period

ure 2.17, subscribing to data from domain 'NEWY and 'ATLA' is likely to yield data with high veracity; whereas, subscribing to STAR data may not always lead to accurate correlated anomaly event detection and diagnosis.

Subsequently, we apply the relative weights of the measurement attributes on the similarity scores based on the two monitoring objectives: temporal and spatial. For temporal analysis, we focus on detecting correlated anomaly events in time series measurements where measurement metric needs to be of similar type. Hence, we follow the path: 'Analysis Objective' → 'Temporal' → 'Short-term Change' → 'Measurements Metric Specific', and assign the weights according to the rule $w_a > w_r > w_m > w_t$. Whereas for spatial analysis, focus is on using measurements topological information to drill down the location of events. Hence,

Trace	Sanity	Temporal Correlation		Spatial Correlation		Domain Reputation	
Name	Score	Similarity Score	Ranking	Similarity Score	Ranking	Source	Destination
bnl↔bois	0.993	0.938	**1**	0.938	**1**	0.983	0.984
bnl↔lbl	**0.642**	0.933	**2**	0.933	**2**	0.985	**0.611**
denv↔fnal	**0.972**	**0.765**	3	**0.327**	5	0.991	0.984
newy↔sacr	0.982	0.762	4	0.312	7	0.979	0.986
anl↔newy	0.984	0.221	5	0.273	10	0.987	0.986
bnl↔bost	0.953	0.197	10	0.453	3	0.983	0.964
hous↔srs	0.976	0.666	7	0.418	4	0.934	0.986

Table 2.7: Data sanity score and domain reputation results for selected traces used in exemplar analysis case study

it follows the path: 'Analysis Objective' → 'Spatial' → 'Measurements Metric Specific' → 'Short-term Change', with final relative weights following the rule $w_t > w_m > w_a > w_r$.

The final recommendation outcomes and corresponding ranking of a subset of candidate traces are shown in Table 2.7 along with the traces' data sanity scores, and corresponding source and destination domains' reputation scores. Table 2.7 is a snapshot of the actual manifestation of our proposed recommendation scheme to assist the operators and application users to better gauge the relevance and veracity of collected samples. For example, the Trace BNL↔BOIS (shown in Table 2.7) is the best choice among the candidates (1^{st} ranked) in terms of similarity and high instantaneous data sanity score (0.993). Whereas, Trace BNL↔LBL, although being 2^{nd} ranked for both temporal and spatial analysis, may not be a good choice for candidacy as the low sanity score (0.642) suggests sub-par data quality which can be attributed to low reputation (0.611) of the destination domain (LBL). Thus, operators should be advised to use Trace DENV↔FNAL over Trace BNL↔LBL for temporal analysis as the data quality of the former is much better (0.972) in spite of having a slightly lower similarity (0.765). However, DENV↔FNAL will not be a significantly better choice over BNL↔LBL for spatial analysis due to the former's very low similarity score (0.327).

2.7 Summary

In this chapter, we showed the need of a "measurement recommender" for network operators and users to effectively troubleshoot bottlenecks affecting Big Data applications in a trustworthy manner. We leveraged the concept of recommendation system to enhance measurements sharing/subscription, diagnosis expertise sharing and collaborations. Using content-based filtering, we filter and rank best relevant measurement traces from a pool of candidate traces. The recommendation scheme was complemented by a Bayesian Inference based domain reputation calculation scheme that indicates the trustworthiness of the collected samples among the involved domains. With collaborative filtering, we find those measurement traces having similar anomaly symptoms with target measurements, and through measurements traces, we can connect those people for sharing knowledge, or working collaboratively in a trustworthy manner. Using real measurements traces and synthetic events, we showed how our content-based filtering scheme enables operators to intelligently use less but relevant measurement samples to accurately detect and diagnose performance bottleneck causing network events. In addition, we showed how our collaborative filtering scheme finds similar anomaly issues efficiently and effectively.

Our future work is to adapt our "measurement recommender" approach to be more deeply integrated into diverse Big Data application communities, to help them better socialize around measurements and achieve expected performance.

Chapter 3

Domain-specific Topic Recommender for Knowledge Discovery

In this chapter, we present our topic recommender by proposing a novel Domain-specific Topic Model (DSTM) algorithm to discover latent knowledge patterns about relationships among research topics, tools and datasets from exemplary scientific domains. Our DSTM is a generative model that extends the Latent Dirichlet Allocation (LDA) model and uses the Markov chain Monte Carlo (MCMC) algorithm to infer latent patterns within a specific domain in an unsupervised manner.

3.1 Background and Related Work

3.1.1 Topic Model

Topic models have been found to be successful in automatically extracting useful information from text corpus in an unsupervised learning manner. Latent Dirichlet Allocation (LDA) [20] is one of the most popular topic models that was invented by Blei *et al.* in 2003. It discovers the latent topic structures from a collection

of documents or text corpus automatically. In LDA, each topic is modeled as a distribution over words, and each document is represented as a mixture over topic proportions. The LDA model has been widely applied to document classifications, searching, and recommendations.

Based on the LDA model, many researchers have tried to propose model variants for discovering different patterns of documents. Rosen-Zvi *et al.* [21] proposed an Author-Topic model that extends the LDA by including authorship information to establish the relationships between topics and authors. Relationships between topics and authors are explored by representing each author with a mixture of weights for different topics.

Mimno *et al.* [52] also proposed a similar author topic model for matching papers with peer-reviewers. Blei *et al.* [53] proposed a dynamic topic model in order to extract the evolution of topics within sequentially organized documents. Blei and Lafferty in [54] proposed a correlated topic model (CTM) to demonstrate the correlations between topics using a logistic normal distribution on the simplex to model dependence between two topics. This distribution represents the correlations between components.

Other researchers also successfully applied latent based topic modeling to different areas. Li *et al.* [55] adapted the LDA model for image scene categorization without any human annotations, which achieved comparable performance. Flaherty *et at.* [56] developed a model that is able to cluster genes within experiments that do not require inputs of a gene or drug. Wang *et al.* [57] combined the collaborative filtering and probabilistic topic models (i.e., LDA variants) in a recommendation system to recommend scientific articles. Their model leverages collaborative filtering and a topic model to perform matrix factorization that decomposes a rating matrix into latent users and items structure. Luo *et al* [58] proposed a generative probabilistic model for optimizing workforce personnel allocation using employee-activity logs. They used latent variables to learn hidden patterns between employees and activities in terms of their job performance.

3.1.2 Inference Algorithm

Inferring the latent variables in a probabilistic model (such as the LDA) has also been an area of active research investigations. The original LDA work [20] used a variational expectation maximization algorithm to estimate latent parameters. Hoffman *et al.* [59] designed an online stochastic optimization with a natural gradient step. Their optimization results showed the variational Bayes objective function convergence to a local optimum. Griffiths *et at.* [60] presented a collapsed Gibbs sampling algorithm (i.e., the Markov chain Monte Carlo method) to infer latent parameters of their model. In our work, we investigated many of the inference algorithms listed above, and decided to use the Gibbs sampling method. This is primarily because of the ease of implementation property of the Gibbs sampling method that does not however, compromise the learning speed and generalization performance.

3.2 Domain-specific Topic Model

In this section, we first detail our Domain-specific Topic model (DSTM) in terms of its generative process, inference algorithm and model parameter estimation. Following this, we present three possible applications for the DSTM to be used for discovery of knowledge patterns for guiding researchers in computational and data-intensive scientific communities.

3.2.1 The Generative Model

The purpose of our DSTM generative model is to discover the latent knowledge patterns underlying a collection of documents for a specific scientific domain in terms of the relationships among research topics, tools, and datasets. The generative model refers to our text modeling approach, where we generate each word in a document based on the distributions of words. As opposed to the LDA model

Table 3.1: Notations for the generative model

Symbols	Description
D	a collection of documents $D = \{d_1, ..., d_n\}$
K	the number of topics
T	a set of tools
S	a set of datasets
V	a set of words in the vocabulary
N_d	the number of word tokens in a document d
$\mathbf{t}_d$	a set of tools in a document d
$\mathbf{s}_d$	a set of dataset in a document d
$w_{dn}, \mathbf{w}$	the n^{th} word token in a document d
$x_{dn}, \mathbf{x}$	tool indicator chosen from $\mathbf{t}_d$ for word w_{dn}
$y_{dn}, \mathbf{y}$	dataset indicator chosen from $\mathbf{s}_d$ for word w_{dn}
$z_{dn}, \mathbf{z}$	the topic assignment for word w_{dn}
$L_{dn}, \mathbf{L}$	binary indicator to label which is responsible for z_{dn}
$\pi_d, \boldsymbol{\pi}$	Bernoulli parameter for generating label $\mathbf{L}$ for a document d
$\boldsymbol{\eta}$	$(\eta_{\pi_0}, \eta_{\pi_1})$ parameters for Beta distribution prior
$\phi_z, \boldsymbol{\phi}$	multinomial distribution over words specific to a topic z
$\theta_t, \boldsymbol{\theta}$	multinomial distribution over topics specific to a tool t
$\lambda_s, \boldsymbol{\lambda}$	multinomial distribution over topics specific to a dataset s
α, β, γ	parameters of symmetric Dirichlet priors

that generates each word based on random topics, our model generates each word based on reference tools or datasets occurrence in a document. In the generative process, we do not assume that a tool and/or dataset is responsible for a certain word simultaneously. For simplifying the computational complexity, each word is generated by either a tool or a dataset. Considering this sentence that is extracted from the BMC Bioinformatics journal: "We tested the performance of three widely used short-read alignment tools (*BWA*, *Bowtie* and *Bowtie2*) on simulated sequencing runs of varying coverage", we can intuitively infer that the tools *BWA*, *Bowtie* and *Bowtie2* contribute to generate the bioinformatics topic "short-read alignment tools".

The graphical representation of our generative model is illustrated in Figure 3.1 using a plate notation with all of the various notations summarized in Table 3.1. During the pre-processing stage, we collect papers from particular collections of documents $D = \{d_1, ..., d_n\}$. Next, we label the corresponding tools and dataset categories mentioned in each document based on our collections of tool names and

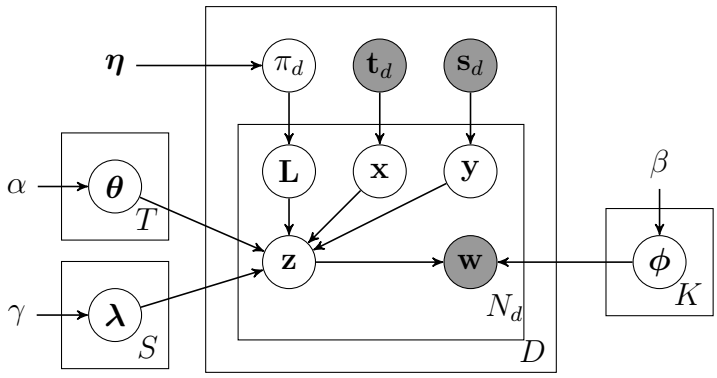


Figure 3.1: Graphical representation of the generative model. The boxes are "plates" representing replicates; the "shaded" nodes are observed variables; the "unshaded" nodes are unobserved variables; the nodes without cycle are hyperparameters; See Table 3.1 for node notations.

dataset categories provided by a domain scientist as domain-specific knowledge. A document d is represented as a bag-of-words with N_d unique word tokens, and the n^{th} word in document d is denoted as w_{dn}. T denotes the total number of tools and S denotes the total number of dataset categories we created in a catalog. In each document d, the word is an observed variable with "shaded" color, and the other observed variables are a set of tools $\mathbf{t}_d$ and a set of dataset categories $\mathbf{s}_d$. In the model, we assume that there are K number of topics for collection documents D. $\boldsymbol{\phi}$ denotes the $K \times V$ matrix of topics distribution over vocabulary V. $\boldsymbol{\theta}$ denotes the $T \times K$ matrix of tools distribution over topics, and $\boldsymbol{\lambda}$ denotes the $S \times K$ matrix of datasets distribution over topics. L is a binary indicator variable to label whether the topic assignment is from the tool distribution $\boldsymbol{\theta}$ or the dataset distribution $\boldsymbol{\lambda}$.

Algorithm 1 describes the generative process of the model. First, each topic is associated with a multinomial distribution over V vocabulary drawn from symmetric $Dirchlet(\beta)$ prior. Each tool t draws a multinomial distribution over topics from $Dirchlet(\alpha)$ prior, represented by θ_t. And each dataset s draws a multinomial distribution over topics from $Dirchlet(\gamma)$ prior, denoted as λ_s. Second, for each word in document d, we draw a binary indicator $\mathbf{L}$ from $Bernoulli(\pi_d)$ distribution to decide whether this word is generated by a tool or a dataset. The $Bernoulli(\pi_d)$ distribution is applied when both $\mathbf{t}_d$ and $\mathbf{s}_d$ are not empty. If either of them is

Algorithm 1 Generative process in the model

for each topic $k = 1, ..., K$ **do**
    Draw a multinomial over vocabulary $\phi_k \sim Dir(\beta)$;
end for
for each tool $t = 1, ..., T$ **do**
    Draw a multinomial over topics $\theta_t \sim Dir(\alpha)$;
end for
for each dataset $s = 1, ..., S$ **do**
    Draw a multinomial over topics $\theta_t \sim Dir(\gamma)$;
end for
for each doc $d = 1, ..., D$ **do**
    Sample binary indicator $L_{dn} \sim Bern(\pi_d)$;
    if $L_{dn} == 0$ **then**
        Select a tool $x_{dn} \sim Unif(\mathbf{t}_d)$;
        Sample a topic $z_{dn} \sim Multi(\theta_{x_{dn}})$;
    end if
    if $L_{dn} == 1$ **then**
        Select a dataset $y_{dn} \sim Unif(\mathbf{s}_d)$;
        Sample a topic $z_{dn} \sim Multi(\lambda_{y_{dn}})$;
    end if
    Choose a word $w_{dn} \sim Multi(\phi_{k=z_{dn}})$;
end for

empty, the $\mathbf{L}$ is assigned to the non-empty one. Then, a tool or a dataset is chosen from either set of tools ($\mathbf{t}_d$) or set of datasets ($\mathbf{s}_d$) randomly and uniformly. A topic assignment z_{dn} is selected based on the tools ($\boldsymbol{\theta}$) or datasets ($\boldsymbol{\lambda}$) distributions over topics. Finally, a word is generated according to topic distribution ($\boldsymbol{\phi}$) over words.

3.2.2 Inference and Parameter Estimation

In this subsection, we describe the algorithm to estimate the latent variables by using the Gibbs sampling method [60]. More specifically, we explain how we use the Gibbs sampling method [60] to infer and estimate latent variables $\{\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{z}, \mathbf{x}, \mathbf{y}, \boldsymbol{\pi}, \mathbf{L}\}$ of the model. To build the Gibbs sampling, we construct a posterior distribution of latent variables conditioned on all other variables, and repeatedly sample from the conditional probability distribution until it converges to a target or equilibrium distribution. In practice, we do not need to construct Gibbs sampling equations for each latent variable. By taking advantage of con-

jugate prior, the latent variables $\{\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \boldsymbol{\phi}\}$ can be integrated out as follows: Dirichlet is the conjugate prior of multinomial, and Beta is the conjugate prior of Bernoulli. Using density estimation of $\mathbf{x}, \mathbf{y}, \mathbf{z}$, we can still estimate $\{\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}\}$ through posterior distribution. To simplify equations, we define the set of hyper-parameters as $\boldsymbol{\Omega} = \{\alpha, \beta, \gamma, \boldsymbol{\eta}, T, S\}$.

Inference Algorithm

For each n^{th} word of document d, we construct Gibbs sampling equation for label L_{dn}, topic assignment z_{dn}, and tool assignment x_{dn} or dataset assignment y_{dn} jointly as a block $(L_{dn} = 0, z_{dn}, x_{dn})$ or $(L_{dn} = 1, z_{dn}, y_{dn})$ conditioned on all other variables. Then, the full conditional probability for considering the n^{th} word generated by tool $(L_{dn} = 0, z_{dn} = k, x_{dn} = t)$ is as follows:

$$\begin{aligned} &\mathrm{P}(L_{dn} = 0, z_{dn} = k, x_{dn} = t | \mathbf{L}_{-dn}, \mathbf{x}_{-dn}, w_{dn}, \mathbf{z}_{-dn}, \mathbf{w}_{-dn}, \mathbf{y}, \boldsymbol{\Omega}) \\ =&\frac{C^L_t + \eta_{\pi_0} - 1}{C^L_t + C^L_s + \eta_{\pi_0} + \eta_{\pi_1} - 1} \times \frac{C^{TK}_{tk,-dn} + \alpha}{\sum_k C^{TK}_{tk,-dn} + K\alpha} \times \frac{C^{VK}_{vk,-dn} + \beta}{\sum_v C^{VK}_{Vk,-dn} + V\beta} \end{aligned} \tag{3.1}$$

where C^L_t is the number of times including current instance that a tool is selected for generating word in document d, C^L_s is the number of times including current instance that dataset is selected for generating a word in document d, $\mathbf{L}_{-dn}$ denotes all the label assignments excluding the current instance. C^{TK}_{tk} is the number of times tool t is assigned to topic k, and the subscript $C^{TK}_{tk,-dn}$ denotes the exclusion of the current instance. C^{VK}_{vk} is the number of times word v in vocabulary V is assigned to topic k, and the subscript $C^{VK}_{vk,-dn}$ denotes excluding the current instance.

