MULTI-DIMENSIONAL MINING OF UNSTRUCTURED DATA WITH LIMITED SUPERVISION

BY

CHAO ZHANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair
Professor ChengXiang Zhai
Professor Tarek Abdelzaher
Professor Qiaozhu Mei

ABSTRACT

As one of the most important data forms, unstructured text data plays a crucial role in data-driven decision making in domains ranging from social networking and information retrieval to healthcare and scientific research. In many emerging applications, people's information needs from text data are becoming multi-dimensional—they demand useful insights for multiple aspects from the given text corpus. However, turning massive text data into multi-dimensional knowledge remains a challenge that cannot be readily addressed by existing data mining techniques.

In this thesis, we propose algorithms that turn unstructured text data into multi-dimensional knowledge with limited supervision. We investigate two core questions:

1. *How to identify task-relevant data with declarative queries in multiple dimensions?*
2. *How to distill knowledge from data in a multi-dimensional space?*

To address the above questions, we propose an integrated cube construction and exploitation framework. First, we develop a **cube construction** module that organizes unstructured data into a cube structure, by discovering latent multi-dimensional and multi-granular structure from the unstructured text corpus and allocating documents into the structure. Second, we develop a **cube exploitation** module that models multiple dimensions in the cube space, thereby distilling multi-dimensional knowledge from data to provide insights along multiple dimensions. Together, these two modules constitute an integrated pipeline: leveraging the cube structure, users can perform multi-dimensional, multi-granular data selection with declarative queries; and with cube exploitation algorithms, users can make accurate cross-dimension predictions or extract multi-dimensional patterns for decision making.

The proposed framework has two distinctive advantages when turning text data into multi-dimensional knowledge: **flexibility** and **label-efficiency**. First, it enables acquiring multi-dimensional knowledge flexibly, as the cube structure allows users to easily identify task-relevant data along multiple dimensions at varied granularities and further distill multi-dimensional knowledge. Second, the algorithms for cube construction and exploitation require little supervision; this makes the framework appealing for many applications where labeled data are expensive to obtain.

To my family and Shili, for their love and support.

ACKNOWLEDGMENTS

First and foremost, I want to thank Professor Jiawei Han—the most ideal advisor one can hope for. As an advisor, Jiawei has provided tremendous support and guidance to train me into a researcher throughout my Ph.D. study. He has taught me essential skills to become a mature researcher, from identifying and formulating important problems, to developing solutions critically and rigorously, to collaborating with other researchers, and to communicating ideas clearly. The influence of Jiawei is beyond the role of advior—he is also a life-long role model to me. I have learned from him the courage to take challenges, the positive attitude in front of setbacks, the diligence and self-discipline, and many other characteristics. Such influences will be a forever treasure in my life.

I would like to thank my thesis committee members, Professor ChengXiang Zhai, Professor Tarek F. Abdelzaher, and Professor Qiaozhu Mei. They have provided not only careful commentary on this document, but also tremendous help in my job search process and invaluable advice for crafting a successful academic career.

I have been fortunate to interact with and learn from many wonderful researchers in academia: Dr. Jessie Zhenhui Li, Dr. Yizhou Sun, Dr. Xifeng Yan, Dr. Feida Zhu, Dr. Jian Pei, Dr. Quanquan Gu, Dr. Lu Su, Dr. Jing Gao, Dr. Yu Zheng, Dr. Shaowen Wang, Dr. Phillip Yu, Dr. Hong Cheng, Dr. Jimeng Sun, Dr. Anthony K. H. Tung. As experienced researchers, they have given me a lot of help and advice both on my research and on career development. Special thanks to Jessie, it is because of our conversation in KDD 2016 that I finally decided to join academia after graduation.

Many thanks to my dear friends in the DMG and DAIS family! Alphabetically, they are: Xiusi Chen, Joe Chen, Yucheng Chen, Ahmed El-Kishky, Xiaotao Gu, Huan Gui, Meng Jiang, Shan Jiang, Dongming Lei, Min Li, Ji Li, Qi Li, Zoey Li, De Liao, Liyuan Liu, Mengxiong Liu, Jialu Liu, Yuning Mao, Yu Meng, Meng Qu, Xiang Ren, Jingbo Shang, Jiaming Shen, Yu Shi, Fangbo Tao, Wenzhu Tong, Frank Xu, Chenguang Wang, Chi Wang, Qi Wang, Jingjing Wang, Xuan Wang, Ellen Wu, Carl Yang, Hongkun Yu, Xiao Yu, Quan Yuan, Keyang Zhang, Yu Zhang, Shi Zhi, Qi Zhu, Wanzheng Zhu, and Honglei Zhuang. Because of them, Ph.D. life at UIUC has been so much fun. I am indebted to many of them who took great care of me during my achilles tendon injury. It is their care and encouragement that helped me through the recovery process. I wish all of them a successful career and a happy life.

I am very grateful to all my collaborators outside the UIUC data mining group: Tim

Hanratty, Lance Kaplan, Shuochao Yao, Huajie Shao, Wei Zhang, Wei Feng, Yong Li, Di Yao, Jie Feng, Julie Yixuan Zhu, and Luming Zhang. The collaborations with them are always enjoyable—I love working together in idea brainstorming, coding, paper writing, and catching up with conference deadlines. Our collaborations have been so fruitful. I look foward to future opportunities in collaborating with them and meeting them at conferences.

I also want express my deep gratitude to my M.S. advisors at Zhejiang University: Professor Gang Chen, Professor Lidan Shou, and Professor Ke Chen. They guided me to step into Computer Science research and realize its beauty in an early stage. Their hands-on mentoring in research methodology, intellectual audacity, writing, and presentation prepared me so well for starting my Ph.D. at UIUC. And over time, I have realized how important these skills are to my academic career.