Similarly, the full conditional probability considering the n^{th} word generated by dataset $(L_{dn} = 1, z_{dn} = k, y_{dn} = s)$ is as follows:

$$\begin{aligned} &\mathrm{P}(L_{dn} = 1, z_{dn} = k, y_{dn} = s | \mathbf{L}_{-dn}, \mathbf{y}_{-dn}, w_{dn}, \mathbf{z}_{-dn}, \mathbf{w}_{-dn}, \mathbf{x}, \boldsymbol{\Omega}) \\ =&\frac{C^L_s + \eta_{\pi_1} - 1}{C^L_t + C^L_s + \eta_{\pi_0} + \eta_{\pi_1} - 1} \times \frac{C^{SK}_{sk,-dn} + \gamma}{\sum_k C^{SK}_{sk,-dn} + K\gamma} \times \frac{C^{VK}_{vk,-dn} + \beta}{\sum_v C^{VK}_{Vk,-dn} + V\beta} \end{aligned} \tag{3.2}$$

where C_{sk}^{SK} represents the number of times dataset s is assigned to topic k, with the subscript $C_{sk,-dn}^{SK}$ denoting the exclusion of the current instance.

Having obtained the full conditional distributions from Equations 3.1 and 3.2, the whole Gibbs sampling algorithm is straightforward. First, we initialize the variables $\{\mathbf{L}, \mathbf{z}, \mathbf{x}, \mathbf{y}\}$ randomly. Then, in each iteration, we update $\{\mathbf{L}, \mathbf{z}, \mathbf{x}, \mathbf{y}\}$ in turn from the full conditional distributions with Equations 3.1 and 3.2, until it converges to a target distribution.

Parameter Estimation

Collecting sets of samples $\mathbf{L}, \mathbf{z}, \mathbf{x}, \mathbf{y}$ obtained from the Gibbs sampling algorithm, we can estimate variables $\{\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \boldsymbol{\pi}\}$ with expectation of posterior distribution. The posterior distribution of topics k over vocabularies ϕ_k is written as,

$$\begin{aligned} \mathrm{P}(\phi_k|\mathbf{z}, \mathbf{w}, \beta) &\propto \mathrm{P}(\mathbf{w}|\mathbf{z}, \phi_k)\mathrm{P}(\phi_k|\beta) \\ &\propto \prod_{d=1}^{D}\prod_{n=1}^{N_d} \phi_{k,w_{dn}} \prod_{v=1}^{V} \phi_k^{\beta-1} \\ &= \prod_{v=1}^{V} \phi_k^{C_{vk}^{VK}+\beta-1} = \mathrm{Dir}(C_{vk}^{VK} + \beta) \end{aligned} \tag{3.3}$$

Then, the expectation of the Dirichlet distribution to estimate parameter ϕ_{vk}, which is the probability of vocabulary v assigned to topic k for any single sample, is given as,

$$\phi_{vk} = \frac{C_{vk}^{VK} + \beta}{\sum_v C_{vk}^{VK} + V\beta} \tag{3.4}$$

Similarly, the parameter estimations of θ_{tk} which is the probability of tool t assigned to topic k, and λ_{sk} which is the probability of dataset s assigned to topic k are as follows:

$$\theta_{tk} = \frac{C_{tk}^{TK} + \alpha}{\sum_k C_{tk}^{TK} + K\alpha} \tag{3.5}$$

$$\lambda_{sk} = \frac{C_{sk}^{SK} + \gamma}{\sum_k C_{sk}^{SK} + K\gamma} \tag{3.6}$$

Next, the posterior distribution of π_d that describes the probability of choosing a tool or dataset for generating a word, is written as,

$$\begin{aligned} \mathrm{P}(\pi_d|\mathbf{L}, \boldsymbol{\eta}) &\propto P(\mathbf{L}|\pi_d)P(\pi_d|\boldsymbol{\eta}) \\ &\propto \pi_d^{C_t^L}(1-\pi_d)^{C_s^L}\pi_d^{\eta_{\pi_0}-1}(1-\pi_d)^{\eta_{\pi_1}-1} \\ &= \pi_d^{C_t^L+\eta_{\pi_0}-1}(1-\pi_d)^{C_s^L+\eta_{\pi_1}-1} \\ &= \mathrm{Beta}(C_t^L + \eta_{\pi_0}, C_s^L + \eta_{\pi_1}) \end{aligned} \tag{3.7}$$

Lastly, the expectation of the Beta distribution to estimate the probability of choosing tools or datasets for a document d is as follows:

$$\pi_d^{L_{dn}=0} = \frac{C_t^L + \eta_{\pi_0}}{C_t^L + C_s^L + \eta_{\pi_0} + \eta_{\pi_1}} \tag{3.8}$$

$$\pi_d^{L_{dn}=1} = \frac{C_s^L + \eta_{\pi_1}}{C_t^L + C_s^L + \eta_{\pi_0} + \eta_{\pi_1}} \tag{3.9}$$

3.2.3 Applications of Knowledge Pattern Discovery

We now present three applications for our DSTM to be used for discovery of knowledge patterns for guiding researchers via user interfaces that feature conversational agents (or chatbots) as outlined in [61]. Through an interactive interface, DSTM can help the researchers in their common research activities involving: (i) *Intra-domain Knowledge Pattern Representation* that can be directly achieved by DSTM when used to explore potential tools or datasets in individual domains to solve a research problem with better performance; (ii) *Cross-domain Knowledge Pattern Representation* that involves a topic embedding solution for visualizing topics from multiple domains in a two-dimensional space in order to allow researchers to explore close topics and adapt successful solutions from other domains; and

(iii) *Trend Knowledge Pattern Representation* that tracks the change in trends or reveals emerging trends over time, in order to guide researchers to make intelligent decisions on choosing the pertinent tools or datasets relevant to solving their research problem at hand.

Intra-domain Knowledge Pattern Representation

Researchers seek to discover intra-domain knowledge patterns pertaining to tools or datasets from existing solutions or prior successful experiences in a given scientific domain. The algorithm for implementing this DSTM application for intra-domain knowledge pattern representation is shown in Algorithm 2. To understand the algorithm, we can consider the following example: a research group has applied a deep learning method to achieve better performance in bioinformatics tasks such as sequence analysis and structure prediction. In this context, the knowledge pattern from their solution can be learned and re-purposed by other bioinformatics scientists for similar research tasks. The algorithm to obtain intra-domain knowledge patterns involves training a DSTM object using a dataset (documents D, tools T, datasets S) from an individual domain (e.g., bioinformatics or neuroscience that are exemplar for the purposes of our work) and generation of topic distribution $\boldsymbol{\phi}$, tool distribution $\boldsymbol{\theta}$, and dataset distribution $\boldsymbol{\lambda}$.

Algorithm 2 Intra-domain Knowledge Pattern Representation

Input: D, S, T from particular domain
Output: $\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}$

1: **function** IntraDomain(D, S, T)
2: Train DSTM object: $\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}$ = DSTM(D, S, T)
3: **return** $\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}$
4: **end function**

Cross-domain Knowledge Pattern Representation

Besides exploring existing solutions in their own respective domains, researchers in many scientific communities seek to investigate new solutions by referring to knowledge patterns (such as e.g., methods, tools) from other synergistic domains.

In such scenarios for creation of cross-domain knowledge representations, our DSTM can be applied for user guidance. Common approaches for cross-domain recommendations involve using social information to build connections from other domains [62, 63], and also using transfer learning that makes use of auxiliary data [64]. Our idea for the DSTM application builds upon these existing approaches for finding similar topics from different synergistic domains to help with references to new methods, tools, and datasets from a cross-domain perspective.

The topics in our model are represented as multinomial distributions over a fixed vocabulary. The methods to measure difference or distance between two distributions could be Kullback-Leibler (KL) divergence or Maximum Mean Discrepancy (MMD) distance. The output value indicates the similarity between the two topics. In our work, instead of only providing a similarity score for users to compare and rank topics, we provide a novel visualization solution that embeds topics into a two-dimensional plane for easier exploration of similar topics in a lower dimensional space.

There have been prior algorithms implemented for topic embedding purposes. A few implementations directly treat multinomial distributions as a high dimensional dataset and then use PCA or t-SNE for mapping all topics onto a two-dimensional (2D) space. However, such an approach may not measure the real distance between any two distributions. Authors in [65] use Jensen-Shannon divergence to measure the distance between two distributions and then use multidimensional scaling (also known as Principal Coordinates Analysis) or t-SNE [66] to project the topics onto a 2D space. However, Jensen-Shannon divergence is a scale factor that outputs a value between 0 and 1. If the two distributions are similar, the value measured by JS divergence is very close, which makes it hard to differentiate in a 2D space. On the other hand, if the two distributions are not overlapped, the value will be always 1 no matter how different the two distributions are.

In our work, we have investigated an effective way of using t-SNE [66] for topic embedding via a novel MMD-t-SNE algorithm that maps topics onto a 2D

space. The original t-SNE uses Euclidean distance to measure distance between two high-dimensional datasets, which gives equally weights for each data entry. This approach is suitable for those datasets (e.g., images, bag-of-words), where each entry is a real value. However, for multinomial distributions, each entry is a probability that cannot be equally weighted.

So, we use Maximum Mean Discrepancy (MMD) [67] to measure the distance between distributions in a high-dimensional space. More specifically, we replace the Euclidean distance that is used in the original t-SNE algorithm for measuring pairwise similarities in the high-dimensional space with the MMD kernel function. Hence, the symmetric conditional probability between topic i and j is defined as,

$$p_{ij} = \frac{\exp\big(-\mathrm{MMD(i,\,j)}^2/2\sigma^2\big)}{\sum_{k\neq l}\exp\big(-\mathrm{MMD(k,\,l)}^2/2\sigma^2\big)} \tag{3.10}$$

where l denotes i or j, because of its symmetry. The Maximum Mean Discrepancy (MMD) distance between topic distribution ϕ_i and ϕ_j for multinomial distribution is suggested using the Pearson's Chi-square statistic test [67]. In order to make Chi-square distance symmetric, the final $\mathrm{MMD}(i,j)$ is calculated by the average of the two Chi-square χ^2_{ij} and χ^2_{ji} values as given by,

$$\mathrm{MMD}(i,j) = \frac{1}{2}\Big[\sum_v \frac{(\phi_{iv}-\phi_{jv})^2}{\phi_{jv}} + \sum_v \frac{(\phi_{jv}-\phi_{iv})^2}{\phi_{iv}}\Big] \tag{3.11}$$

Algorithm 3 shows the details for creation of cross-domain knowledge representations. With MMD-t-SNE algorithm, we can project any of the topics from any number of domains into the same 2D space in the form of a cross-domain knowledge pattern representation. For cross-domain recommendation, we simply need to explore topics from a 2D space to find closely related topics from other domains for referring research ideas, tools, as well as datasets. The algorithm to obtain cross-domain knowledge pattern fundamentally involves training multiple DSTM objects using datasets collected from multiple domains (for instance, using two domains a and b in our work) to generate topics, and embedding topics into a

low-dimensional topic representation $\hat{\boldsymbol{\phi}}$ using MMD-t-SNE.

Algorithm 3 Cross-domain Knowledge Pattern Representation

Input: Dataset from domain $a = \{D_a, S_a, T_a\}$, Dataset from domain $b = \{D_b, S_b, T_b\}$
Output: low-dimensional topics representation $\hat{\boldsymbol{\phi}}$

1: **function** CROSSDOMAIN(a, b)
2: Train DSTM object: $\boldsymbol{\phi}_a, \boldsymbol{\theta}_a, \boldsymbol{\lambda}_a$ = DSTM(D_a, S_a, T_a)
3: Train DSTM object: $\boldsymbol{\phi}_b, \boldsymbol{\theta}_b, \boldsymbol{\lambda}_b$ = DSTM(D_b, S_b, T_b)
4: Set $\boldsymbol{\phi}_c = \{\boldsymbol{\phi}_a, \boldsymbol{\phi}_b\}$
5: Embed topic: $\hat{\boldsymbol{\phi}}$ = MMD-t-SNE($\boldsymbol{\phi}_c$)
6: **return** $\hat{\boldsymbol{\phi}}$
7: **end function**

Trend Knowledge Pattern Representation

As detailed above, the intra-domain and cross-domain knowledge pattern representations in our DSTM applications allow discovery of broadly popular tools and/or datasets featured in all publications over the last ten years being considered. However, their use in a visually interactive manner within an interactive user interface requires a new kind of DSTM application. Our trend knowledge pattern representation application not only addresses this issue, but also ensures that the interactive knowledge discovery uses the most current information i.e., the information presented to the user tracks the change in trends or reveals the latest emerging trends in the relevant text corpus. For example, bioinformatics scientists in the past used to use the MATLAB SVM toolbox for pattern recognition; however, these days most scientists choose a deep learning framework (e.g., Keras, TensorFlow, PyTorch) to accomplish similar tasks. Our DSTM application can discover such trend knowledge patterns with an unsupervised learning approach, and can be adapted within effective visualizations in user interfaces.

Algorithm 4 details our novel algorithm to discover trend knowledge patterns for selecting tools or datasets. Our trend pattern analysis does not involve simply ranking tools or datasets by counting the number of times they are mentioned in papers for each year. Our explicit goal is to represent the trend knowledge pattern for tools or datasets based on trends of particular topics by year.

Algorithm 4 Trend Knowledge Pattern Representation

Input: D, S, T from particular domain
Output: $S_{ks}^{(i)}, T_{kt}^{(i)}$

function TREND(D, S, T)
    Train DSTM object: $\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}$ = DSTM(D, S, T)
    Initialize $T_{kt}^{(i)} = 0$; $S_{ks}^{(i)} = 0$; i denotes year;
    for each doc $d \in D$ **do**
        Set y to publication year of doc d;
        Initialize tool assignment $TA_t = 0$;
        Initialize dataset assignment $SA_s = 0$;
        Initialize topic assignment $KA_k = 0$;
        for each word $w_{dn} \in d$ **do**
            Sample a topic z_{dn} and tool x_{dn} with Eqn. 3.1;
            $TA_{x_{dn}} += 1$; $KA_{z_{dn}} += 1$;
            Sample a topic z_{dn} and dataset y_{dn} with Eqn. 3.2;
            $SA_{y_{dn}} += 1$; $KA_{z_{dn}} += 1$;
        end for
        Choose tool id $tid = argmax(TA_t)$;
        Choose dataset id $sid = argmax(SA_s)$;
        Choose topic id $kid = argmax(KA_k)$;
        $T_{kid,tid}^{(y)} += 1$;
        $S_{kid,sid}^{(y)} += 1$;
    end for
    return $S_{ks}^{(i)}, T_{kt}^{(i)}$
end function

The intuitive idea behind our trend knowledge pattern representation approach is to first calculate the occurrence of tools or datasets used for a particular topic in papers. Next, we group this information in terms of the year of publication. Following this, we can track the changes in trends or the evolution of the use of tools or datasets for a particular topic by year. However, we cannot train our model separately with publications grouped by year. This is because, if we train the model separately, the topics generated by the model may not be consistent, and will vary from year to year. This in turn makes it difficult to track a particular topic over several years. Therefore, we train our model once using all publications. Subsequently, we group papers by year and run our inference algorithm using this trained model as shown in Algorithm 4 to infer topics and relevant tools and/or datasets for each paper.

More specifically, in Algorithm 4, we first train a model once using all papers from an individual domain. Secondly, we initialize two 3D arrays $T_{kt}^{(i)}$ (with dimensions of the number of years, the number of topics, the number of tools) and $S_{ks}^{(i)}$ (with dimensions of number of years, the number of topics, the number of tools) to count samples generated by our inference algorithm. Thirdly, for each paper, we infer its topic and relevant tools or datasets based on our trained model, update relevant $T_{kt}^{(i)}$ or $S_{ks}^{(i)}$. We run these steps for several iterations to make sure the model converges. After completion of the inference procedure, we finally obtain sets of samples $T_{kt}^{(i)}, S_{ks}^{(i)}$ to estimate the trends within tools or datasets for a particular topic by year.

3.3 Evaluation

In this section, we first perform model selection to choose optimal parameters (such as number of topics, number of iterations) based on the two collections of datasets. Following this, we evaluate the generalization and information retrieval performance with state-of-the-art models such as Latent Dirichlet Alloca-

Table 3.2: Description of collected data for analysis from neuroscience and bioinformatics domain communities.

Category	Neuroscience	Bioinformatics
Papers	We have collected 16,721 latest neuroscience papers from four reputed journal archives: *Frontiers in Computational Neuroscience*, *Journal of Computational Neuroscience*, *Journal of Neuroscience*, and *Neuron* published from 2009 to 2019. This results in a vocabulary size of $V = 19,488$ unique words joint with bioinformatics text corpus and a total of $4,527,987$ word tokens.	We collected 10,681 bioinformatics papers from *Journal of BMC Bioinformatics, Journal of BMC Genomics, Genome Biology, Nucleic Acids Research, PLOS Computational Biology* published in the recent 10 years between 2009 to 2019. We extracted abstracts for each of the papers. This resulted in a vocabulary size of $V = 19,488$ unique words with neuroscience text corpus, and a total of $4,527,987$ word tokens.
Tools	We have collected the commonly used tools in neuroscience research activities including computation, simulations, databases and visualization, such as MATLAB, NEURON [68], PyNN [69], ModelDB [70], and new machine learning frameworks (e.g., TensorFlow, Keras) may be applied in recent neuroscience research. This results in a total of 189 tools.	We have collected 219 types of common used, tools which cover a variety of bioinformatics research works, including sequencing alignment tools (e.g., FASTA, BLAST), genome analysis tools (e.g., GATK, GenomeTools), quality control tools (e.g., FastQC, RSeQc), workflow management tools (e.g., Pegasus), and new machine learning frameworks (e.g., TensorFlow, Keras).
Datasets	Datasets described in neuroscience literature are usually recognized by cell types (i.e., pyramidal, interneuron) or brain regions (i.e., neocortex, retina). We collected the common dataset types in neuroscience experiments, which results in a total of 169 different types of datasets.	We have collected types of datasets in bioinformatics, including types of Ribonucleic acid (e.g., rRNA, tRNA, miRNA), types of sequencing (e.g., Chip-seq, Dap-seq, RNA-seq). This results in a total of 32 type datasets.

tion (LDA) [53] and Probabilistic Latent Semantic Analysis (PLSA) [71].