Finally, my deepest gratitude to my family! I thank my parents and sister for their endless love and constant support in every aspect of my life. Special thanks to Shili—my dear sunshine lighting up my world. Without you I would not have made it this far and been so happy every day. I am so fortunate to have you around, and want to thank you for your love, encouragement, and support in this wonderful journey.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION 1
- 1.1 Overview 1
- 1.2 Main Modules 3
- 1.3 Technical Contributions 7
- 1.4 Organization 11

CHAPTER 2 TAXONOMY GENERATION FOR CUBE SCHEMA DISCOVERY . 12
- 2.1 Overview 12
- 2.2 Related Work 16
- 2.3 Preliminaries 18
- 2.4 Adaptive Term Clustering 19
- 2.5 Adaptive Term Embedding 22
- 2.6 Experimental Evaluation 24
- 2.7 Summary 31

CHAPTER 3 DOCUMENT ALLOCATION WITHOUT LABELED DATA 32
- 3.1 Overview 32
- 3.2 Related Work 35
- 3.3 Preliminaries 36
- 3.4 Graph-Based Joint Embedding 38
- 3.5 Dimension-Aware Embedding Learning 40
- 3.6 The Overall Doc2Cube Algorithm 43
- 3.7 Experimental Evaluation 45
- 3.8 Summary 53

CHAPTER 4 CROSS-DIMENSION PREDICTION IN CUBE SPACE 54
- 4.1 Overview 54
- 4.2 Related Work 57
- 4.3 Preliminaries 57
- 4.4 Semi-Supervised Multimodal Embedding 59
- 4.5 Online Updating of Multimodal Embedding 63
- 4.6 Experiments 68
- 4.7 Summary 80

CHAPTER 5 EVENT DETECTION IN CUBE SPACE 82
- 5.1 Overview 82
- 5.2 Related Work 84
- 5.3 Preliminaries 86
- 5.4 Candidate Generation 90
- 5.5 Candidate Classification 93

5.6 Supporting Continuous Event Detection 94
5.7 Complexity Analysis 94
5.8 Experiments 95
5.9 Summary 103

CHAPTER 6 CONCLUSION 104
6.1 Summary 104
6.2 Future Work 105

REFERENCES 107

CHAPTER 1: INTRODUCTION

1.1 OVERVIEW

Text is one of the most important data forms for the human to record and communicate information. In a wide spectrum of domains, hundreds of millions of textual contents are being created, shared, and analyzed every single day—examples including tweets, news articles, Google searches, and medical notes. It is estimated that more than 80% of human knowledge is encoded in the form of unstructured text [29]. Due to such ubiquity, extracting useful insights from text data is crucial to decision making in various applications, ranging from automated medical diagnosis [12, 90] and disaster management [52, 75] to fraud detection [80, 50] and personalized recommendation [53, 51].

In many newly emerged applications, people's information needs from text data are becoming **multi-dimensional**. That is, people can demand useful insights for **multiple aspects** from the given text corpus. Consider an analyst who wants to use news corpora for disaster analytics. Upon identifying all disaster-related news articles, she needs to understand *what* disaster each article is about, *where* and *when* is the disaster, and *who* are involved. She may even need to explore the multi-dimensional *what-where-when-who* space at varied granularities, so as to answer questions like 'what are all the hurricanes happened in the US in 2018' or 'what are all the disasters happened in California in June'. Disaster analytics is only one of the many applications where multi-dimensional knowledge is required. To name a few other examples, (1) analyzing a biomedical research corpus often requires digging out what *genes*, *proteins*, and *diseases* each paper is about and uncover their inter-correlations; (2) leveraging the medical notes of diabetes patients for automatic diagnosis requires correlating *symptoms* with *genders*, *ages*, and even *geographical regions*; (3) analyzing Twitter data for an presidential election event requires understanding people's sentiments about various *political aspects* in different *geographical regions* and *time periods*.

Acquiring such multi-dimensional knowledge is made possible due to the rich context information in text data. Such context information can arrive in the form of explicit meta-data, such as geographical locations and timestamps alongside Google searches, creating time and geo-tags associated with tweets, and patients' meta-data linked with medical notes. Alternatively, they can arrive in the form of implicit information mentioned in the text itself, such as various entity information (location, person, time, *etc.*) in news articles, point-of-interest names in tweets, and various typed concepts in biomedical papers. It is the availability of such rich context information that enables understanding the text along multiple dimensions and extracting multi-dimensional knowledge for decision making.

In this thesis, our goal is to develop algorithms that facilitate **turning massive unstructured text data into multi-dimensional knowledge for decision making**. We investigate two core questions for this goal:

1. *How to identify task-relevant data with declarative queries in multiple dimensions?*

2. *How to distill knowledge from data in a multi-dimensional space?*

We propose a cube-based framework that approaches the above two questions with minimal supervision. As shown in Figure 1.1, our proposed framework consists of two key modules. First, the **cube construction** module organizes unstructured data into a *multi-dimensional* and *multi-granular* cube structure. With the cube structure, users can identify task-relevant data by specifying query clauses along multiple dimensions at varied granularities. Second, the **cube exploitation** module consists of a set of algorithms that extract useful patterns by jointly modeling multiple dimensions in the cube space. Specifically, it offers algorithms that make predictions across different dimensions or identify unusual events in the cube space. Together, these two modules constitute an integrated pipeline: (1) leveraging the cube structure, users can perform multi-dimensional, multi-granular data selection with declarative queries; and (2) with cube exploitation algorithms, users can make accurate cross-dimension predictions or extract multi-dimensional patterns for decision making.

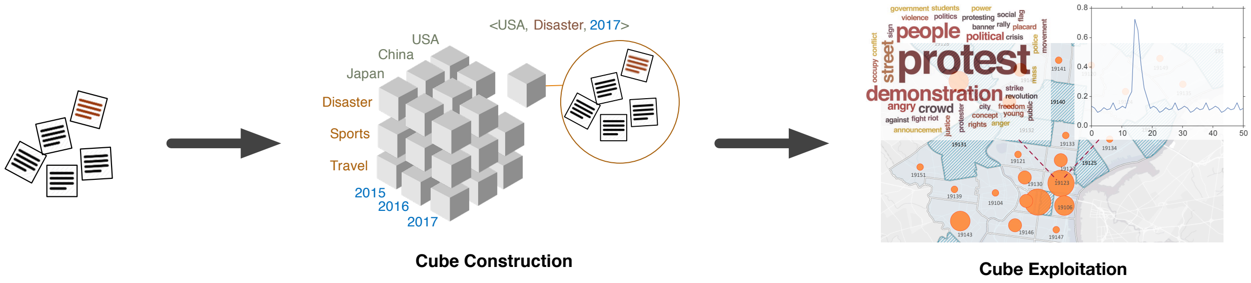


Figure 1.1: Our proposed framework consists of two key modules: (1) a cube construction module that organizes the input data into a multi-dimensional, multi-granular cube structure; (2) a cube exploitation module that discovers interesting patterns in the multi-dimensional space.

Our work has two distinctive advantages when turning text data into multi-dimensional knowledge: **flexibility** and **label-efficiency**. First, it enables acquiring multi-dimensional knowledge flexibly by virtue of the cube structure. Notably, users can use concise declarative queries to identify relevant data along multiple dimensions at varied granularities, *e.g.*,

⟨topic=‘disaster’, location = ‘US’, time=‘2018’⟩, or ⟨topic=‘earthquake’, location = ‘California’, time=‘June’⟩[1], and then apply any data mining primitives (*e.g.*, event detection, sentiment analysis, summarization, visualization) for subsequent analysis. Second, it is label-efficient and thus applicable to many real-life scenarios where labeled data are unavailable. For both cube construction and exploitation, our proposed algorithms all require no or little supervision. This property breaks the bottleneck of lacking labeled data and makes our framework attractive in applications where acquiring labeled data is expensive.

Our framework represents a departure from existing techniques for multi-dimensional data analysis. Data warehouse and online analytical processing techniques [34, 15] have been successful in multi-dimensional analysis of structured data. They allow end users to perform ad-hoc analysis of structured data along multiple dimensions to acquire task-specific insights. Unfortunately, the task of extracting multi-dimensional knowledge from text challenges conventional data warehousing techniques. It is not only because the schema of the cube structure remains unknown, but also allocating the text documents into different cells in the cube space is difficult. Thus, our proposed algorithms bridge the gap between data warehousing and multi-dimensional analysis of unstructured text data.

Along with another line, our work is closely related to text mining. Nevertheless, the success stories of existing text mining techniques still largely rely on the supervised learning paradigm. For instance, our document allocation problem is related to multi-dimensional text classification, yet existing text classification models use massive amounts of labeled documents to learn classification models. Event detection is another example: event extraction techniques in the natural language processing community rely on human-curated sentences to train discriminative models that determine whether a specific type of event has occurred; but if we are to build an event alarm system, it is hardly possible to enumerate all event types and manually curate enough training data for each type. Our work complements existing text mining techniques with unsupervised or weakly-supervised algorithms, which distill knowledge from the given text data with limited supervision.

1.2 MAIN MODULES

We propose an integrated framework that turns unstructured data into multi-dimensional knowledge with limited supervision. As aforementioned, there are two main modules in our proposed framework: (1) a *cube construction* module that organizes unstructured data a multi-dimensional, multi-granular structure; and (2) a *cube exploitation* module that extracts

[1] In many parts of the thesis, we many abbreviate ⟨topic=‘disaster’, location = ‘US’, time=‘2018’⟩ as ⟨disaster, US, 2018⟩ for brevity.

multi-dimensional knowledge in the cube space. In this section, we provide a more detailed overview of these two modules, and then illustrate a number of example applications of our framework.

1.2.1 Module I: Cube Construction

Extracting multi-dimensional knowledge from the unstructured text for decision making necessarily begins with identifying task-relevant data. When an analyst exploits the Twitter stream for sentiment analysis of the 2016 Presidential Election, she may want to retrieve all tweets discussing this event by California users in 2016. Such information needs are often structured and multi-dimensional, yet the input data are unstructured text. Naturally, the first critical question is, *can we use declarative queries along multiple dimensions to identify task-relevant data for on-demand analytics*?

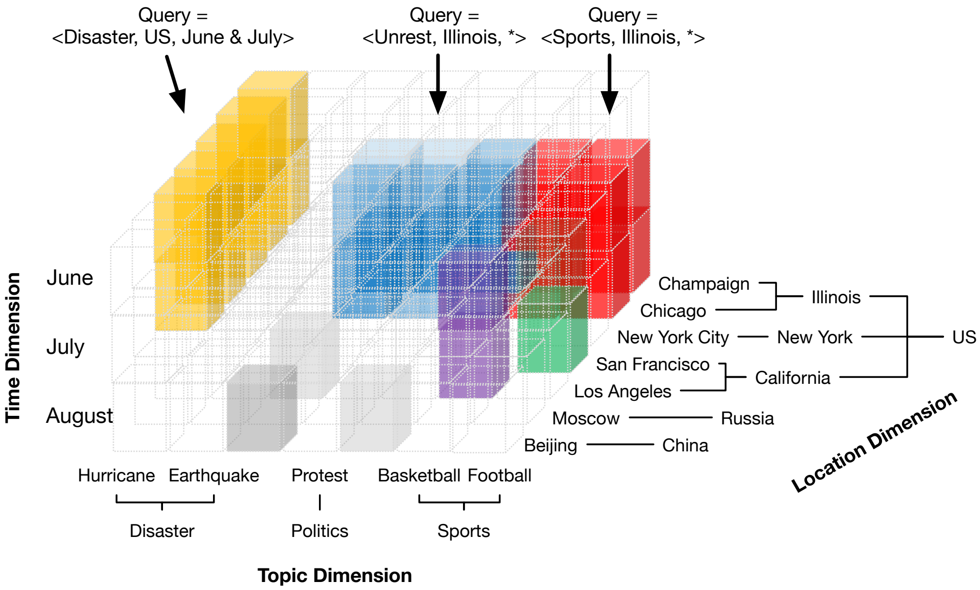


Figure 1.2: An example three-dimensional (topic, location, time) cube structure constructed for social media data. Each dimension has a taxonomic structure, and the three dimensions partition the whole data space into a three-dimensional, multi-granular structure with social media records residing in. End users can use flexible queries to retrieve relevant data for on-demand data analysis.

We approach this question by organizing massive unstructured data into a neat text cube structure, with minimum supervision. For example, Figure 1.2 shows a three-dimensional topic-location-time cube, where each dimension has a taxonomic structure automatically discovered from the input text corpus. With the multi-dimensional, multi-granular cube structure, users can easily explore the data space and select relevant data with structured and declarative queries, *e.g.*, ⟨topic='hurricane', location='Florida', time='2017'⟩, ⟨topic='disaster', location='Florida', time='*'⟩. Better still, they can subsequently apply any statistical primitives (*e.g.*, sum, count, mean) or machine learning tools (*e.g.*, sentiment analysis, text summarization) on the selected data to facilitate on-the-fly exploration.

To turn the unstructured data into such a multi-dimensional, multi-granular cube, there are two central subtasks: (1) taxonomy generation; and (2) document allocation. The first task aims at automatically defining the cube schema from data by discovering the taxonomic structure for each dimension; the second aims at allocating documents into proper cells in the cube. While the two subtasks are closely related to existing literature on taxonomy generation and document classification, most existing methods are inapplicable because they rely on excessive training data. Later, we will propose methods that require no labeled data by learning task-specific text embeddings for cube construction.