3.3.1 Dataset

We use three categories of datasets (papers, tools, and datasets) from two exemplar scientific domains viz., *neuroscience* and *bioinformatics* for understanding the synergistic relationships among research topics, tools, and datasets. Table 3.2 provides a more detailed description of the collected data for analysis, and the related data pre-processing steps performed on this collected data are as follows:

- **Papers**: We collected full papers from well-known journals in neuroscience and bioinformatics domain communities. We removed any words that occurred in less than 10 papers that are supposed to be highly infrequent words, and belonged to the list of "stop words" that are significantly frequent (e.g., "the", "a") in papers. Each paper was represented as a "bag of words" in our model.

- **Tools**: We collected the most commonly used tools of neuroscience and

bioinformatics domains separately in collaboration with domain experts. This list of tools covers a wide range of research efforts in computation, simulations, databases and visualization.

- **Datasets**: We collected common types of datasets used in experiments featured in the neuroscience and bioinformatics domains.

3.3.2 Model Selection

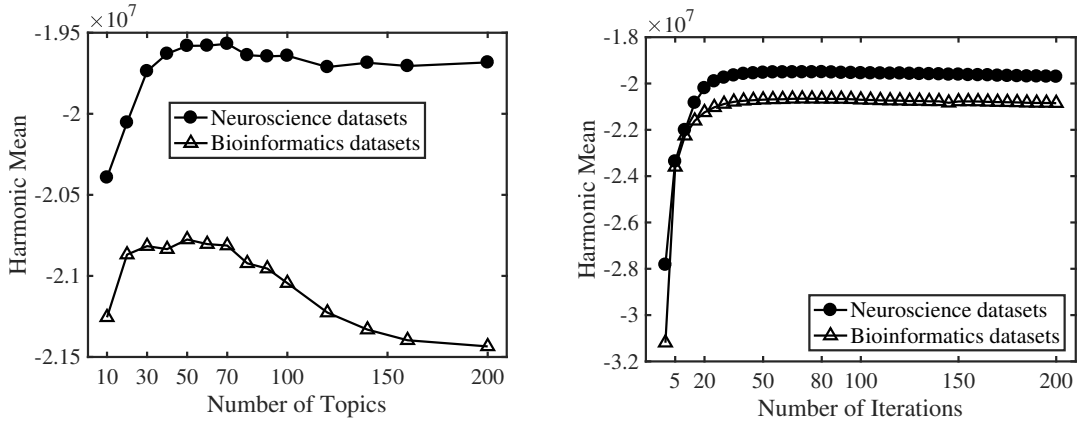
The model we described in Section 3.2.1 has four hyperparameters $\{\alpha, \beta, \gamma, \eta, T\}$. The $\{\alpha, \beta, \gamma\}$ are hyperparameters for symmetric Dirichlet prior. They may affect the number of topics: smaller values are supposed to find more topics from a text corpus; and larger values tend to collect a relatively smaller number of topics from a text corpus. Hence, for processing large volume of text corpus with wide ranging research problems, we choose smaller values of these hyperparameters, and vice versa. For simplifying the hyperparameters tuning in our model, we follow the suggestions from [60], keeping them constant: $\alpha = \gamma = 50/K, \beta = 0.1$, respectively. And the hyperparameter η adds prior probability for choosing tools or datasets for generating a single word. The hyperparameter is also kept fixed at $\eta = (3, 2)$. This because, we empirically suppose that tools in each paper have a higher chance to be selected to generate words. Hence, keeping the hyperparameters $\{\alpha, \beta, \gamma, \eta\}$ fixed, we decide the optimal number for topics and iterations. In addition, these optimal numbers are decided in terms of datasets, and our results are obtained based on the collected bioinformatics and neuroscience datasets.

Number of Topics

We can find the optimal number of topics T for the model based on a particular text corpus. Specifically, we need to compute the likelihood of words give a particular number of topics $P(\mathbf{w}|T)$ for all documents. This involves a step to

integrate out all possible topic assignments $\mathbf{z}$ for each word using $\sum_{\mathbf{z}} P(\mathbf{w}|\mathbf{z}, T)$. However, we can approximate the likelihood of words by using the harmonic mean of $P(\mathbf{w}|T)$ when $\mathbf{z}$ is sampled from a posterior distribution $P(\mathbf{z}|\mathbf{w})$ [60, 72]. Hence, we get,

$$P(\mathbf{w}|T) \approx \left\{ \frac{1}{m} \sum_{i=1}^{m} P(\mathbf{w}|z_i, T)^{-1} \right\}^{-1} \tag{3.12}$$



(a) Model selection for optimal number of topics

(b) Model selection for optimal number of iterations

Figure 3.2: DSTM model selection for choosing optimal number of topics and number of iterations using harmonic mean based on neuroscience and bioinformatics dataset collections.

As shown in Figure 3.2(a), the harmonic mean suggests that the optimal number of topics are nearly $K = 50$ for the bioinformatics dataset collection, and $K = 70$ for the neuroscience dataset collection.

Number of Iterations

In our DSTM, we use the Gibbs sampling as explained earlier in Section 3.2.2 for inferring latent variables in our model. The Gibbs sampling starts from a random initialization and runs over a few iterations to reach its equilibrium distribution. Hence, it is also important to estimate the optimal number of iterations for either achieving the best performance or saving computation time.

Using the optimal number of topics $K = 50$ and $K = 70$ for the bioinformatics

and neuroscience datasets respectively, we apply Equation 3.12 to compute the harmonic mean in terms of the number of iterations. As shown in Figure 3.2(b), our DSTM reaches its equilibrium distribution at iteration 50 with the bioinformatics dataset, and at iteration 80 with the neuroscience dataset.

3.3.3 Model Evaluation

In this section, we apply our DSTM on large collections of publications from two exemplar scientific domains: *neuroscience* and *bioinformatics*. Specifically, we evaluate the generalization performance of our model with the state-of-the-art models: Latent Dirichlet Allocation (LDA) [53] and Probabilistic Latent Semantic Analysis (PLSA) [71]. Next, we evaluate information retrieval performance and discuss issues by comparing with state-of-the-art models. Following this, we demonstrate the benefits of applying our model for choosing appropriate tools or datasets for the computational and data-intensive research problems discussed earlier in Section 3.2.3.

Generalization Performance Evaluation

The generalization performance is an important factor to evaluate how well a probabilistic model predicts a previously not observed sample based on the model parameters learned in the training stage. *Perplexity* is a standard metric that is widely used in probabilistic or text modeling to measure the predictive power of a model. A lower perplexity score indicates better generalization performance of held-out test datasets. Formally, the perplexity score of a test document d that contains words $\mathbf{w}_d$, and is conditioned on the known tools $\mathbf{t}_d$, datasets $\mathbf{s}_d$ of the document d and trained model, is defined as,

$$\text{perplexity}(\mathbf{w}_d|\mathbf{t}_d, \mathbf{s}_d, D_{train}) = \exp\Big\{ - \frac{\log \text{P}(\mathbf{w}_d|\mathbf{t}_d, \mathbf{s}_d, D_{train})}{N_d} \Big\} \tag{3.13}$$

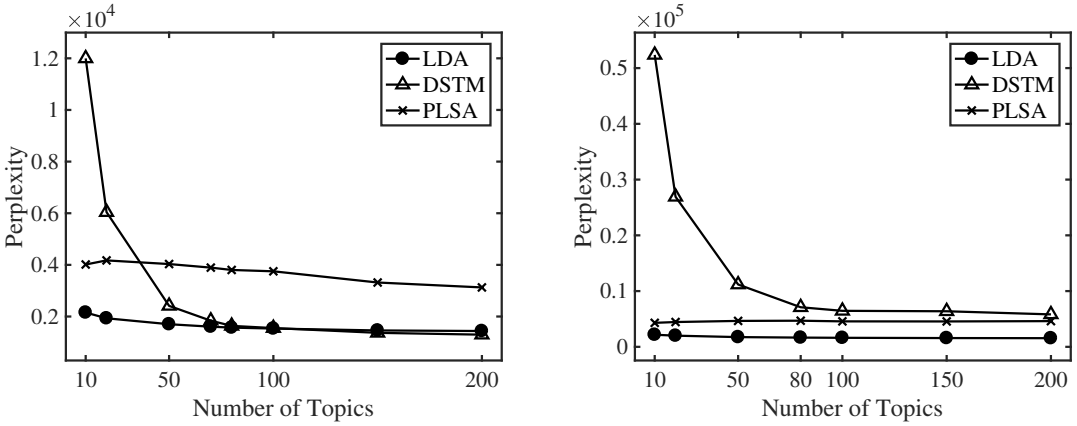
where $\mathrm{P}(\mathbf{w}_d|\mathbf{t}_d, \mathbf{s}_d, D_{train})$ is the probability of words $\mathbf{w}_d$ conditioned on known tools $\mathbf{t}_d$ or datasets $\mathbf{s}_d$ in the document d, and where the N_d is the number of words in the document d. To compute the overall perplexity score of all test documents D_{test}, we simply average the perplexity over test documents:

$$\text{perplexity}(D_{test}) = \frac{\sum_{d=1}^{D_{test}} \text{perplexity}(\mathbf{w}_d|\mathbf{t}_d, \mathbf{s}_d, D_{train})}{D_{test}} \tag{3.14}$$

The probability of words $\mathbf{w}_d$ in the document d with known tools $\mathbf{t}_d$ or datasets $\mathbf{s}_d$ can be obtained by integrating all latent variables,

$$\mathrm{P}(\mathbf{w}_d|\mathbf{t}_d, \mathbf{s}_d) = \prod_{n=1}^{N_d} \sum_{k=1}^{K} \Big[\frac{1}{\mathbf{t}_d} \sum_{t=1}^{T} \pi_d^{L_n=0} \phi_{k,w_n} \theta_{kt} + \frac{1}{\mathbf{s}_d} \sum_{s=1}^{S} \pi_d^{L_n=1} \phi_{k,w_n} \lambda_{ks} \Big] \tag{3.15}$$

Where the $\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\lambda}$ can be estimated through model training stage using Equations 3.4, 3.5, 3.6, respectively; also, the π_d needs to be sampled based on the new test documents d. This equation is similar to Equation 3.12, but it is computed with new test documents. Practically, we run the Gibbs sampling algorithm in Equations 3.8, 3.9 with a few iterations to get a stable estimation for each test document.



(a) Perplexity comparison on the neuroscience dataset

(b) Perplexity comparison on the bioinformatics dataset

Figure 3.3: Perplexity comparison with LDA, PLSA models on different dataset collections and for different number of topics.

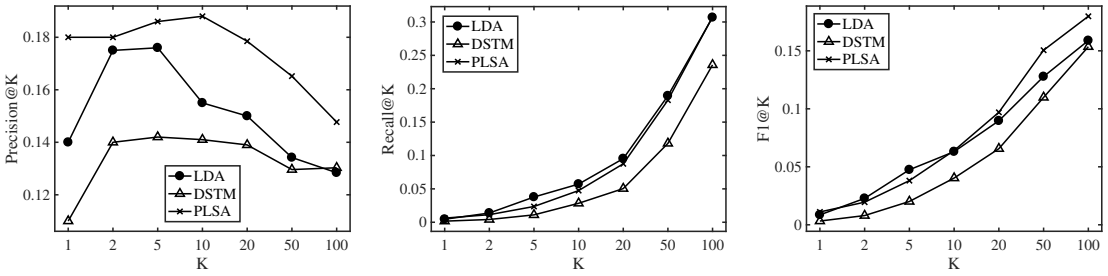
In our experiments, we compared the generalization performance of our DSTM

with the LDA and PLSA for both dataset collections (i.e., neuroscience and bioinformatics) shown in Table 3.2. In both cases and in both the models, we held out 20% of the same data for testing the generalization performance and used 80% of the same data for training.

Figure 3.3 shows that the perplexity scores of DSTM are significantly higher than the LDA, PLSA perplexity scores in both cases initially. We note that this has been caused by overfitting issues when the number of topics is relatively small. However, after increasing the number of topics, the DSTM quickly achieves similar generalization performance after topic ($K = 50$). Especially, the neuroscience dataset achieves better performance after topic ($K = 70$) as shown in Figure 3.3(a). Additionally, in both cases, the LDA and PLSA models' perplexity scores change slightly with the different number of topics, and the difference between the maximum and minimum scores is indistinguishable.

The above evaluation results provide insights of an interesting phenomenon where the LDA, PLSA models have overall better generalization performance for the most number of topics; whereas, our DSTM has better performance within a range of the particular number of topics (that in turn depends upon the diversity of the number of tools or datasets). The reason for this phenomenon is that the LDA, PLSA have a random generative process for producing each word. However, our DSTM generates words based on the occurrence of tools or datasets within each document to guide our model to find particular topics based on tools and datasets. From Figure 3.3, we can also find that the neuroscience dataset achieves better performance than the bioinformatics dataset. The reason is that the number of tools and datasets in the neuroscience dataset are larger than the same in the bioinformatics dataset. This finding also indicates that our model's performance can be improved by providing more data for model training. In essence, our DSTM has good generalization performance for finding highly specific topics within a domain, which is more suitable for domain scientists in finding particular resources (such as tools or datasets) as part of their knowledge discovery to solve research

problems at hand.



(a) Precision@K (b) Recall@K (c) F1@K

Figure 3.4: Information retrieval performance comparison of Precision, Recall, and F1 score at top-K recommendations.

Information Retrieval Performance Evaluation

In this section, we will evaluate the information retrieval performance with LDA and PLSA models. Information retrieval performance is used to evaluate - "how relevant is the retrieved document d that satisfies the user's query q". Mathematically, we need to estimate the conditional probability P($q|d$).

To construct query and retrieval datasets, we use the same datasets from bioinformatics domain as mentioned in Section 3.3.1, from which we randomly select 100 papers as the query sets and 500 papers as the retrieval sets. In order to test performance under different length of query sets, we construct query sets with different ratios of word length from abstracts. Given that we do not have the ground truth label for indicating the relevance level between each query set and each retrieval set, we arbitrarily estimate it by computing the cosine similarity score between the query and retrieval sets. We define that they are relevant when their cosine similarity score >= 0.2 (a mid-value in our data), otherwise, we conclude that they are not relevant. Based on this definition, we can evaluate information retrieval performance for each query based on various metrics (Precision@K, Recall@K, F1@K) for top-K recommendations as defined below,

$$\text{Precision@K} = \frac{|\{\text{relevant sets}\} \cap \{\text{retrieved sets}\}|}{|\{\text{retrieved sets}\}|}$$

$$\text{Recall@K} = \frac{|\{\text{relevant sets}\} \cap \{\text{retrieved sets}\}|}{|\{\text{relevant sets}\}|}$$
$$\text{F1@K} = \frac{2 \cdot \text{Precision@K} \cdot \text{Recall@K}}{\text{Precision@K} + \text{Recall@K}} \tag{3.16}$$

Figure 3.4 presents the information retrieval performance comparison with the state-of-the-art models. We can observe that the DSTM has similar performance at retrieving documents with the other two models (LDA, PLSA) among all three metrics (precision, recall, f1). The advantage of using DSTM over using LDA or PLSA is evident because - the user needs to provide fewer words input in their query (i.e., a smaller length of query set) to obtain comparable performance for retrieval of relevant documents.

We remark that it is not fair to compare retrieving documents performance of DSTM with other state-of-the-art models, given the fact that our DSTM is designed mainly for retrieving relevant tools or datasets. Our model does not have the distribution that indicates each document's topics proportion as in the case of the LDA, PLSA models. To estimate $\mathrm{P}(q|d)$, we approximately compute the probability of query words $\mathbf{w}_q$ given tools or datasets in retrieval documents $\mathrm{P}(\mathbf{w}_q|\mathbf{t}_d, \mathbf{s}_d)$, which can be computed using Equation 3.15. This can be understood as knowing whether the topics of tools or datasets are a fit for the query words or not. For example, the tool TensorFlow is not good fit for a query sentence that the user inputs as "Bayesian statistical inference". However, in LDA, PLSA models, the $\mathrm{P}(q|d)$ is estimated by directly considering topics of retrieval documents and query words $\mathbf{w}_q$ given the topics represented by $\sum_{k=1}^{K} P(\mathbf{w}_q|k)P(k|d)$. Hence, PLSA and LDA are more accurate at retrieving relevant documents. Another surprising finding is that PLSA achieves the best retrieval performance amongst these three models. The main reasons for this phenomenon is that we use cosine similarity to estimate the relevance between any two documents, and the cosine similarity tends to measure the word similarity score rather than the semantic similarity score. Further, this result also implicitly indicates that the DSTM and LDA models give higher consideration to semantic meaning rather than to the

words frequency.

3.4 Demonstration and Discussion

In this section, we demonstrate the knowledge patterns discovery enabled by our model for exemplar scientific domains. Our demonstrations use the DSTM constructed with appropriate parameters based on the discussion provided in Section 3.3.2. Our demonstration experiments use full datasets for both neuroscience and bioinformatics cases and feature three types of patterns relating to the applications outlined in Section 3.2.3 for helping researchers to choose appropriate tools or datasets for a particular research topic: (i) *Intra-domain Knowledge Pattern Demonstration* presents the overall patterns for choosing tools or dataset within a specific scientific domain; (ii) *Cross-domain Knowledge Pattern Demonstration* shows the patterns for helping researchers to investigate new approaches, tools, or datasets by referring to solutions from other synergistic domains; (iii) *Trend Knowledge Pattern Demonstration* presents the evolution of tools/datasets for particular topics over time, which helps researchers to make guided decisions based on the past successes and the latest emerging trends.

3.4.1 Intra-domain Knowledge Pattern Application

To obtain intra-domain knowledge patterns, we apply the Algorithm 2 to train a neuroscience model and a bioinformatics model with neuroscience and bioinformatics dataset collections, respectively.

Exemplar Neuroscience Domain Dataset Discussion

Table 3.3 illustrates 4 samples of topics from 70 topics learned by DSTM for the neuroscience dataset. These samples are extracted from a single chain at the 80^{th} iteration of the Gibbs sampler. Each sub-table in Table 3.3 shows the top 10

Table 3.3: 4 sample topics (out of 70 topics in total) extracted for the neuroscience publications from 2009 to 2019. Each topic is associated with 10 most likely words, 4 most likely tools and datasets that have the highest probability conditioned on that topic.

Topic 13	
Sensory & Anatomical Signals	
Word	**Prob.**
signals	.0320
stimulus	.0299
single	.0181
phase	.0174
movement	.0155
sensory	.0149
recording	.0147
response	.0140
timing	.0107
Tool	**Prob.**
MATLAB	.3550
EEGLAB	.1480
Klusters	.0693
opengl	.0476
Dataset	**Prob.**
vertical	.2555
horizontal	.2477
modulated	.1741
dorsal	.0637

Topic 18	
Circuit Analysis	
Word	**Prob.**
circuit	.0543
circuits	.0362
sensory	.0214
changes	.0187
input	.0147
circuitry	.0112
connections	.0108
stimuli	.0106
results	.0104
Tool	**Prob.**
MATLAB	.1147
SPSS	.1077
Topological	.0411
SPM	.0270
Dataset	**Prob.**
circuit	.6181
excitatory	.1576
cholinergic	.0390
inhibitory	.0296

Topic 27	
Structural Morphology	
Word	**Prob.**
axon	.0524
axons	.0464
axonal	.0407
cells	.0198
myelin	.0137
nerve	.0122
growth	.0109
cell	.0081
channels	.0068
Tool	**Prob.**
Neo	.2521
ansys	.0657
GENESIS	.0257
Caret	.0257
Dataset	**Prob.**
axon	.6568
myelinated	.1865
cone	.0350
ganglion	.0206

Topic 38	
Neuron Model	
Word	**Prob.**
neurons	.0328
channels	.0323
neuron	.0255
somatic	.0220
bursting	.0162
network	.0162
bursts	.0155
model	.0154
channel	.0153
Tool	**Prob.**
NEURON	.2520
XPPAut	.1037
ModelDB	.0762
GENESIS	.0547
Dataset	**Prob.**
somatic	.3782
bursting	.3252
excitatory	.0365
pyloric	.0318

Table 3.4: 4 sample topics (out of 50 topics in total) extracted for the bioinformatics publications from 2009 to 2019. Each topic is associated with 10 most likely words, 4 most likely tools and datasets that have the highest probability conditioned on that topic.

Topic 4	
Next-Generation Sequencing Analysis	
Word	**Prob.**
sequencing	.0484
reads	.0411
read	.0230
reference	.0174
sequence	.0141
mapping	.0141
assembly	.0141
short	.0141
high	.0141
Tool	**Prob.**
BWA	.2147
Bowtie	.1393
BLAST	.1382
Samtools	.1028
Dataset	**Prob.**
wgs	.3441
rnaseq	.1753
ENCODE	.0972
wes	.0646

Topic 10	
Protein Sequence & Structure Modeling	
Word	**Prob.**
structure	.0610
sequence	.0368
structures	.0353
secondary	.0307
structural	.0248
sequences	.0212
prediction	.0160
method	.0132
alignment	.0126
Tool	**Prob.**
rfam	.1486
psipred	.1215
BLAST	.1196
pfam	.0915
Dataset	**Prob.**
ENCODE	.2940
mirna	.1468
Hi-C	.1269
mrna	.0980

Topic 12	
Metabolic Analysis	
Word	**Prob.**
metabolic	.0818
flux	.0315
reactions	.0255
metabolites	.0204
network	.0152
analysis	.0146
different	.0109
fluxes	.0108
metabolomics	.0097
Tool	**Prob.**
COBRA	.3689
MATLAB	.3318
cplex	.1059
BLAST	.0625
Dataset	**Prob.**
metabolomics	.9074
mrna	.0274
wgs	.0208
mtdna	.0154

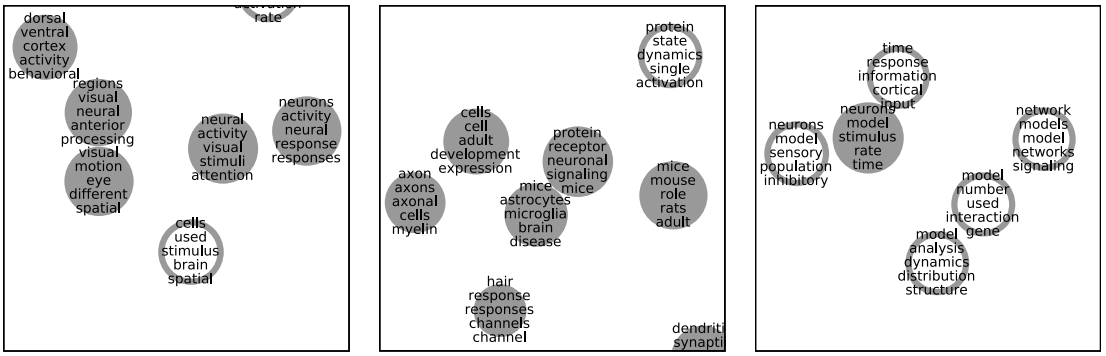
Topic 44	
Pattern Recognition	
Word	**Prob.**
learning	.0349
neural	.0316
features	.0220
image	.0195
images	.0178
based	.0124
performance	.0113
predict	.0111
training	.0106
Tool	**Prob.**
MATLAB	.7481
Keras	.0668
TensorFlow	.0437
Theano	.0423
Dataset	**Prob.**
mrna	.2412
tcga	.1637
rnaseq	.1162
dnaseseq	.0810

words that are most likely to be generated conditioned on that topic; the top 4 most likely tools to be used for the topic; the top 4 most likely types of datasets that have come from the topic. Note that we have provided the topic heading annotation to each topic for demonstration in the sub-tables. The first sample Topic 13 "*Sensory & Anatomical Signals*" involves intra-/extra-cellular recordings, LFP, EEG, etc. that use popular tools such as MATLAB, EEGLAB, Klusters with commonly used datasets such as vertical and horizontal. The second sample Topic 18 is related to "*Circuit Analysis*" that determines circuit connections using datasets that include excitatory/inhibitory type with possible neuromodulation, and standard tools for topological analysis, including SPM. The third sample Topic 27 "*Structural Morphology*" focuses on the cellular and circuit morphology including 3-D orientation and coursing of axons along tracts. The fourth sample Topic 38 describes research about the *"Neuron Model"* that features single neuron models focusing on the role of channels in cellular excitability and the variety of response patterns such as tonic spiking and bursting, using several new tools popular in the past decade including NEURON, GENESIS, and XPPAut that are hosted in databases such as ModelDB. These knowledge patterns were confirmed to be valid and helpful by neuroscience experts, which demonstrates that our DSTM effectively captures the salient domain knowledge patterns.

Exemplar Bioinformatics Domain Dataset Discussion

Table 3.4 shows 4 samples topics out of 50 topics for the bioinformatics dataset that are extracted from a single chain at the 50^{th} iteration of the Gibbs sampler. Each sub-table in Table 3.4 shows the top 10 words that are most likely to be generated based on that topic; the top 4 most likely tools to be used for the topic; the top 4 most likely types of datasets that are commonly used in or related the topic. Here also, we have provided the topic heading annotation to each topic for demonstration in the sub-tables. The first sample Topic 3.4 falls under the research topic of *"Next-Generation Sequencing Analysis"*, for which the

most popular tools being used are BWA, Bowtie, or BLAST, and the dataset types commonly used/related for this topic are *wgs* ("Whole Genome Shotgun"), *rnaseq* ("RNA sequencing"), etc. The second sample Topic 10 is related to *"Protein Sequence & Structure Modeling"*, for which popular tools such as *rfam*, *psipred* are accurately captured by our model, types of datasets such as *ENCODE*, *mirna* ("microRNA") are also commonly used for this topic. The third sample Topic 12 describes the research area of *"Metabolic Analysis"*, and tools such as COBRA, MATLAB, and datasets e.g., metabolomics are highly selected for this research. The fourth sample Topic 44 represents the *"Pattern Recognition"* research area in bioinformatics (such as gene expression, protein structure prediction). These knowledge patterns were also confirmed to be valid and helpful by bioinformatics experts, which demonstrates that our DSTM once again effectively captures the salient domain knowledge patterns.



(a) cross-domain topic for understanding the relationship between vision and brain

(b) cross-domain topic for understanding the role of protein in brain

(c) cross-domain topic for neuron modeling or neuron simulation

Figure 3.5: Exemplars of cross-domain knowledge patterns recognized by our model. Each node denotes a topic learned by our model that is either from neuroscience domain ("shaded" node) or bioinformatics domain ("unshaded" node); and each topic is annotated by the top 5 most likely words.

3.4.2 Cross-domain Knowledge Pattern Application

To discover cross-domain knowledge patterns, we apply the Algorithm 3 to train a neuroscience model and a bioinformatics model with relevant dataset collections

Table 3.5: Description of the exemplars of cross-domain knowledge patterns shown in Figure 3.5

		Tools For Sharing		Datasets For Sharing	
Figure No.	**Description**	**Neuro**	**Bio**	**Neuro**	**Bio**
Figure 3.5(a)	This figure presents the cross-domain topics about understanding the relationship between vision and brain, which are a mature area in neuroscience, but a rare area in bioinformatics	MATLAB, Fieldtrip, EEGLAB, Talairach, FreeSurfer	MATLAB	horizontal, vertical, retina, modulated, excitatory, inhibitory, circuit	mrna, rnaseq
Figure 3.5(b)	This figure shows the cross-domain topics about discovering the effects of protein structures on brain stimulation. Neuroscience researchers attempt to understand this problem from different approaches (such as signaling, gene expression, cell regions or types). Bioinformatics researchers also try to understand this problem from the signal activation aspect.	graphpad, MATLAB	MATLAB, Gromacs	dendritic, axon, tyrosine, gabaergic, microglia, astrocytes	mrna, mirna
Figure 3.5(c)	This figure illustrates the cross-domain topics the cross-domain topics about neuron stimulation	Helmholtz, klustakwik	MATLAB, Taverna, DARIO, glasso, Mikado	vertical, excitatory, horizontal	mirna, mtdna, metabolomics, rnaseq, tcga, ENCODE

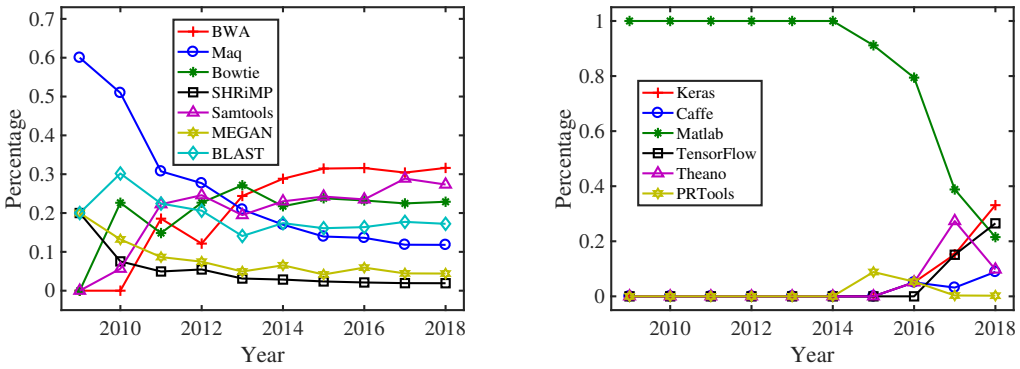
mentioned in Table 3.2, respectively. We embed the topics in a 2D space to generate a low dimensional topics representation vector $\hat{\boldsymbol{\phi}}$. Subsequently, we can easily plot this information into a 2D space and observe the cross-domain knowledge patterns. Due to the plot size limitations, we note that we had to crop some exemplars of cross-domain patterns from the original embedding plot for demonstration.

As shown in Figure 3.5, we list three cross-domain patterns extracted from the original plot. In each sub-figure, the node denotes a topic learned by our model that is either from the neuroscience domain ("shaded" node) or from the bioinformatics domain ("unshaded" node). Figure 3.5(a) illustrates that researchers from both neuroscience and bioinformatics domains aim to understand the relationship between vision and brain. We can see from the plot that these type of research efforts are very mature in the neuroscience domain but rare in the bioinformatics domain. Neuroscience researchers try to understand this problem from different

aspects (e.g., responses, attention, motion, spatial and different cell regions) compared with the single aspect perspective in the bioinformatics domain. Hence, bioinformatics researchers can learn notably from neuroscience domain about synergistic research ideas, tools, and datasets.

We also observe from Figure 3.5(b) that there is an overlapping research topic in two domains for understanding the effects of protein structure on brain stimulation. Neuroscience researchers attempt to understand this problem using approaches such as signaling, gene expression, cell regions or types. Bioinformatics researchers also try to understand this problem from the aspect of signal activation. Hence, cross-domain knowledge recommendation can be fostered by mutually sharing resources and findings across the two synergistic domains. Specifically, cross-domain knowledge sharing can notably improve the understanding of the neuron simulation shown in Figure 3.5(c).

3.4.3 Trend Knowledge Pattern Application

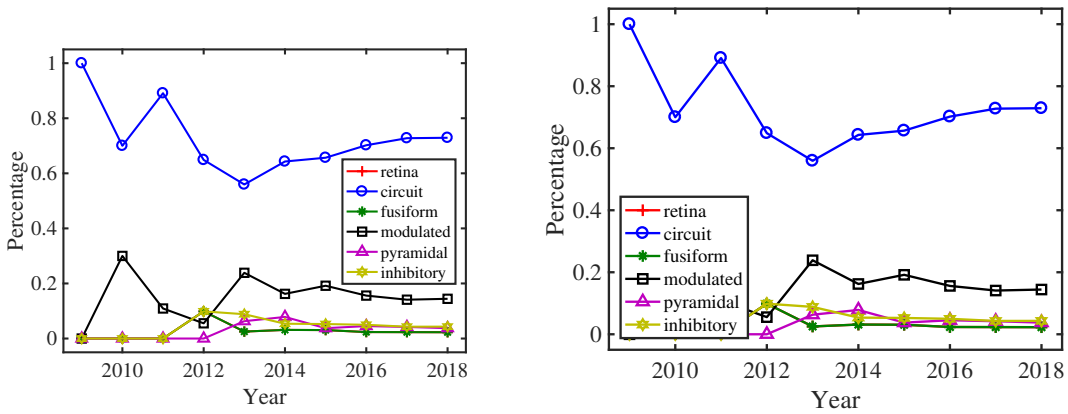


(a) Tools trend pattern on Topic 4 (*Next-Generation Sequencing Analysis*)

(b) Tools trend pattern on Topic 44 (*Pattern Recognition*)

Figure 3.6: Exemplars of trend knowledge patterns captured by DSTM for the bioinformatics domain.

We apply Algorithm 4 to obtain trend the knowledge pattern for each topic. Fig. 3.6 illustrates two examples of trend knowledge patterns in the bioinformatics domain. In Fig. 3.6(a), we can clearly see that the tools trend pattern for Topic 4



(a) Datasets trend pattern on Topic 18 (*Circuit Analysis*)

(b) Tools trend pattern on Topic 38 (*Neuron Model*)

Figure 3.7: Exemplars of trend knowledge patterns captured by DSTM for the neuroscience domain.

(*"Next-Generation Sequencing Analysis"*): the Maq [73] used to be the most popular tool for sequence analysis. However, in recent years, tools such as BWA [74], Samtools [75], and Bowtie [76] have become increasingly popular. Fig. 3.6(b) clearly captures the history of deep learning research in bioinformatics for pattern recognition. As we can see from the plot, MATLAB almost dominated this area before 2015, for which SVM toolboxes in MATLAB were commonly used. From 2015 to 2016, scientists started to use deep learning methods for pattern recognition, for which tools such as Keras, Caffe [77], Theano were used for deep learning research. In 2016, we can see observe that TensorFlow started to be used for deep learning, which can be also be validated by the fact that Google released the first TensorFlow version on November 9, 2015. After 2017, Theano has become less popular, because MILA stopped supporting Theano. Consequently, TensorFlow and Keras have emerged as the most popular tools in bioinformatics for deep learning; and MATLAB lost its dominating position in pattern recognition.

Fig. 3.7 illustrates two exemplars of trends in knowledge patterns in neuroscience. In Fig. 3.7(a), we can observe trends as captured by our model in the reports of usage of datasets related to circuit analysis research i.e., circuit analysis is the most popular in usage of datasets in the last ten years. Other datasets

such as those related to modulation, inhibition, and pyramidal cells are also used by researchers with a slightly increasing trend that has now saturated in recent times. The second neuroscience example shows the trend patterns in the usage of neuron models in Fig. 3.7(b). Among the numerous neuronal modeling packages, NEURON seems to be the most popular in the last ten years. Other popular packages are XPPAut and GENESIS as captured by our model. The same plot also shows that the model database repositories for code using these tools are the ModelDB and NeuronDB.