1.2.2 Module II: Cube Exploitation

Raw unstructured text data (*e.g.*, social media, SMS messages) are often noisy. Identifying relevant data is merely the first step of the multi-dimensional analytics pipeline. Upon identifying relevant data, next question is to distill interesting multi-dimensional patterns in the cube space to aid decision making. Continue the example in Figure 1.2: Can we detect abnormal activities happened in the New York City in 2017—this translates to the task of finding abnormal patterns in the cube cell ⟨*, New York City, 2017⟩? Can we predict where traffic jams are most likely to take place in Los Angeles around 5pm—this translates to the task of making predictions with data residing in the cube cell ⟨Travel, Los Angeles, 5pm⟩. Can we find out how the earthquake hotspots evolve in the US—this translates to the task of finding evolving patterns across a series of cube cells matching the query ⟨Earthquake, US, *⟩.

The *cube exploitation* module answers the above questions by offering a set of algorithms that discover multi-dimensional knowledge in the cube space. The unique characteristic here is that it is necessary to jointly model multiple factors and uncover their collective patterns in the multi-dimensional space. Under this principle, we investigate two essential tasks in the cube exploitation module: (1) We first study the cross-dimension prediction problem, which aims at modeling the correlations among multiple dimensions (*e.g.*, topic, location, time) for predictive analytics. This leads to the development of a cross-dimension prediction model that can make predictions across different dimensions, so as to answer questions like 'where do NYC people go for a drink at 9pm' or 'what are the correlated genes for breast cancer'. (2)

We then study the abnormal pattern discovery problem, which aims at detecting abnormal patterns in any cube cell. The discovered patterns reflect abnormal behaviors with respect to the concrete contexts of the user-selected cell, *e.g.*, ⟨Los Angeles, *, 2018⟩, ⟨Protest, US, 2018⟩, providing context-aware insights that help with decision making.

1.2.3 Example Applications

Together, the *cube construction* module and the *cube exploitation* module serve as an integrated framework for many applications that demand multi-dimensional knowledge. The modules can be either used by themselves or combined with other existing data mining primitives for the end task. In the following, we provide several examples to illustrate the applications of our proposed framework.

Example Application I: Disaster Detection and Relief

Social media has shown to be an important source for detecting disastrous events (*e.g.*, wildfire, hurricane) in real time. When an emergent disaster outbreaks, social media websites can be instantly filled with reports from witnesses long before the event is covered by traditional news sources. With our work, it is possible to structure massive social media streams into a *what-where-when-who* cube for disaster analytics. With the cube structure, an analyst can easily find out not only what is happening, but also where it is, who are involved, and how it is evolving. She can further visualize the information in the multi-dimensional cube space or identify abnormal disaster occurring patterns. Such multi-dimensional knowledge is highly useful for taking effective disaster-relief actions.

Example Application II: Biomedical Literature Mining

PubMed hosts as many as 27.3 million research articles and serves as an indispensable database for biomedical research. As such a massive amount of biomedical papers are too enormous for the human to analyze, automated analysis of such a large biomedical literature corpus is becoming a pressing need. Consider a system that can automatically organize all the papers according to multiple facets (*e.g.*, diseases, genes, proteins and chemicals). This results in a *disease-gene-protein-chemical* cube, which enables quick retrieval of relevant biomedical papers with simple queries (*e.g.*, ⟨disease = '*breast cancer*', gene = '*BRCA1*', *, * ⟩). It is also possible to model gene-disease correlations in the multi-dimensional space and make predictions for inspiring new biomedical research.

Example Application III: Contextual Sentiment Analysis

Suppose a smartphone company (*e.g.*, Apple) wants to understand their users' attitudes towards a product (*e.g.*, iPhone X) from massive customer reviews. To design most effective advertising and product-upgrading strategies, it is critical for analysts to understand sentiments for different user groups (*e.g.*, partitioned by gender, age, location) about different aspects (*e.g.*, price, size, battery, speed) of the product. For this purpose, it is feasible to apply our work on the review corpus to construct a product review cube. With such a cube structure, the analyst can select data along multiple dimensions at varied granularities, make predictions across dimensions, and apply sentiment analysis tools for context-dependent opinions.

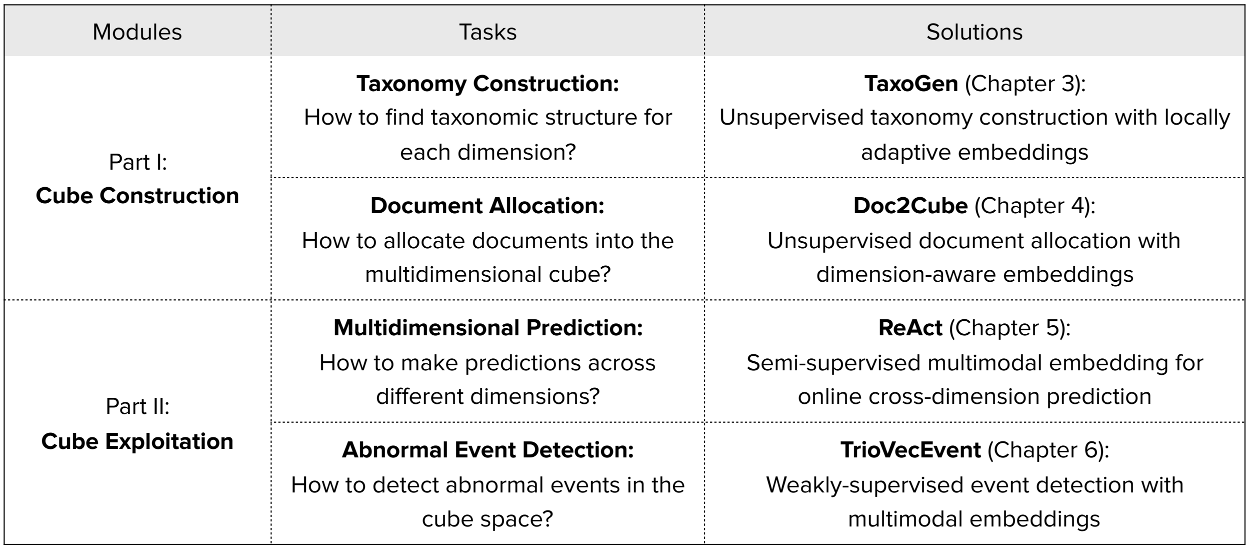
1.3 TECHNICAL CONTRIBUTIONS

Towards the goal of turning unstructured data into multi-dimensional knowledge, our technical contributions are **unsupervised or weakly supervised** methods that perform cube construction and exploitation. Specifically, we develop a set of embedding-based techniques for cube construction and exploitation with limited supervision. In the first part, we propose unsupervised algorithms that learn *task-specific embeddings* for cube construction. In the second part, we present algorithms that leverage *multimodal embeddings* for extracting multi-dimensional knowledge in the cube space.