Another major benefit in using trend knowledge patterns can be seen in cases where we can combine it with the intra-domain knowledge pattern discussed in Section 3.4.1 pertaining to the results in Table 3.4. We can see that MATLAB is the traditional tool that is highly used in this area over the last last ten years as captured by our model. Our model also captures trends in the use of new deep learning tools such as Keras, TensorFlow that are increasing at a dramatic pace in the recent years, but with lower probability in the past years comparison with MATLAB. This intra-domain knowledge pattern result might mislead researchers to select MATLAB for pattern recognition research. However, combining the trend knowledge pattern with the intra-domain knowledge pattern, researchers can have better guidance to select the more latest trending tools to suit their research problems.

3.5 Summary

In this chapter, we presented a novel "domain-specific topic model" (DSTM) for discovering latent knowledge patterns among research topics, tools and datasets for computational and data-intensive scientific communities. Our DSTM is a generative model whose design incorporates as little, or any amount of domain knowledge while exploring highly-specific topic patterns within a given domain. DSTM can shorten the time to knowledge discovery and help scientists/researchers to adapt

prior domain knowledge when pursuing multi-disciplinary investigations. Similar to the popular LDA method in prior works, DSTM uses a completely randomly generative process to generate words based on reference tools or datasets. Using large collections of two different text corpus from neuroscience and bioinformatics domains (includes more than 25,000 papers over the ten years from reputed journal archives), our evaluation experiments with quantitative perplexity scores and qualitative domain expert feedback show that our DSTM has better generalization performance for revealing highly specific latent topics within a domain. We have also shown that the information retrieval performance results for DSTM is comparable to other state-of-the-art methods such as LDA and PLSA, when users provide a small length of query set. We proposed three exemplar applications of DSTM with concrete algorithms for *Intra-domain*, *Cross-domain*, and *Trend* knowledge patterns discovery from large datasets obtained from scientific publication archives. We showed how DSTM can be relevant for researchers seeking to make intelligent decisions using knowledge discovery for developing solutions to multi-disciplinary research problems using state-of-the-art best practices for tools and datasets.

Possible future directions for this work include building visualization/drill-down interfaces to browse the knowledge patterns among research topics, tools and datasets. Such interfaces can further foster an efficient query to obtain appropriate resources (e.g., tools, and datasets) for cutting-edge research investigations in many other disciplines (e.g., material science, business analytics). Our DSTM extensions can be developed and integrated within a recommendation system to recommend appropriate distributed computing resource configurations to domain scientists based on their workflow requirements. Further, it can be extended to be used within conversational agents (i.e., chatbots) to help users of science gateways to discover latent knowledge patterns among research topics, tools and datasets in an interactive manner [61].

Chapter 4

Scholar Recommender using a Deep Generative Model

Bold scientific research tasks today need multi-disciplinary knowledge and interdisciplinary collaborations that require finding scholars from a particular domain with relevant knowledge. Given the variety of scholars and diversity of research tasks, finding the appropriate scholar is a critically important and challenging problem for scientific communities. Prior approaches to identify scholars use supervised learning with fixed or high-level research interest tags. Such approaches make it hard to recognize scholars with specific interests, or track their changes in research interests. Hence, there is a need to investigate suitable methods to quantify scholars' expertise knowledge for matching research tasks. In this section, we propose a novel model viz., "ScholarFinder" that uses contextual information (abstracts or publications) for embedding a scholars' knowledge with a deep generative model in an unsupervised learning manner. Subsequently, with the pre-trained scholars' knowledge embeddings, we can perform machine learning tasks such as classification, visualization or checking whether a scholar is suitable for particular research tasks or not. Based on our pre-trained techniques, we also provide a novel negative sampling method to overcome the issues of missing negative samples. Using a "follow-the-money" strategy, we apply our model to a large collection of NSF

(National Science Foundation) grant awards dataset collected over the last twenty years that contains more than 20,000 award records (with project abstract and names), and 15,074 scholars who received grants. We evaluate different deep learning models to see how to use pre-trained knowledge embedding for achieving optimal performance, and how our negative sampling method improves the performance. We also compare our model with state-of-the-art baseline models (e.g., XGBoost, DNN), and our results show that the ScholarFinder model outperforms those models in terms of precision, recall, F1-score, Accuracy metrics.

4.1 Background and Related Work

Deep learning shows its best-in-class performance on problems that significantly outperforms other solutions in multiple domains such as speech, language, vision, which trains a deep neural network using the back-propagation algorithm with a large amount of dataset. It can effectively reduce the need for handcrafting feature engineering, one of the most time-consuming parts of machine learning practice.

Representation learning is also an important area in deep learning that learns the latent representation of high-dimensional data to extract useful features, such as basic shapes with image dataset, and semantic meaning with text dataset. Those features can be effectively used to perform classification, detection, as well as recommendation. In earlier 1986, Hinton [78] presented a distributed representations learning of concepts with a simple neural network to learn family relationships. For a while, RBM and Autoencoder [79] were the popular deep generative models to model high-dimensional data for extracting features. Recently, VAE [80] and GAN [81] have shown impressive performance in generating data, which have made them become the most popular deep generative models. Autoencoder and VAE learn the latent representation (or embedding) through reconstructing input data, and GAN learns the latent representation by playing games through generator and discriminator. In our work, we choose VAE to learn expertise embedding,

because our datasets are bag-of-words, which are not continuous. GAN is more suitable for use when given continuous datasets. In comparison with autoencoder, VAE can achieve better generalization performance by replacing latent variables with distributions instead of discrete value.

Recommendation system also leverages a deep learning technique to achieve good performance. Salakhutdinov *et al.* [82] proposed a restricted Boltzmann machine with one visible layer and one hidden layer to identify those users with similar interests. Kim *et al.* [83] combined convolutional neural network (CNN) and probabilistic matrix factorization (PMF) to capture contextual information of documents for improving performance. He *et al.* [25] also fused matrix factorization (MF) and Multi-Layer Perceptron (MLP) to achieve better recommendation performance. Our approach involves using a deep generative model for knowledge embedding that obtains embeddings through modeling of given public datasets in order to capture semantic meaning of data.

Embedding is a popular technique that is widely used in dimension reduction [79], and for finding good representations as well as visualizations [24]. Normally, we can use matrix factorization [22] and neural network based embedding [23] to learn embeddings. However, the former method performs a matrix decomposition manner that needs to recompute when a new user or item is added into the rating matrix. Neural network based embedding have become popular, because of its flexibility. When a new user or item is added, it only needs to update the weights using the stochastic gradient descent (SGD).

Embedding is also useful in a recommendation system for solving the issue of sparsity. Hence, various embedding methods are applied to a recommendation system: Grbovic *et at.* [84] propose a skip-gram based model "prod2vec" to learn product embeddings for product-to-product predictions, and user embeddings for user-to-products predictions. Li *et al.* [85] proposed a mixture embedding method for questions classification, which combines topic embeddings, word embeddings, and entity embeddings. Ramanath *et al.* [86] combined embedding and semantic

representations for talent search at LinkedIn. Miao *et al.* [87] proposed a method that uses variational autoencoder (VAE) to capture the latent representation of the documents.

Similar to word2vec [24], we would like to embed scholar's knowledge and use pre-trained knowledge embedding to perform any kinds of tasks (such as matching and classification). We assume that a scholar's expertise knowledge is similar to word embedding because its semantic meaning of knowledge, which can be trained based their publications, and used for later prediction.

4.2 ScholarFinder Methodology

In this section, we present the methodology of our ScholarFinder model, where we mainly focus on three aspects:

- Learning good representation of scholars' knowledge using an embedding technique based on their publications.
- Demonstrating a novel negative sampling scheme for solving the issue of unbalanced labels in datasets.
- Using pre-trained knowledge embeddings to predict whether a scholar is suitable for a proposed task.

Table 4.1 lists all the notations used in this section and in the rest of the paper.

4.2.1 Knowledge Map

Our knowledge embedding fully relies on scholars' publications. Scholars' publications are represented by a bag-of-words with a fixed size of the vocabulary C. These vocabulary are generated by using keywords from scientific topics taken from the ScienceDirect website [88] and removing those keywords that do not appear in their publications. In this way, we can recognize a scholar's expertise

Table 4.1: Notations listing for ready reference of the terms used in the methodology.

Notation	Description
M	Number of scholars
N	Number of tasks
C	Size of the vocabulary
K	Dimension of embeddings
L	Number of layers
$x_i \in \mathbb{R}^{1\times C}$	Bag-of-words representation of the i-th scholar's publication
$\boldsymbol{X} \in \mathbb{R}^{M\times C}$	Bag-of-words representation matrix for all M scholars' publications
$y_i \in \mathbb{R}^{1\times C}$	Bag-of-words representation of the i-th proposed task abstracts
$\boldsymbol{Y} \in \mathbb{R}^{M\times C}$	Bag-of-words representation matrix for all Y proposed task abstracts
$u_i \in \mathbb{R}^{1\times K}$	Knowledge embedding for scholar i
$\boldsymbol{U} \in \mathbb{R}^{M\times K}$	Knowledge embedding for all M scholars
$v_j \in \mathbb{R}^{1\times K}$	Proposed task embedding for task j
$\boldsymbol{V} \in \mathbb{R}^{N\times K}$	Proposed task embedding for all N task
$W_l, \boldsymbol{W}$	Weights for hidden layer l, and $l \in \{1, 2, ..., L\} \cup \{\mu, \sigma\}$
$b_l, \boldsymbol{b}$	Biases for hidden layer l, and $l \in \{1, 2, ..., L\} \cup \{\mu, \sigma\}$

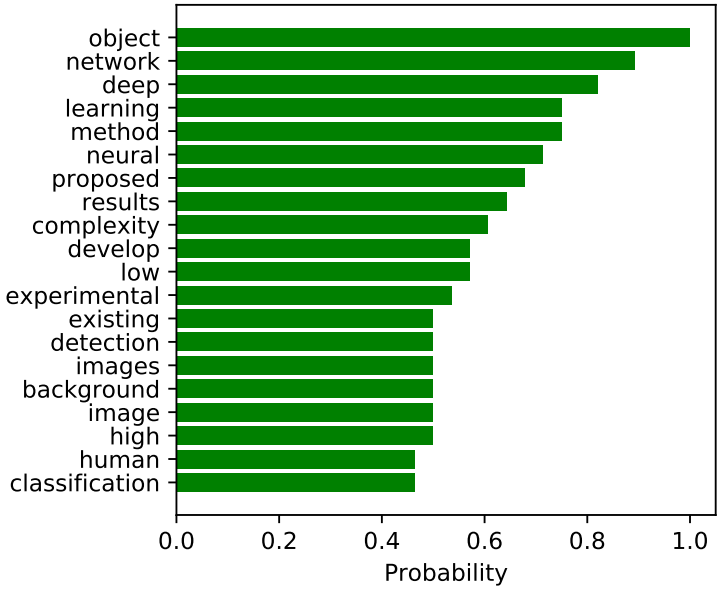


Figure 4.1: Vector representation of bag-of-words based on a scholar's publications with top 20 frequent words shown above; words counts are normalized between 0 and 1.

knowledge. As shown in Fig. 4.1, we can easily recognize that this particular scholar has expertise knowledge in deep learning from the most frequent words occurrence in his publications, which are normalized between 0 and 1.

Suppose we have M scholars, each scholar's publication in bag-of-words is denoted as x_i, which is a vector with the size of vocabulary C dimension. Then,

all scholars' publications are represented by $\boldsymbol{X} \in \mathbb{R}^{M \times C}$.

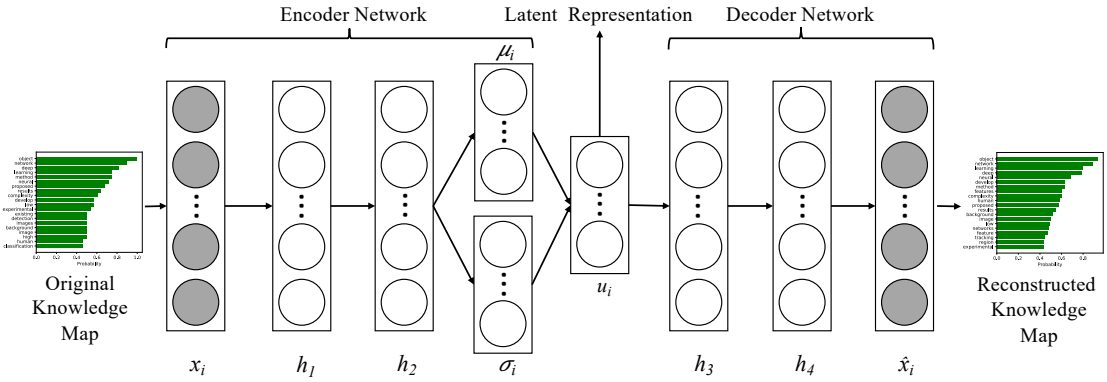


Figure 4.2: Architecture of Knowledge embedding using variational autoencoder, which has two hidden layers in encoder network and two hidden layers in decoder network respectively.

4.2.2 Knowledge Embedding using Variational Autoencoder

To learn good knowledge embedding of a scholar, we adopt the variational autoencoder (VAE) [80] to find the latent representation for a scholar's publication (or knowledge map), which is used for knowledge embedding.

Autoencoder [79] is typically a neural network that is trained to reconstruct input data through encoder/decoder networks in an unsupervised manner. In addition, a latent representation (coder) is also learned by reconstructing input data during the training process. The VAE model is similar to the Autoencoder, but instead of using discrete variables for latent representation in Autoencoder, VAE uses a distribution (such as, Gaussian distribution or other differentiable distribution) to present it. This in turn can help achieve better generalization performance. In order to infer the latent variables, VAE uses the Stochastic Gradient Variational Bayes (SGVB) wit reparameterization trick [80].

In Fig. 4.2, we present our VAE architecture for learning knowledge embedding. The goal of the VAE model is to learn knowledge embedding u_i for each scholar based on the input of his/her publication x_i, and all scholars shared Weights.

We define that the latent representation u_i is drawn from the Gaussian distri-

bution,

$$u_i \sim \mathcal{N}\left(\mu_i, \sigma_i^2 \mathbf{I}\right) \tag{4.1}$$

With SGVB algorithm, the u_i can be approximated with,

$$u_i = \mu_i + \sigma_i \odot \epsilon \quad \text{and} \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{4.2}$$

And u_i, μ_i, σ_i are hidden vectors with K dimension in VAE model shown in Fig. 4.2 that are computed by a regular feed-forward neural network. The K is also a dimension of embedding, which needs to be defined. In our model, we use $K = 50$ for learning knowledge embedding. We use two hidden layers in our encoder network and two hidden layers in our decoder network with 500 hundred neurons in all encoder and decoder layers. Except the layers for μ_i, σ_i, and u_i, the activation functions for each layer is ReLU [89]. Thus, the l-th hidden layer h_l can be computed by,

$$h_l = \text{ReLu}(W_l^T h_{l-1} + b_l) \tag{4.3}$$

And, the μ, σ for i-th scholar can be calculated by,

$$\mu_i = W_\mu^T h_2 + b_\mu \tag{4.4}$$

$$\sigma_i = W_\sigma^T h_2 + b_\sigma \tag{4.5}$$

To learn weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ for obtaining knowledge embedding u_i, we need to solve the optimization problem by minimizing the reconstruction loss and KL-divergence loss. Hence, the VAE loss for i-th example is defined as,

$$\mathcal{L}(\boldsymbol{W}, x_i) \simeq -\frac{1}{2}\sum_{k=1}^{K}\left(1 + \log((\sigma_i^k)^2) - (\mu_i^k)^2 - (\sigma_i^k)^2\right) - \frac{1}{L}\log P\left(x_i | u_i, \boldsymbol{W}\right) \tag{4.6}$$

This optimization problem can be solved using the stochastic gradient descent

(SGD) or any of the other optimizers. In our work, we use the Adam optimizer [90].

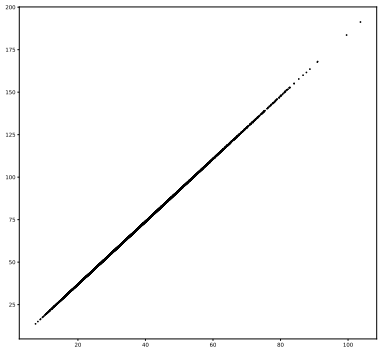
After the training completion of our VAE model, we obtain knowledge embedding u_i for each scholar i. Then, we can use the scholars' pre-trained knowledge embedding to perform further proposed tasks. Our subsequent task is to check whether a scholar is suitable for a proposed NSF fundable research task. We remark that NSF funding tasks are also described in the text format. Similarly, we use our VAE model to generate NSF funding task embeddings $\boldsymbol{V}$. The VAE architecture for embedding NSF funding is the same as the architecture for scholars' knowledge embedding, but we train and obtain embeddings separately.

4.2.3 Negative Sampling

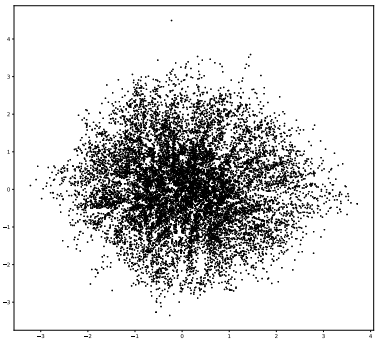
The negative sample issue is common in most machine learning tasks. This is because, it is easier to obtain a positive sample than a negative sample in the real world. For example, in job matching tasks, we can easily get datasets showing a person who got a job offer, but it is hard to obtain datasets showing persons who rejected offers or failed in the interviews. We faced a similar missing negative samples issue in our dataset that only has positive samples. More specifically, in the NSF grant award datasets, we only have records that show that the scholars who got grants from NSF, but there are no records showing the scholars who failed to apply for a certain NSF grant. Only positive examples training is not effective for a machine learning model that then always produces positive predictions.