Figure 1.3 gives an overview of our proposed methods. In the cube construction part, we propose (1) TaxoGen, an unsupervised method that constructs topic taxonomy from a text corpus by learning locally adaptive embeddings[102]; and (2) Doc2Cube, an unsupervised method that performs multi-dimensional document classification by learning dimension-aware embeddings [85]. In the cube exploitation part, we propose (1) CrossMap, a semi-supervised method that learns multimodal embeddings for cross-dimension prediction [103, 104]; and (2) TrioVecEvent, a weakly-supervised method that detects abnormal events based on multimodal embeddings [105, 101]. In the remainder of this section, we describe the main technical novelty for each of these methods and summarize our contributions.

1.3.1 Cube Construction Task 1: Taxonomy Generation

As mentioned earlier, to turn the unstructured data into a multi-dimensional and multi-granular cube, the first central task is **taxonomy generation**. This task aims at automatically defining the schema for each cube dimension, by discovering the taxonomic structure



Modules	Tasks	Solutions
Part I: **Cube Construction**	**Taxonomy Construction:** How to find taxonomic structure for each dimension?	**TaxoGen** (Chapter 3): Unsupervised taxonomy construction with locally adaptive embeddings
	Document Allocation: How to allocate documents into the multidimensional cube?	**Doc2Cube** (Chapter 4): Unsupervised document allocation with dimension-aware embeddings
Part II: **Cube Exploitation**	**Multidimensional Prediction:** How to make predictions across different dimensions?	**ReAct** (Chapter 5): Semi-supervised multimodal embedding for online cross-dimension prediction
	Abnormal Event Detection: How to detect abnormal events in the cube space?	**TrioVecEvent** (Chapter 6): Weakly-supervised event detection with multimodal embeddings

Figure 1.3: An overview of our main algorithms for cube construction and exploitation. In the cube construction part, we propose (1) an unsupervised method (TaxoGen) that constructs topic taxonomy from a text corpus by learning locally adaptive embeddings; and (2) an unsupervised method (Doc2Cube) that performs multi-dimensional document classification without labeled data, by learning dimension-aware document representations. In the cube exploitation part, we propose (1) a semi-supervised method (CrossMap) that learns multimodal embeddings for cross-dimension prediction; and (2) a weakly-supervised method (TrioVecEvent) that detects abnormal events based on multimodal embeddings.

from text.

For this task, we propose TaxoGen [102], which organizes a given collection of concept terms into a topic taxonomy in an unsupervised way. To generate quality taxonomies, TaxoGen learns locally adaptive embeddings that achieve high discriminative power, and features an adaptive spherical clustering procedure that can assign terms to proper levels during a hierarchical clustering process. TaxoGen demonstrates, for the first time, that word embeddings can be exploited for topic taxonomy construction even without supervision. Compared with state-of-the-art hierarchical topic modeling methods, TaxoGen improves the parent-child relation accuracy from 27.2% to 77.5% and the topical coherency from 44.2% to 72.8%.

1.3.2 Cube Construction Task 2: Document Allocation

Following taxonomy generation, the second task of cube construction is **document allocation**, which aims at allocating documents into proper cells in the cube by choosing the most appropriate label along each dimension. Document allocation is essentially a multi-dimensional document classification task. But the key challenge that prevents existing text classification techniques from being applied is the lack of labeled training data.

For document allocation, we develop a method that requires only label surface names for multi-dimensional classification. Our method, named Doc2Cube [85], achieves unsupervised classification by learning *dimension-aware* embeddings for documents with a joint embedding framework. It iteratively learns dimension-aware document representations, by selectively focusing on discriminative terms for different dimensions and propagating the information from the seeds to other terms. Doc2Cube outperforms state-of-the-art weakly supervised methods by more than 35% in terms of F1 score.

1.3.3 Cube Exploitation Task 1: Cross-Dimension Prediction

In the cube exploitation part, we first study an essential problem of cube-based multi-dimensional analysis: can we predict the value of any one dimension given observations of other dimensions? Consider Figure 1.2 as an example. Using data from the cube cell ⟨Los Angeles, *, 2017 ⟩, can we predict where do *protest* usually occur, or what are the typical activities around *UCLA* at *8pm*?

In our study, we approach the **cross-dimension prediction** problem with multimodal embedding. Different from many existing studies based on latent variable models, our method CrossMap directly maps elements from different dimensions into a latent space. To learn high-quality multimodal embeddings, our method incorporates information from external knowledge sources (*e.g.*, Wikipedia, geographical gazetteer), by linking text with entity type information in such external knowledge sources and automatically labeling the linked records. This leads to a semi-supervised multimodal embedding framework, which leverages distant supervision to guide the embedding learning process. By instantiating our method for spatiotemporal activity prediction, we found that CrossMap leads to inspiring results: it outperforms existing latent variable models by more than 84% for activity prediction. Furthermore, the learned representations are broadly applicable and useful for downstream applications such as activity classification.

1.3.4 Cube Exploitation Task 2: Abnormal Event Detection

In the second subtask of cube exploitation, we study the abnormal event detection problem: given any ad-hoc cube cell, can we identify any abnormal events from the given instances? We focus on detecting spatiotemporal events, which represent abnormal patterns in a multi-dimensional topic-location-time space.

Existing event detection methods often require excessive human-curated training data to learn discriminative models for a set of specific event types. Further, they do not explicitly model the correlations among different modalities to uncover abnormal event in the multi-dimensional space. Our proposed TrioVecEvent method combines two powerful machine learning techniques: representation learning and latent variable model. The former can well encode the semantics of unstructured text, while the latter is good at expressing the complex structural correlations among different factors. TrioVecEvent combines the two with a novel Bayesian mixture model, which generates locations with Gaussian distributions and text embeddings with von-Mishes Fisher distributions. The Bayesian mixture model is able to cluster records into geo-topic clusters as candidate events, then true spatiotemporal events are identified with a concise set of features. TrioVecEvent improves the precision of state-of-the-art methods from 32% to 80% and the pseudo recall from 40% to 58% — such performance makes it feasible to be deployed for real-world event detection and alarm systems.