Based on the above discussion, we need to generate negative samples for each scholar to train our model. Common approaches such as word2vec [24] use a random sampling scheme to randomly pick up a select number of dataset samples as negative samples. This approach works well if the dataset is very large and has a high dimensional manifold that creates low correlations among samples. For instance, in the word2vec scenario, it is very likely to randomly select a sample that has a completely different semantic meaning compared to the target dataset.

However, in our case, the manifold dimensions are low owing to the fact that



(a) Embedding 15,074 scholars in 2D space using AutoEncoder model discussed in Section 4.2.2 with $K = 2$ embedding dimension.



(b) Embedding 15,074 scholars in 2D space using VAE model discussed in Section 4.2.2 with $K = 2$ embedding dimension.

Figure 4.3: Comparison different deep embedding techniques for knowledge embedding of scholars.

we only have hundreds of research fields with high correlation. As shown in Fig. 4.3(b), we applied our VAE model to embed all 15,074 scholars into a 2D space. It clearly shows that the correlations among scholars are very high in terms of scientific areas such as e.g., "computer system" is highly correlated with "computer networking" or "cloud computing"; "bioinformatics" is highly correlated with "data mining" or "machine learning".

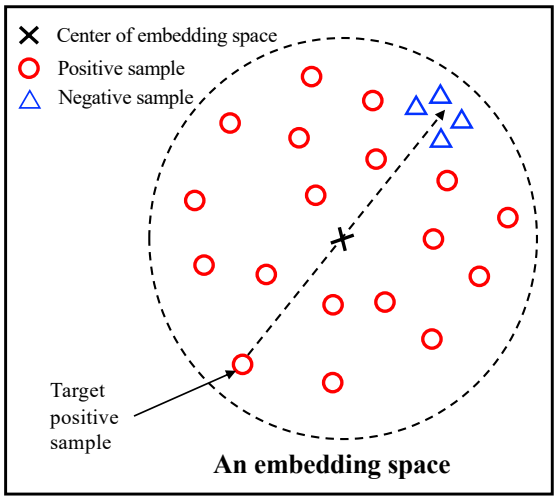


Figure 4.4: Illustration of a negative sampling scheme in our ScholarFinder model.

The goal in our problem is to sample adequate negative samples for each scholar. In the random sampling scheme, for each scholar, we randomly sample

the same amount of negative samples with positive samples from whole datasets for indicating those NSF grants that are not suitable for that scholar. Given that the data correlation is high, this approach results in cases with a higher chance to sample an NSF grant that is similar or has some connection with a positive sample. This in turn makes the model training less effective for accurate predictions of scholar recommendations.

Hence, based on our pre-trained embedding technique, we propose a novel negative sampling scheme to solve this issue. Instead of sampling from the real dataset, we directly sample from the embedding space. As shown in Fig. 4.4, there is an embedding of positive sample u_i for a scholar. The good negative sample embedding should be far from a positive example, or in other directions of our embedding space. We can easily compute the negative embedding by first computing the center of embedding space S, and then subtracting the positive embedding, which is given by,

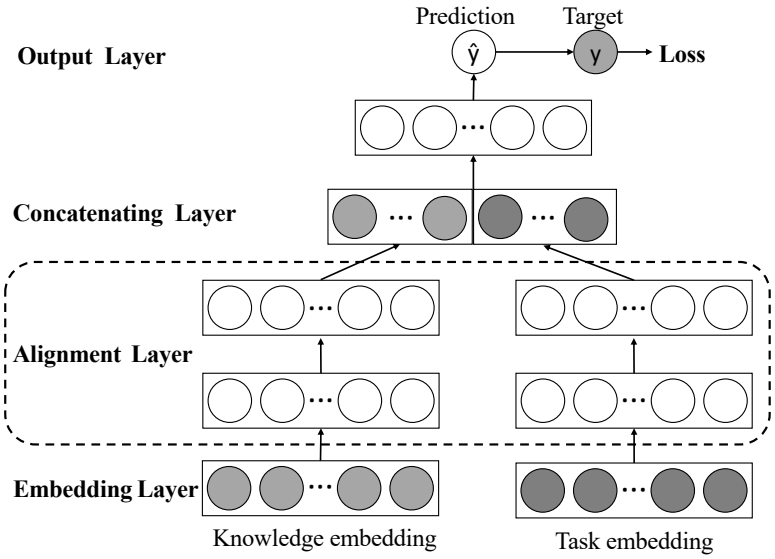
$$2S - v_i + \epsilon \tag{4.7}$$

where the ϵ is random noise.

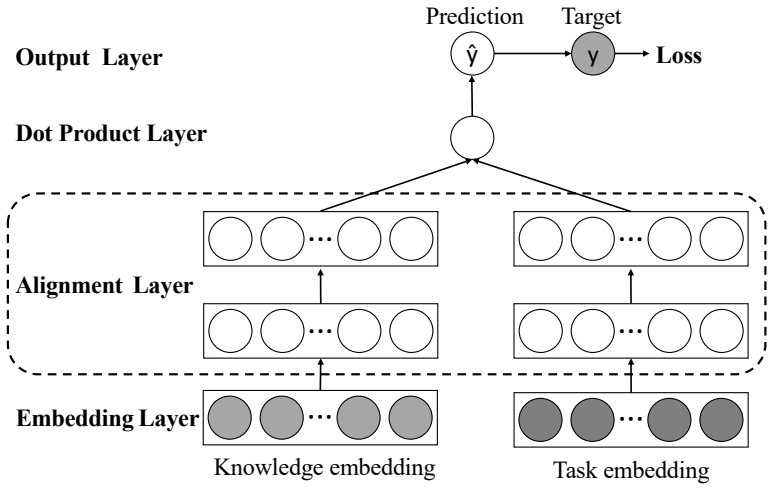
This is allowed, because we pre-train the embeddings (knowledge embedding and tasks embeddings) separately in an unsupervised learning manner, and perform subsequent prediction tasks using the pre-trained embeddings. This approach cannot be applied to those methods when the tasks and knowledge embeddings are trained jointly. This is because, the knowledge and task embeddings are in the same space. Task embeddings will have similar information in comparison with the knowledge embeddings. Consequently, after training the embedding, we compute the center of embedding space, which is equal to the mean value of the embedding vectors and then use the Equation 4.7 to obtain negative embedding samples for performing prediction tasks.

4.2.4 Prediction with Pre-trained Knowledge Embedding

As we discussed in Section 4.2.2, we use the VAE model to get scholars' embeddings $\boldsymbol{U}$ and proposed tasks embedding $\boldsymbol{V}$ separately. In this section, we will discuss our method to build the prediction task using the pre-trained embeddings. For any scholar i and proposed task j, our goal is to predict whether the scholar i is suitable for the proposed task j based on their embeddings u_i, v_j, so that the two embeddings are in the same latent space.



(a) Concatenating layer deep learning model architecture.



(b) Dot product layer deep learning model architecture.

Figure 4.5: Illustration showing the use of pre-trained knowledge embedding for proposed tasks prediction.

Intuitively, this problem looks easy to solve, which just requires computing the dot product of these two vectors u_i, v_j. This method can be used when u_i and v_j embeddings are jointly trained. In our case, however, the two pre-trained embeddings u_i and v_j are learned separately, which are not in the same embedding space although they have the same latent dimension K. Hence, the dot product computation will not work in this scenario.

To solve this problem, we propose two deep learning models shown in Fig. 4.5. The major difference between these two models is that: the first one uses "Concatenating Layer", and the other uses "Dot Product Layer"; the other parts are the same. First, we have the "Alignment Layer" on the top of the "Embedding Layer" as shown in Fig. 4.5. The purpose of the "Alignment Layer" in our model is to align the two vector spaces u_i and v_j before performing concatenation i.e., a dot product on them. The "Alignment Layer" is constructed by two or three fully connected hidden layers with the ReLU activation function and 10% dropout [91]. If two vectors are matched, the "Alignment Layer" will align them into the same space, in which the "Output Layer" will output 1, and vice versa. After "Alignment Layer", the model(a) will concatenate them and connect with another fully connected layer with output results of 0 or 1. Alternately, the model(b) will directly perform dot product to output the results. To learn the model parameters, we need to minimize cross-entropy loss between prediction value $\hat{y}_{ij}$ and target value y_{ij} for all pairs of (u_i, v_j),

$$\mathcal{L} = \sum_{i=1}^{M} \sum_{j=1}^{N} \Big[- y_{ij} \log \hat{y}_{ij} - (1 - y_{ij}) \log(1 - \hat{y}_{ij}) \Big] \tag{4.8}$$

In summary, our ScholarFinder model involves three stages for finding suitable scholars for particular tasks. The first two stages involve learning embeddings: (i) learning good knowledge embeddings for scholars, as shown in Algorithm 5; (ii) learning good task embeddings for scholars as shown in Algorithm 6. In the stage (iii), predictions are performed to check whether a particular scholar is suitable

Algorithm 5 Pseudocode for learning knowledge embeddings U and proposed tasks embeddings V

Input: Bag-of-words representations of scholars' abstracts X
Output: Knowledge embedding U

1: **function** KnowledgeEmbedding(X)
2: Train knowledge embeddings using an VAE model;
3: **return** U
4: **end function**

Algorithm 6 Pseudocode for learning knowledge embeddings U and proposed tasks embeddings V

Input: Bag-of-words representations of proposed tasks' abstracts Y
Output: Proposed task embedding V;

1: **function** TaskEmbedding(Y)
2: Train task embeddings V using an VAE model;
3: **return** V
4: **end function**

Algorithm 7 Pseudocode for learning knowledge embeddings U and proposed tasks embeddings V

Input: A scholar's abstract x_i, proposed task's abstract y_j
Output: True or False

1: **function** isMatch(x_i, y_j)
2: Get the knowledge embedding u_i from U based on input x_i;
3: Get the task embedding v_j from V based on input y_j;
4: Build a DNN model to make prediction based u_i and v_j;
5: **return** True or False
6: **end function**

for a proposed research task given the particular scholars' publication abstract x_i, and the proposed task abstract y_j shown in Algorithm 7.

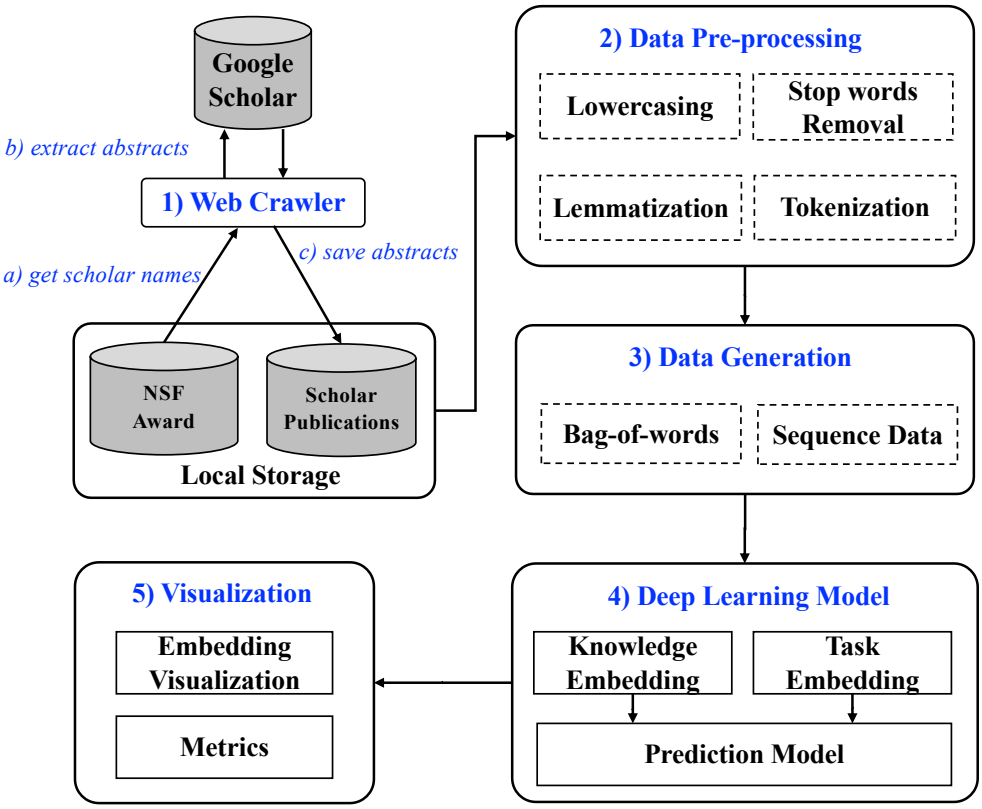


Figure 4.6: System architecture of our ScholarFinder model.

4.2.5 System Architecture

In this section, we present our proposed system architecture for the ScholarFinder model. As shown in Fig. 4.6, there are five major components in our ScholarFinder system architecture: (i) Web Crawler; (ii) Data Pre-processing; (iii) Deep Learning model; and (iv) Visualization.

Web Crawler In our system, the web crawler is used to extract publication abstracts from Google Scholar, which has the following steps shown in Fig. 4.6: a) get all scholars names from NSF award dataset (Details of NSF award dataset will be discussed in Section 4.3.1); b) for each scholar, obtain his/her publications from Google Scholar using Google Scholar APIs [92]; c) for each publication, the web crawler will extract its abstract using Google Scholar APIs; d) save abstracts

into a local storage system for later pre-processing.

Data Pre-processing The goal of pre-processing is to generate bag-of-words for each abstract we collect. The basic steps for data pre-processing are: lowercasing the texts; removing the stop words that are not meaningful words, and significantly frequent (e.g., "the", "a") in texts; lemmatization to reduce or group different inflected forms of a word to a common base form; tokenization to tokenize words from; and bag-of-words to generate words counts for each abstract.

Data Generation In the data generation stage, we use the text dataset to generate different data structures (Bag-of-Words or Sequence) based on the different scenarios. For example, we can use Bag-of-words to understand basic distributions of datasets, and we can use sequence datasets for scholar summarization.

Deep Learning Model In our system, the deep learning model used is the one discussed in Section 4.2 which comprises of a "Knowledge Embedding" model that is used to train each scholar's knowledge embedding based on his/her publication abstracts as bag-of-words representations which are processed in the previous stage. After finishing the training procedure, we obtain pre-trained knowledge embedding for each scholar, which can be used for future prediction, classification, and visualization. In addition, a "Prediction" model is used to leverage the pre-trained knowledge embedding to perform another deep learning model. In our case, we use pre-trained knowledge embedding to predict whether a scholar is suitable for a proposed set of research tasks or not.

Visualization The last part visualization is used to plot embedding in a 2D space, or monitoring the model training status (such as plotting the loss or checking the Reconstruction errors). It can also be used to evaluate the model performance in terms of precision, recall, F1-score, and Accuracy metrics.

4.3 Evaluation

In this section, we evaluate the precision, recall and F1 scores of our model with state-of-the-art XGBoost and Deep Neural Network (DNN) model. Following this, we further demonstrate visualization of embeddings based on our pre-trained knowledge embeddings.

4.3.1 Datasets

In our work, we used two categories of open and large scholar related datasets: *NSF Awards*, dataset, and *Publications* dataset, which cover most of the scientific fields. The *Publications* dataset is used to obtain scholars' knowledge embedding, and from the *NSF Awards* dataset, we extract NSF award abstract and scholars' names who successfully received competitive funding for checking whether a scholar is suitable for an NSF award funding or not.

NSF Awards

The NSF award dataset [93] mainly consists of categorical data related to abstracts by various authors who received NSF grants during the past twenty years. The award summaries consist of funded researchers' project profiles with keywords of research areas, tools, data sets, and research collaborators. The attributes of the dataset include abstract, title, award id, author details such as name, role as (PI) Principle investigator or Co-PI, institution information such as name, address, department, etc. We have collected 20,000 abstracts from the NSF award dataset that contains information that is relevant to 15,074 scholars who successfully obtained competitive funding from NSF grants over the past twenty years. Those scholars who received awards are considered as positive labels. Given the lack of negative samples in our dataset, we apply our negative sampling scheme discussed in Section 4.2.3 to generate the same amount of negative samples for training and testing activities. In order to compare performance of our negative sampling

scheme, we consider a random negative sampling method as the competing solution.

Publications

The web crawler will extract all scholars' publications abstracts from all open publication achieves, and saves them in our local storage system. Then, the collected abstracts are pre-processed by lowercasing, removing "stop words", tokenization, and each abstract is represented as a "bag of words" in our model. And for each author, we will use at most 10 recent papers abstracts in our model.

4.3.2 Experimental Setup

Metrics

we use *Precision, Recall, F1-Score, Accuracy* for our evaluation metrics:

- *Precision* is a metric to measure the ratio of correctly predicted positive labels (TP) to the total predicted positive labels (TP+FP).

$$Precision = \frac{TP}{TP + FP}$$

- *Recall* is the ratio of correctly predicted positive lables (TP) to the all labels (TP + FN) in actual class.

$$Recall = \frac{TP}{TP + FN}$$

- *F1 score* is the Harmonic mean of *Precision* and *Recall*, which consider the impact of both false positives (FP) and false negatives (FN). *F1 score* is usually more useful in the unbalanced dataset.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

- *Accuracy* is metric to measure overall accuracy for both positive and negative labels

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Baselines

To evaluate the proposed model, we consider two state-of-the-art method sets including deep learning model and tradition models. These models are trained using the bag-of-words directly. For the baseline models, we only use the random negative sampling scheme.

- *DNN*: A deep neural network with 3 hidden layers is used in our evaluation experiments. Each layer uses ReLU activation function, and output layer uses sigmoid activation function with cross-entropy loss.
- *XGBoost* [94]: XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. XGBoost is based on Gradient Boosting algorithm and provides a parallel tree boosting (also known as GBDT, GBM) that solves many data science problems in a fast and accurate way.

ScholarFinder

As discussed in Section 4.2.4, we have proposed two DNN models ("Concatenating Layer" and "Dot Product Layer") for demonstrating the use of our pre-trained model to perform predictions. In the evaluation experiments, we evaluate both the proposed DNN models. In addition, evaluation of each model is performed by comparing our proposed negative sampling scheme with the (competing) normal random negative sampling scheme. Thus, the variants of our ScholarFinder models are defined as follows:

- **ScholarFinder-Concatenating (SC)**: ScholarFinder with "Concatenating Layer" using random negative sampling scheme

- **ScholarFinder-Dot (SD)**: ScholarFinder with "Concatenating Layer" using random negative sampling scheme
- **ScholarFinder-Concatenating-Negative (SCN)**: ScholarFinder with "Concatenating Layer" using our proposed negative sampling scheme
- **ScholarFinder-Dot-Negative (SCD)**: ScholarFinder with "Concatenating Layer" using our proposed negative sampling scheme

Table 4.2: Evaluation results of our proposed ScholarFinder models comparison with state-of-the-art XGBoost and DNN models in terms of precision, recall, F1 score and accuracy metrics.

Model	**Label**	**Precision**	**Recall**	**F1-score**	**Accuracy**
XGBoost	+	0.60	0.55	0.57	0.59
	-	0.59	0.64	0.61	
DNN	+	0.74	0.80	0.77	0.76
	-	0.78	0.72	0.75	
SC	+	0.79	0.89	0.84	0.83
	-	0.88	0.76	0.81	
SD	+	0.84	0.83	0.83	0.84
	-	0.83	0.84	0.84	
SCN	+	**0.96**	**0.98**	**0.97**	**0.97**
	-	**0.98**	**0.96**	**0.97**	
SDN	+	0.95	0.94	0.95	0.95
	-	0.94	0.95	0.95	

4.3.3 Evaluation Results

As shown in Table 4.2, all our models show better performance in comparison to the XGBoost and DNN models in terms of all Precision, Recall, F1-Score, and Accuracy metrics across all positive/negative class labels. Particularly, the ScholarFinder-Concatenating model shows slightly better performance than the ScholarFinder-Dot model. Improved performance however comes at the cost of

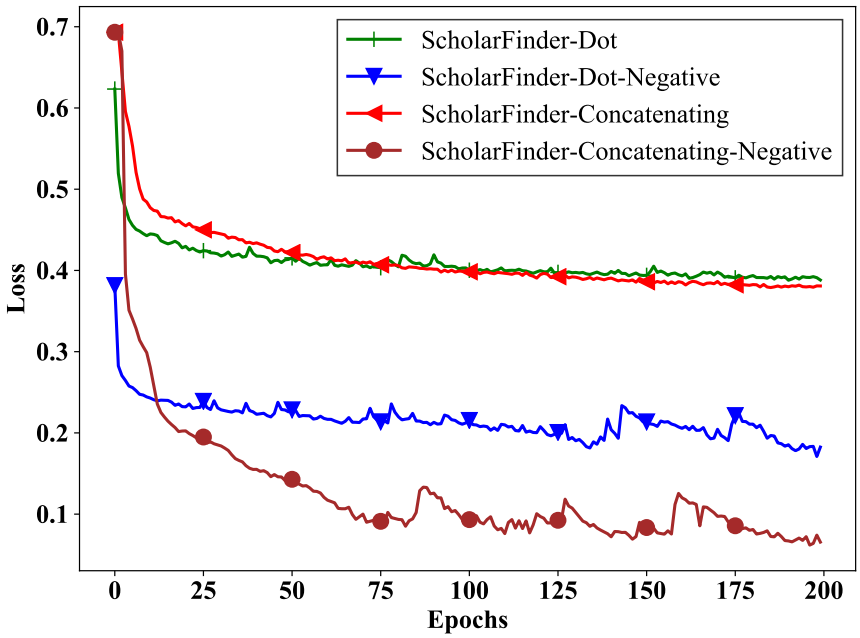
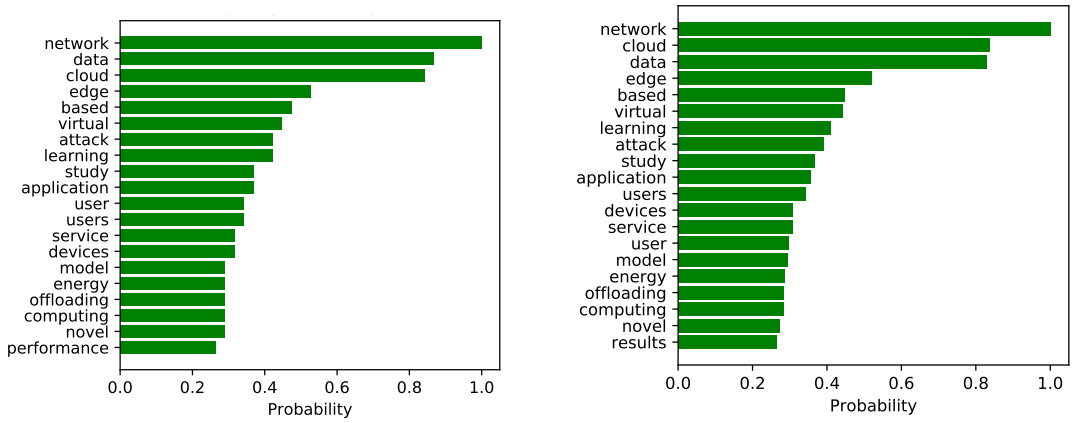


Figure 4.7: Loss comparison results for the different ScholarFinder model variants.

increased use of memory and computation time for training using the pre-trained knowledge embedding. The results also show that our proposed negative sampling scheme can significantly improve the prediction performance by around 10% in all Precision, Recall, F1-Score and Accuracy metrics. Among the ScholarFinder model variants, the ScholarFinder-Concatenating-Negative model clearly achieves the best performance.

The loss plot in Fig. 4.7 shows that the ScholarFinder-Dot model converges faster than the other ScholarFinder-Concatenating models in the cases of both random negative sampling scheme and our proposed negative sampling scheme. However, ScholarFinder-Concatenating models have the ability to achieve better performance as indicated in our loss plots. Nevertheless, using our proposed negative sampling scheme, both ScholarFinder-Dot and the two ScholarFinder-Concatenating models achieve much better performance than the random negative sampling scheme case.



(a) Input of a normalized bag-of-words representation of a scholar's publications with top 20 frequent words shown above.

(b) Reconstruction of a normalized bag-of-words representation of the scholar's publications.

Figure 4.8: Reconstruction performance monitoring of Knowledge embedding using VAE model.

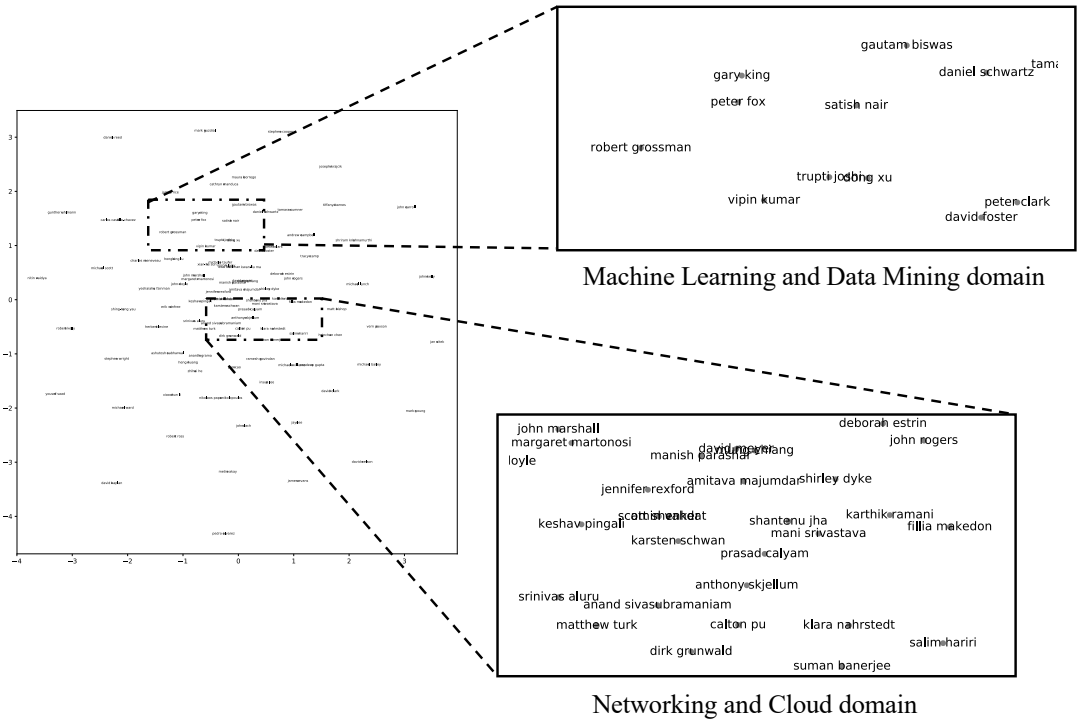


Figure 4.9: Knowledge Embedding in 2D space using VAE model discussed in Section 4.2.2 with $K = 2$ embedding dimension.

4.3.4 Visualization Knowledge Embedding

In order to monitor the learning performance of the knowledge embedding schemes, we can use visualizations of the reconstruction process. As shown in Fig. 4.8, we can see that our model can basically capture the scholar's research areas (such

as cloud computing, networking) through the reconstruction results. In addition, to evaluate the qualities of the entire embedding space, we visualize the scholars' knowledge embedding in a 2D space. A good knowledge embedding is one that is able to capture and group those scholars' with similar research interests. To generating knowledge embeddings in a 2D space, we train the same VAE model discussed in Section 4.2.2 with $K = 2$ embedding dimension.

As shown in Fig. 4.9, we embed only 100 scholars' into a 2D space for the sake of our results discussion. We found that our ScholarFinder model can group those scholars with similar research interests. In this example, we capture the scholars in the "Networking and Cloud" domain, as well as scholars in the "Machine Learning and Data Mining" domain. Thus, our model can learn good embedding for distinguishing scholars based on their research interests, which in turn can help in providing effective scholar recommendations.

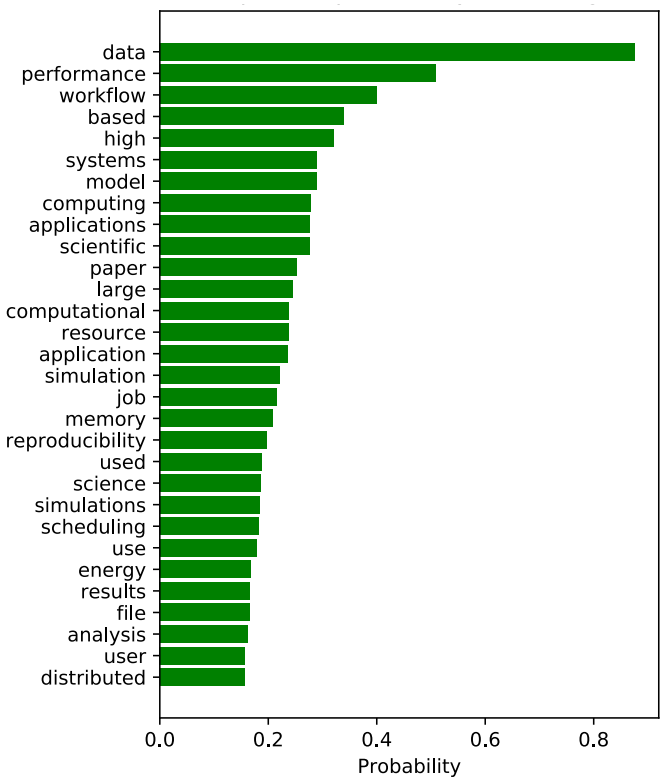


Figure 4.10: Sample a new scholar using a scholar from networking and cloud domains, and a scholar from machine learning and data mining domains.

We explored another interesting phenomenon in our experiments that involves

a generative nature to sample a new scholar using existing knowledge. For example, we can get an expertise middle point between “Networking and Cloud" domain and “Machine Learning and Data Mining" domain, and then use generate network in VAE architecture shown in Fig. 4.2 to sample a scholar, represented as bag-of-words that capture this new scholar’s expertise knowledge. As shown in Fig. 4.10, the scholar we sampled has expertise knowledge in the area of “data computation”, “workflow performance” or ”distributed system”, which are really synergistic and bridge areas to connect between “Networking and Cloud” and “Machine Learning and Data Mining” domains. We can thus validate that our ScholarFinder has generative characteristics given the fact many emerging collaborations among these two domains these days are increasingly focused on building efficient infrastructures for data computing, machine learning, and data analysis workflows.

4.4 Summary

In this paper, we proposed a novel ScholarFinder model to find scholars who are suitable for specific research tasks that require expertise in multiple domains. Our ScholarFinder uses Variational Autoencoder (VAE), to embed their expertise knowledge based on scholars’ publications and each scholar can be represented with latent knowledge representations. These knowledge representations can be used for matching similar interests for collaborations, or to evaluate whether an individual scholar is qualified to perform particular research tasks. We have shown how our method uses pre-trained knowledge embedding to build further predictors or classifiers. We have proposed two different models to investigate how to use pre-trained embeddings to perform predictions or classifications tasks. Leveraging the pre-trained embedding scheme, we also proposed a novel negative sampling scheme that solves the issue of unbalanced labels in our dataset and also significantly improve performance in terms of precision, recall, F1-score and Accuracy metrics. To evaluate our model, we compared the ScholarFinder model variants with state-

of-the-art models (such as DNN and XGBoost). Our evaluation results show that our ScholarFinder model variants consistently can achieve better performance with pre-trained knowledge embedding. In addition, our visualization results show that our embedding can group those scholars with similar research interests and has generative characteristics to be able to sample new scholar recommendations from existing knowledge.

Possible future directions for this work include building visualization interfaces in our ScholarFinder to browse and drill-down knowledge patterns. Our ScholarFinder could be also integrated with a recommendation system or conversational agents (chatbot) for recommending scholars to solve multidisciplinary research tasks in emerging multi-disciplinary areas such as precision medicine, data-driven agriculture and autonomous materials design.

Chapter 5

Conclusion and Future Work

In this thesis, Firstly, we showed the need of a "measurement recommender" for network operators and users to effectively troubleshoot bottlenecks affecting Big Data applications in a trustworthy manner. We leveraged the concept of recommendation system to enhance measurements sharing/subscription, diagnosis expertise sharing and collaborations. Using content-based filtering, we filter and rank best relevant measurement traces from a pool of candidate traces. The recommendation scheme was complemented by a Bayesian Inference based domain reputation calculation scheme that indicates the trustworthiness of the collected samples among the involved domains. With collaborative filtering, we find those measurement traces having similar anomaly symptoms with target measurements, and through measurements traces, we can connect those people for sharing knowledge, or working collaboratively in a trustworthy manner. Using real measurements traces and synthetic events, we showed how our content-based filtering scheme enables operators to intelligently use less but relevant measurement samples to accurately detect and diagnose performance bottleneck causing network events. In addition, we showed how our collaborative filtering scheme finds similar anomaly issues efficiently and effectively.

Secondly, we presented a novel "domain-specific topic model" (DSTM) for discovering latent knowledge patterns among research topics, tools and datasets for

computational and data-intensive scientific communities. Our DSTM is a generative model whose design incorporates as little, or any amount of domain knowledge while exploring highly-specific topic patterns within a given domain. DSTM can shorten the time to knowledge discovery and help scientists/researchers to adapt prior domain knowledge when pursuing multi-disciplinary investigations. Similar to the popular LDA method in prior works, DSTM uses a completely randomly generative process to generate words based on reference tools or datasets. Using large collections of two different text corpus from neuroscience and bioinformatics domains (includes more than 25,000 papers over the ten years from reputed journal archives), our evaluation experiments with quantitative perplexity scores and qualitative domain expert feedback show that our DSTM has better generalization performance for revealing highly specific latent topics within a domain. We have also shown that the information retrieval performance results for DSTM is comparable to other state-of-the-art methods such as LDA and PLSA, when users provide a small length of query set. We proposed three exemplar applications of DSTM with concrete algorithms for *Intra-domain*, *Cross-domain*, and *Trend* knowledge patterns discovery from large datasets obtained from scientific publication archives. We showed how DSTM can be relevant for researchers seeking to make intelligent decisions using knowledge discovery for developing solutions to multi-disciplinary research problems using state-of-the-art best practices for tools and datasets.

Lastly, we proposed a novel ScholarFinder model to find scholars who are suitable for specific research tasks that require expertise in multiple domains. Our ScholarFinder uses Variational Autoencoder (VAE), to embed their expertise knowledge based on scholars' publications and each scholar can be represented with latent knowledge representations. These knowledge representations can be used for matching similar interests for collaborations, or to evaluate whether an individual scholar is qualified to perform particular research tasks. We have shown how our method uses pre-trained knowledge embedding to build further predictors

or classifiers. We have proposed two different models to investigate how to use pre-trained embeddings to perform predictions or classifications tasks. Leveraging the pre-trained embedding scheme, we also proposed a novel negative sampling scheme that solves the issue of unbalanced labels in our dataset and also significantly improve performance in terms of precision, recall, F1-score and Accuracy metrics. To evaluate our model, we compared the ScholarFinder model variants with state-of-the-art models (such as DNN and XGBoost). Our evaluation results show that our ScholarFinder model variants consistently can achieve better performance with pre-trained knowledge embedding. In addition, our visualization results show that our embedding can group those scholars with similar research interests and has generative characteristics to be able to sample new scholar recommendations from existing knowledge.