1.3.5 Summary of Contributions

Our main contributions in this thesis are summarized as follows:

- We propose an integral cube construction and exploitation framework for turning unstructured data into multi-dimensional knowledge. The cube construction module neatly organizes unstructured data into a cube structure, from which users can flexibly perform multi-dimensional, multi-granular exploration with declarative queries. The cube exploitation module offers algorithms to extract useful multi-dimensional knowledge in the cube space for task support and decision making.

- We propose unsupervised methods for cube construction, which learn task-specific embeddings and leverage the embeddings in an unsupervised way. Specifically, our proposed method TaxoGen learns locally adaptive embeddings and combine them with hierarchical spherical clustering for generating topical taxonomies; while our method Doc2Cube learns dimension-aware joint embeddings to perform document allocation without labeled data.

- We propose multimodal embedding techniques for extracting multi-dimensional knowledge in the cube exploitation part. Our method CrossMap is capable of making cross-dimension predictions in the cube space, by incorporating external knowledge into a semi-supervised multimodal embedding process. Based on multimodal embedding, our proposed method TrioVecEvent can detect abnormal events in the cube space, by combining the power of multimodal embedding and latent variable models.

1.4 ORGANIZATION

The remainder of the thesis is organized as follows. Chapter 2 to 5 describe our proposed algorithms for cube construction and exploitation. In the first part, we present algorithms for cube construction, introducing our methods for taxonomy generation (Chapter 2) and document classification (Chapter 3). In the second part, we present algorithms for cube exploitation, including our methods for cross-dimension prediction (Chapter 4) and abnormal event detection (Chapter 5). Finally, we conclude the thesis in Chapter 6 by summarizing our work and pointing out several future directions.

CHAPTER 2: TAXONOMY GENERATION FOR CUBE SCHEMA DISCOVERY

In the first part of the thesis, we describe the cube construction module, which organizes unstructured text data into a multi-dimensional, multi-granular cube structure, such that users can explore and retrieve relevant data with declarative queries easily. Recall Figure 1.2, the cube structure partitions the entire data space with multiple dimensions, each defined as an automatically discovered taxonomic structure. The documents in the entire corpus are allocated into one cell of the cube structure, thereby enabling users to select task-relevant data for on-demand analysis.

The cube construction process mainly involves two subtasks: (1) taxonomy generation—how to discover the taxonomic structure for each dimension? (2) document allocation—how to allocate all the documents into the cube by choosing the most appropriate label in each dimension? Answering these two questions is easy when the desired dimensions naturally inherit from explicit contexts associated with text, *e.g.*, meta-data such as patient age, tweet creating time, and GPS coordinates. However, when the dimensions are implicitly hidden in the text corpus and need to be inferred, these two tasks become nontrivial. In the following two chapters, we will describe unsupervised methods that discover hidden dimension schemas for the given corpus and allocate documents into the cube. Specifically, in this chapter, we introduce TaxoGen, an unsupervised method for taxonomy construction from text corpora; in next chapter, we introduce Doc2Cube, an unsupervised method for document allocation.

2.1 OVERVIEW

Taxonomy generation is essential to automatically defining the schema of a cube dimension. Given a collection of concept terms (*e.g.*, entities, noun phrases) related to a cube dimension, the task of taxonomy generation generally aims at at organizing the given terms into a concept hierarchy that reflect the parent-child relationships among these concepts. There are two major types of taxonomies in literature: *term-level* and *topic-level.* The former defines each node as a single term to represent a concept; and the latter defines each node as a group of topically coherent terms.

For the purpose of discovering the schema of each dimension, we focus on topic-level taxonomy construction. In contrast to term-level taxonomies, each node in our topic taxonomy is defined as a cluster of semantically coherent concept terms. This leads to a more concise taxonomy and less ambiguity of each node. Figure 2.1 shows a concrete example of topic taxonomy construction. Given a collection of computer science research papers, we build

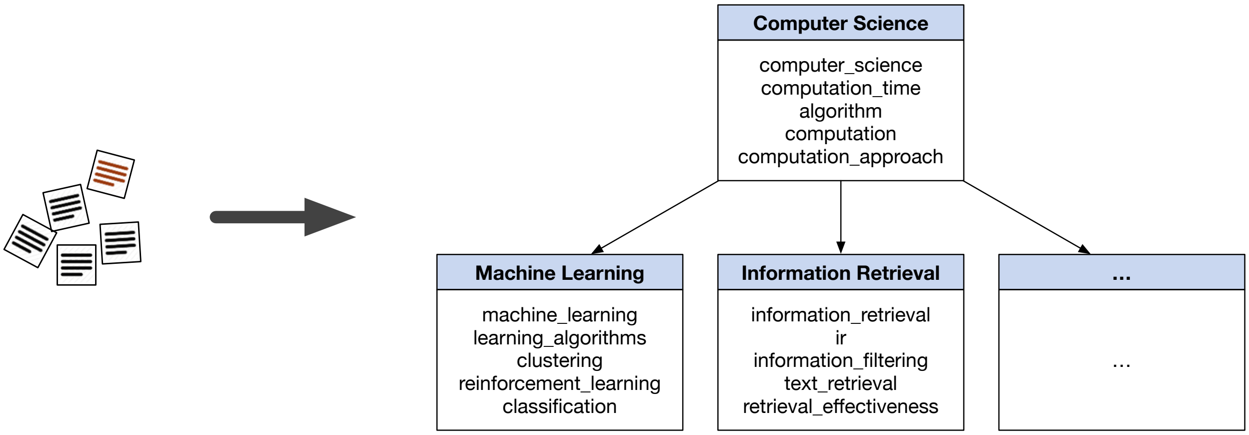


Figure 2.1: An illustration of topic taxonomy generation. Given a text corpus and a collection of concept terms, we aim to organize the concept terms into a topic taxonomy. Each node is a cluster of semantically coherent concept terms representing a conceptual topic.

a tree-structured hierarchy. The root node is the general topic 'computer science', which is further split into sub-topics like 'machine learning' and 'information retrieval'. For every topical node, we describe it with multiple concept terms that are semantically relevant. For instance, for the 'information retrieval' node, its associated terms include not only synonyms of 'information retrieval' (*e.g.*, 'ir'), but also different facets of the IR area (*e.g.*, 'text retrieval' and 'retrieval effectiveness').