Possible future work "measurement recommender" could be adapting our "measurement recommender" approach to be more deeply integrated into diverse Big Data application communities, to help them better socialize around measurements and achieve expected performance. Possible future directions for the "topic recommender" include building visualization/drill-down interfaces to browse the knowledge patterns among research topics, tools and datasets. Such interfaces can further foster an efficient query to obtain appropriate resources (e.g., tools, and datasets) for cutting-edge research investigations in many other disciplines (e.g., material science, business analytics). Our DSTM extensions can be developed and integrated within a recommendation system to recommend appropriate distributed computing resource configurations to domain scientists based on their workflow requirements. Further, it can be extended to be used within conversational agents (i.e., chatbots) to help users of science gateways to discover latent knowledge patterns among research topics, tools and datasets in an interactive manner [61]. Possible future directions for "SchoalrFinder" include building visualization interfaces in our ScholarFinder to browse and drill-down knowledge patterns. Our ScholarFinder could be also integrated with a recommendation sys-

tem or conversational agents (chatbot) for recommending scholars to solve multi-disciplinary research tasks in emerging multi-disciplinary areas such as precision medicine, data-driven agriculture and autonomous materials design.

Bibliography

[1] NRC *et al.*, *Research at the intersection of the physical and life sciences.* National Academies Press, 2010.

[2] NAE, *Grand challenges for engineering.* National Academy of Engineering, 2008.

[3] H. Akil, R. Balice-Gordon, D. L. Cardozo, W. Koroshetz, S. M. P. Norris, T. Sherer, S. M. Sherman, and E. Thiels, "Neuroscience training for the 21st century," *Neuron*, vol. 90, no. 5, pp. 917–926, 2016.

[4] R. L. Ramos, G. J. Fokas, A. Bhambri, P. T. Smith, B. H. Hallas, and J. C. Brumberg, "Undergraduate neuroscience education in the us: an analysis using data from the national center for education statistics," *Journal of Undergraduate Neuroscience Education*, vol. 9, no. 2, p. A66, 2011.

[5] E. P. Wiertelak and J. J. Ramirez, "Undergraduate neuroscience education: blueprints for the 21st century," *Journal of Undergraduate Neuroscience Education*, vol. 6, no. 2, p. A34, 2008.

[6] T. Joshi, M. R. Fitzpatrick, S. Chen, Y. Liu, H. Zhang, R. Z. Endacott, E. C. Gaudiello, G. Stacey, H. T. Nguyen, and D. Xu, "Soybean knowledge base (soykb): a web resource for integration of soybean translational genomics and molecular breeding," *Nucleic Acids Research*, vol. 42, no. D1, pp. D1245–D1252, 2014.

[7] T. Joshi, K. Patil, M. R. Fitzpatrick, L. D. Franklin, Q. Yao, J. R. Cook, Z. Wang, M. Libault, L. Brechenmacher, B. Valliyodan, *et al.*, "Soybean knowledge base (soykb): a web resource for soybean translational genomics," *BMC genomics*, vol. 13, no. S1, p. S15, 2012.

[8] D. Grant, R. T. Nelson, S. B. Cannon, and R. C. Shoemaker, "Soybase, the usda-ars soybean genetics and genomics database," *Nucleic acids research*, vol. 38, no. suppl_1, pp. D843–D846, 2010.

[9] C. J. Lawrence, Q. Dong, M. L. Polacco, T. E. Seigfried, and V. Brendel, "Maizegdb, the community database for maize genetics and genomics," *Nucleic acids research*, vol. 32, no. suppl_1, pp. D393–D397, 2004.

[10] V. Krishnakumar, M. R. Hanlon, S. Contrino, E. S. Ferlanti, S. Karamycheva, M. Kim, B. D. Rosen, C.-Y. Cheng, W. Moreira, S. A. Mock, *et al.*, "Araport: the arabidopsis information portal," *Nucleic acids research*, vol. 43, no. D1, pp. D1003–D1009, 2015.

[11] J. Li, X. Dai, T. Liu, and P. X. Zhao, "Legumeip: an integrative database for comparative genomics and transcriptomics of model legumes," *Nucleic acids research*, vol. 40, no. D1, pp. D1221–D1229, 2012.

[12] V. A. Benedito, I. Torres-Jerez, J. D. Murray, A. Andriankaja, S. Allen, K. Kakar, M. Wandrey, J. Verdier, H. Zuber, T. Ott, *et al.*, "A gene expression atlas of the model legume medicago truncatula," *The Plant Journal*, vol. 55, no. 3, pp. 504–513, 2008.

[13] J. He, V. A. Benedito, M. Wang, J. D. Murray, P. X. Zhao, Y. Tang, and M. K. Udvardi, "The medicago truncatula gene expression atlas web server," *BMC bioinformatics*, vol. 10, no. 1, p. 441, 2009.

[14] T. Joshi, M. R. Fitzpatrick, S. Chen, Y. Liu, H. Zhang, R. Z. Endacott, E. C. Gaudiello, G. Stacey, H. T. Nguyen, and D. Xu, "Soybean knowledge base (soykb): a web resource for integration of soybean translational genomics

and molecular breeding," *Nucleic acids research*, vol. 42, no. D1, pp. D1245–D1252, 2013.

[15] S. Zeng, Z. Lyu, S. R. K. Narisetti, D. Xu, and T. Joshi, "Knowledge base commons (kbcommons) v1. 0: A multi omics'web-based data integration framework for biological discoveries," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 589–594, IEEE, 2018.

[16] A. Hanemann, J. W. Boote, E. L. Boyd, J. Durand, L. Kudarimoti, R. Łapacz, D. M. Swany, S. Trocha, and J. Zurawski, "Perfsonar: A service oriented architecture for multi-domain network monitoring," in *International conference on service-oriented computing*, pp. 241–254, Springer, 2005.

[17] V. Paxson, "Strategies for sound internet measurement," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 263–271, 2004.

[18] P. Calyam, J. Pu, W. Mandrawa, and A. Krishnamurthy, "Ontimedetect: Dynamic network anomaly notification in perfsonar deployments," in *2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 328–337, IEEE, 2010.

[19] Y. Liu, S. M. Khan, J. Wang, M. Rynge, Y. Zhang, S. Zeng, S. Chen, J. V. Maldonado dos Santos, B. Valliyodan, P. P. Calyam, N. Merchant, H. T. Nguyen, D. Xu, and T. Joshi, "Pgen: large-scale genomic variations analysis workflow and browser in soykb," *BMC Bioinformatics*, vol. 17, p. 337, Oct 2016.

[20] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[21] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The author-topic model for authors and documents," in *UAI'04*, (Arlington, Virginia, United States), pp. 487–494, AUAI Press, 2004.

[22] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[23] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[25] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, pp. 173–182, International World Wide Web Conferences Steering Committee, 2017.

[26] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '04, (New York, NY, USA), pp. 219–230, ACM, 2004.

[27] L. Zonglin, H. Guangmin, Y. Xingmiao, and Y. Dan, "Detecting distributed network traffic anomaly with network-wide correlation analysis," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, no. 1, p. 752818, 2008.

[28] P. Calyam, M. Dhanapalan, M. Sridharan, A. Krishnamurthy, and R. Ramnath, "Topology-aware correlated network anomaly event detection and diagnosis," *Journal of Network and Systems Management*, vol. 22, pp. 208–234, Apr 2014.

[29] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC '05, (Berkeley, CA, USA), pp. 31–31, USENIX Association, 2005.

[30] A. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J. Wang, Z. Ge, and C. T. Ee, "Troubleshooting chronic conditions in large ip networks," in *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, (New York, NY, USA), pp. 2:1–2:12, ACM, 2008.

[31] Y. Zhou and G. Hu, "Network-wide anomaly detection based on router connection relationships," *IEICE transactions on communications*, vol. 94, no. 8, pp. 2239–2242, 2011.

[32] P. Kanuparthy and C. Dovrolis, "Pythia: Diagnosing performance problems in wide area providers," in *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, (Berkeley, CA, USA), pp. 371–382, USENIX Association, 2014.

[33] M. A. Marvasti, A. V. Poghosyan, A. N. Harutyunyan, and N. M. Grigoryan, "An enterprise dynamic thresholding system," in *11th International Conference on Autonomic Computing ({ICAC} 14)*, pp. 129–135, 2014.

[34] A. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J. Wang, Z. Ge, and C. T. Ee, "Troubleshooting chronic conditions in large ip networks," in *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, (New York, NY, USA), pp. 2:1–2:12, ACM, 2008.

[35] P. Kanuparthy, D. H. Lee, W. Matthews, C. Dovrolis, and S. Zarifzadeh, "Pythia: detection, localization, and diagnosis of performance problems," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 55–62, 2013.

[36] K. H. Kim, H. Nam, V. Singh, D. Song, and H. Schulzrinne, "Dyswis: Crowdsourcing a home network diagnosis," in *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–10, Aug 2014.

[37] M. Grigoriev, P. DeMar, B. Tierney, A. Lake, J. Metzger, M. Frey, and P. Calyam, "E-center: A collaborative platform for wide area network users,"

in *Journal of Physics: Conference Series*, vol. 396, p. 042025, IOP Publishing, 2012.

[38] R. Soundararajan, E. Celebi, L. Spracklen, H. Muppalla, and V. Makhija, "A social-media approach to virtualization management," *VMware TJ*, vol. 1, no. 2, pp. 59–68, 2012.

[39] Y. Zhang, S. Debroy, and P. Calyam, "Network-wide anomaly event detection and diagnosis with perfsonar," *IEEE Transactions on Network and Service Management*, vol. 13, pp. 666–680, Sept 2016.

[40] L. Xiong and L. Liu, "A reputation-based trust model for peer-to-peer e-commerce communities," in *EEE International Conference on E-Commerce, 2003. CEC 2003.*, pp. 275–284, IEEE, 2003.

[41] L. Mui, M. Mohtashemi, and A. Halberstadt, "A computational model of trust and reputation," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pp. 2431–2439, IEEE, 2002.

[42] A. Josang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th bled electronic commerce conference*, vol. 5, pp. 2502–2511, 2002.

[43] I. Ivanov, P. Vajda, J.-S. Lee, and T. Ebrahimi, "In tags we trust: Trust modeling in social tagging of multimedia content," *IEEE Signal Processing Magazine*, vol. 29, no. 2, pp. 98–107, 2012.

[44] S. L. Lim and A. Finkelstein, "Stakerare: Using social networks and collaborative filtering for large-scale requirements elicitation," *IEEE Transactions on Software Engineering*, vol. 38, pp. 707–735, May 2012.

[45] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 448–456, 2011.

[46] Y. Zhang, S. Debroy, and P. Calyam, "Network measurement recommendations for performance bottleneck correlation analysis," in *2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pp. 1–7, IEEE, 2016.

[47] Y. Zhang, P. Calyam, S. Debroy, and M. Sridharan, "Pca-based network-wide correlated anomaly event detection and diagnosis," in *2015 11th International Conference on the Design of Reliable Communication Networks (DRCN)*, pp. 149–156, March 2015.

[48] P. Calyam, J. Pu, W. Mandrawa, and A. Krishnamurthy, "Ontimedetect: Dynamic network anomaly notification in perfsonar deployments," in *2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 328–337, Aug 2010.

[49] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "Geni: A federated testbed for innovative network experiments," *Comput. Netw.*, vol. 61, pp. 5–23, Mar. 2014.

[50] P. Calyam, M. Sridharan, Y. Xu, K. Zhu, A. Berryman, R. Patali, and A. Venkataraman, "Enabling performance intelligence for application adaptation in the future internet," *Journal of Communications and Networks*, vol. 13, pp. 591–601, Dec 2011.

[51] S. Hemminger *et al.*, "Network emulation with netem," in *Linux conf au*, pp. 18–23, 2005.

[52] D. Mimno and A. McCallum, "Expertise modeling for matching papers with reviewers," in *KDD'07*, (New York, NY, USA), pp. 500–509, ACM, 2007.

[53] D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *ICML'06*, (New York, NY, USA), pp. 113–120, ACM, 2006.

[54] J. D. Lafferty and D. M. Blei, "Correlated topic models," in *NIPS'06* (Y. Weiss, B. Schölkopf, and J. C. Platt, eds.), pp. 147–154, MIT Press, 2006.

[55] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *CVPR'05*, vol. 2, pp. 524–531 vol. 2, June 2005.

[56] P. Flaherty, G. Giaever, J. Kumm, M. I. Jordan, and A. P. Arkin, "A latent variable model for chemogenomic profiling," *Bioinformatics*, vol. 21, no. 15, pp. 3286–3293, 2005.

[57] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, (New York, NY, USA), pp. 448–456, ACM, 2011.

[58] Z. Luo, L. Liu, J. Yin, Y. Li, and Z. Wu, "Latent ability model: A generative probabilistic learning framework for workforce analytics," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 5, pp. 923–937, 2018.

[59] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *NIPS'10* (J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, eds.), pp. 856–864, Curran Associates, Inc., 2010.

[60] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.

[61] Y. Zhang, P. Calyam, T. Joshi, S. Nair, and D. Xu, "Domain-specific topic model for knowledge discovery through conversational agents in data intensive scientific communities," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 4886–4895, IEEE, 2018.

[62] A. Tiroshi, S. Berkovsky, M. A. Kaafar, T. Chen, and T. Kuflik, "Cross social networks interests predictions based ongraph features," in *Proceedings of the 7th ACM Conference on Recommender Systems*, pp. 319–322, ACM, 2013.

[63] Y. Shi, M. Larson, and A. Hanjalic, "Tags as bridges between domains: Improving recommendation with tag-induced cross-domain collaborative filtering," in *International Conference on User Modeling, Adaptation, and Personalization*, pp. 305–316, Springer, 2011.

[64] W. Pan, E. W. Xiang, and Q. Yang, "Transfer learning in collaborative filtering with uncertain ratings," in *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[65] C. Sievert and K. Shirley, "Ldavis: A method for visualizing and interpreting topics," in *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pp. 63–70, 2014.

[66] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[67] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[68] N. T. Carnevale and M. L. Hines, *The NEURON book*. Cambridge University Press, 2006.

[69] A. P. Davison, D. Brüderle, J. M. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger, "Pynn: a common interface for neuronal network simulators," *Frontiers in neuroinformatics*, vol. 2, p. 11, 2009.

[70] M. Migliore, T. M. Morse, A. P. Davison, L. Marenco, G. M. Shepherd, and M. L. Hines, "Modeldb," *Neuroinformatics*, vol. 1, pp. 135–139, Mar 2003.

[71] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New York, NY, USA), pp. 50–57, ACM, 1999.

[72] M. A. Newton and A. E. Raftery, "Approximate bayesian inference with the weighted likelihood bootstrap," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 56, no. 1, pp. 3–26, 1994.

[73] H. Li, J. Ruan, and R. Durbin, "Maq: Mapping and assembly with qualities," *Version 0.6*, vol. 3, 2008.

[74] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[75] A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, *et al.*, "The sequence alignment/map (sam) format and samtools," *Bioinformatics*, vol. 25, pp. 2078–2079, 2009.

[76] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg, "Ultrafast and memory-efficient alignment of short dna sequences to the human genome," *Genome Biology*, vol. 10, p. R25, Mar 2009.

[77] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 675–678, ACM, 2014.

[78] H. Geoffrey H., "Learning distributed representations of concepts," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, 1986.

[79] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[80] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[81] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[82] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*, pp. 791–798, ACM, 2007.

[83] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 233–240, ACM, 2016.

[84] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1809–1818, ACM, 2015.

[85] D. Li, J. Zhang, and P. Li, "Representation learning for question classification via topic sparse autoencoder and entity embedding," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 126–133, IEEE, 2018.

[86] R. Ramanath, H. Inan, G. Polatkan, B. Hu, Q. Guo, C. Ozcaglar, X. Wu, K. Kenthapadi, and S. C. Geyik, "Towards deep and representation learning for talent search at linkedin," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 2253–2261, ACM, 2018.

[87] Y. Miao, L. Yu, and P. Blunsom, "Neural variational inference for text processing," in *International conference on machine learning*, pp. 1727–1736, 2016.

[88] "Sciencedirect." https://www.sciencedirect.com/topics/index. Accessed: 2019-08-19.

[89] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323, 2011.

[90] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[91] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[92] "Google scholar apis." https://pypi.org/project/scholarly/. Accessed: 2019-10-15.

[93] "National science foundation." https://nsf.gov/awardsearch/. Accessed: 2019-10-15.

[94] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, ACM, 2016.

VITA

Yuanxun Zhang was born in Chengdu, China. Currently, he is a Ph.D. Candidate in Computer Science at the University of Missouri - Columbia, advised by Dr. Prasad Calyam. Before that, he graduated from Southwest Jiaotong University with a B.E in Computer Science degree in June 2006. After that, he spent more than 7 years as a software engineer at Huawei Technologies Co., Ltd.

His Ph.D. studies focus on the theory and practice of understanding and learning from data to make better decisions or quantifying uncertainty. He investigated solutions to data science problems involving: time-series anomaly detection, measurement analysis, machine learning, deep learning, statistical learning, probabilistic graphical model, inference algorithm and information retrieval.

During his Ph.D studies, he has published 10 peer-reviewed publications in reputed conferences and journals including (IEEE LANMAN, IEEE DRCN, IEEE TNSM, IEEE TCC, IEEE BigData, IEEE BigDataService and others). He also completed a research internship at JD Digits AI Lab on NLP projects in 2020.