Automatically organizing a set of concept terms into a topic hierarchy is not a trivial task. There have been many supervised learning methods for taxonomy construction [47, 45] in the natural language processing community. Basically these methods extract lexical features and learn a classifier that categorizes term pairs into relations or non-relations, based on curated training data of hypernym-hyponym pairs [95, 79, 19, 55], or syntactic contextual information harvested from NLP tools [94, 57]. However, these methods require excessive amount of training data and cannot be applied to cube construction in applications where curated pairs are unavailable. Along another line, hierarchical topic models [10, 64, 24] have been proposed to generate topic taxonomies in an unsupervised way. Nevertheless, these models rely on strong assumptions of document-topic and topic-term distributions, which can produce poor topic taxonomies when real data do not match well with such assumptions. Furthermore, the learning process of such hierarchical topic models is typically time-consuming, making them unscalable to large text corpora.

We propose an unsupervised method named TaxoGen for constructing topic taxonomies. It is based on the recent success of word embedding techniques [63] that encode text semantics with distributed representations. During the process of learning word embeddings, semantically relevant terms—which share similar contexts—tend to be pushed towards each other in the latent vector space. Take Figure 2.2 as a real-life example. After training the embeddings for computer science concept terms with a DBLP title corpus, one can observe the terms for the concepts 'computer graphics' and 'cryptography' are well clustered in the embedding space. The key idea behind TaxoGen is that: can we leverage such clustering structures of term embeddings to build topic taxonomies in a recursive way?

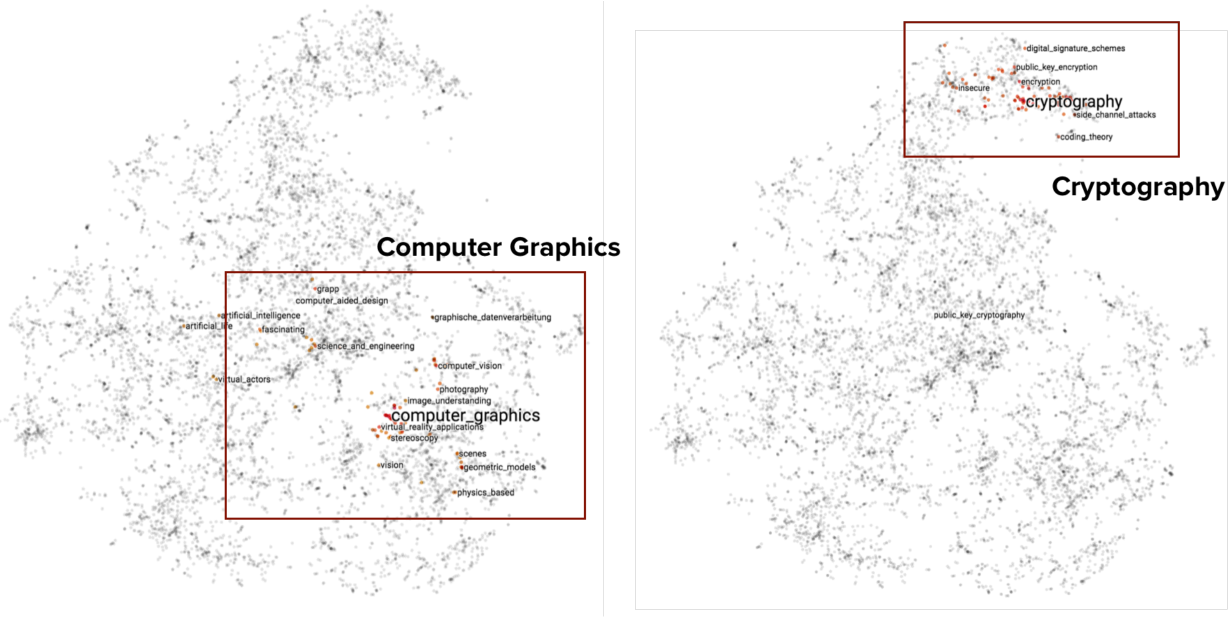


Figure 2.2: Visualizations of word embeddings trained on a DBLP corpus. Left: the embeddings of a set of terms in the 'computer graphics' area. Right: the embeddings of a set of terms in the 'cryptography' area.

While the idea of combining term embedding and hierarchical clustering is intuitive by itself, two key challenges need to be addressed for building high-quality taxonomies. First, *it is nontrivial to determine the proper granularity levels for different concept terms.* When splitting a coarse topical node into fine-grained ones, not all the concept terms should be pushed down to the child level. For example, when splitting the computer science topic in Figure 2.1, general terms like 'cs' and 'computer science' should remain in the parent instead of being allocated into any child topics. Therefore, it is problematic to directly group parent terms to form child topics, but necessary to allocate different terms to different levels. Second, *global embeddings have limited discriminative power at lower levels.* Term embeddings are typically learned by collecting the context evidence from the corpus, such that terms sharing similar contexts tend to have close embeddings. However, as we move down in the hierarchy, the term embeddings learned based on the entire corpus have limited power in capturing subtle semantics. For example, when splitting the machine learning topic, we find the terms 'machine learning' and 'reinforcement learning' have close global embeddings, and it is hard to discover quality sub-topics for the machine learning topic.

TaxoGen consists of two novel modules for tackling the above challenges. The first is an adaptive spherical clustering module for allocating terms to proper levels when splitting a coarse topic. Relying on a ranking function that measures the representativeness of different terms to each child topic, the clustering module iteratively detects general terms that should remain in the parent topic and keeps refining the clustering boundaries of the child topics. The second is a local term embedding module. To enhance the discriminative power of term embeddings at lower levels, TaxoGen uses topic-relevant documents to learn local embeddings for the terms in each topic. The local embeddings capture term semantics at a finer granularity and are less constrained by the terms irrelevant to the topic. As such, they are discriminative enough to separate the terms with different semantics even at lower levels of the taxonomy.

We perform extensive experiments on two real data sets. Our qualitative results show that TaxoGen can generate high-quality topic taxonomies, and our quantitative analysis based on user study shows that TaxoGen outperforms baseline methods significantly.

To summarize, our contributions in this chapter include:

1. We propose a method that combines word embeddings and hierarchical clustering for generating topic taxonomies in an unsupervised way. To the best of our knowledge, this is the first work that exploits word embeddings for topic taxonomy generation.

2. We propose an adaptive hierarchical spherical clustering procedure for topic splitting. It is capable of allocating terms into proper levels in the recursive taxonomy generation process.

3. We propose a local embedding module for learning locally adaptive term embeddings. Such locally adaptive term embeddings are tailored for topic taxonomy generation and can maintain strong discriminative power even at lower levels of the taxonomy;

4. We have performed extensive experiments on two real-life datasets. The results show that our TaxoGen outperforms state-of-the-art hierarchical topic models and hierarchical clustering methods both quantitatively and qualitatively.

2.2 RELATED WORK

In this section, we review existing taxonomy construction methods, including (1) supervised methods, (2) pattern-based methods, and (3) clustering-based methods.

2.2.1 Supervised Taxonomy Learning

Many existing taxonomy construction methods rely on the supervised learning paradigm [47, 45]. Basically these methods extract lexical features and learn a classifier that categorizes term pairs into relations or non-relations, based on curated training data of hypernym-hyponym pairs [95, 79, 19, 55], or syntactic contextual information harvested from NLP tools [94, 57]. Recent techniques [92, 98, 58, 27, 7] in this category leverage pre-trained word embeddings and then use curated hypernymy relation datasets to learn a relation classifier. However, the training data for all these methods are limited to extracting hypernym-hyponym relations and cannot be easily adapted for constructing a topic taxonomy. Furthermore, for massive domain-specific text data, it is hardly possible to collect a rich set of supervised information from experts. Therefore, we focus on technical developments in unsupervised taxonomy construction.

2.2.2 Pattern-Based Extraction

A considerable number of pattern-based methods have been proposed to construct hypernym-hyponym taxonomies wherein each node in the tree is an entity, and each parent-child pair expresses the “is-a” relation. Typically, these works first use pre-defined lexical patterns to extract hypernym-hyponym pairs from the corpus, and then organize all the extracted pairs into a taxonomy tree. In pioneering studies, Hearst patterns like “NP such as NP, NP, and NP” were proposed to automatically acquire hyponymy relations from text data [37]. Then more kinds of lexical patterns have been manually designed and used to extract relations from the web corpus [78, 68] or Wikipedia [70, 32]. With the development of the Snowball framework, researchers teach machines how to propagate knowledge among the massive text corpora using statistical approaches [5, 111]; Carlson et al. proposed a learning architecture for Never-Ending Language Learning (NELL) in 2010 [13]. PATTY leveraged parsing structures to derive relational patterns with semantic types and organizes the patterns into a taxonomy [66]. The recent MetaPAD [40] used context-aware phrasal segmentation to generate quality patterns and group synonymous patterns together for a large collection of facts. Pattern-based methods have demonstrated their effectiveness in finding particular relations based on hand-crafted rules or generated patterns. However, they are not suitable for constructing a topic taxonomy because of two reasons. First, different from hypernym-hyponym taxonomies, each node in a topic taxonomy can be a group of terms representing a conceptual topic. Second, pattern-based methods often suffer from low recall due to the large variation of expressions in natural language on parent-child relations.

2.2.3 Clustering-Based Taxonomy Construction

Clustering methods have been proposed for constructing taxonomy from text corpus [8, 20, 89, 87, 58, 27]. These methods are more closely related to our problem of constructing a topic taxonomy. Generally, the clustering approaches first learn the representation of words or terms and then organize them into a structure based on their representation similarity [8] and cluster separation measures [20]. Fu et al. identified whether a candidate word pair has hypernym-hyponym ("is-a") relation by using the word-embedding-based semantic projections between words and their hypernyms [27]. Luu et al. proposed to use dynamic weighting neural network to identify taxonomic relations via learning term embeddings [58]. Our *local term embedding* in TaxoGen is quite different from the existing methods. First, we do not need labeled hypernym-hyponym pairs as supervision for learning either semantic projections or dynamic weighting neural network. Second, we learn local embeddings for each topic using only topic-relevant documents. The local embeddings capture fine-grained term semantics and thus well separate terms with subtle semantic differences. On the term organizing end, Ciniano et al. used a comparative measure to perform conceptual, divisive, and agglomerative clustering for taxonomy learning [18]. Yang et al. also used an ontology metric, a score indicating semantic distance, to induce taxonomy [95]. Liu et al. used Bayesian rose tree to hierarchically cluster a given set of keywords into a taxonomy [55]. Wang et al. adopted a recursive way to construct topic hierarchies by clustering domain keyphrases [89, 87]. Also, quite a number of hierarchical topic models have been proposed for term organization [10, 64, 24]. In our TaxoGen, we develop an *adaptive spherical clustering* module to allocate terms into proper levels when we split a coarse topic. The module well groups terms of the same topic together and separates child topics (as term clusters) with significant distances.

2.3 PRELIMINARIES

2.3.1 Problem Definition

The input for constructing a topic taxonomy includes two parts: (1) a corpus $\mathcal{D}$ of documents; and (2) a set $\mathcal{T}$ of concept terms related to a dimension. The terms in $\mathcal{T}$ are the key terms extracted from $\mathcal{D}$, representing the terms of interest for taxonomy construction. The term set can be either specified by end users or extracted from the corpus. For example, they can be all the named entities related to the dimension of interest extracted from $\mathcal{D}$.

Given the corpus $\mathcal{D}$ and the term set $\mathcal{T}$, we aim to build a tree-structured hierarchy $\mathcal{H}$. Each node $C \in \mathcal{H}$ denotes a conceptual topic, which is described by a set of terms $\mathcal{T}_C \in \mathcal{T}$ that are semantically coherent. Suppose a node C has a set of children $\mathcal{S}_C = \{S_1, S_2, \ldots, S_N\}$, then each $S_n (1 \leq n \leq N)$ should be a sub-topic of C, and have the same semantic granularity with its siblings in $\mathcal{S}_C$. Each parent-child pair $\langle C, S_n \rangle$ represents a semantically subsuming relationship. That is, anything semantically related to the child topic S_n should be related to the parent C.

2.3.2 TaxoGen: Unsupervised Taxonomy Generation with Term Embeddings

In a nutshell, TaxoGen embeds all the concept terms into a latent space to capture their semantics, and uses the term embeddings to build the taxonomy recursively. As shown in Figure 2.3, at the top level, we initialize a root node containing all the terms from $\mathcal{T}$, which represents the most general topic for the given corpus $\mathcal{D}$. Starting from the root node, we generate fine-grained topics level by level via top-down spherical clustering. The top-down construction process continues until a maximum number of levels L_{max} is reached.

Given a topic C, we use spherical clustering to split C into a set of fine-grained topics $\mathcal{S}_C = \{S_1, S_2, \ldots, S_N\}$. As mentioned earlier, there are two challenges that need to be addressed in the resursive construction process: (1) when splitting a topic C, it is problematic to directly divide the terms in C into sub-topics, because general terms should remain in the parent topic C instead of being allocated to any sub-topics; (2) when we move down to lower levels, global term embeddings learned on the entire corpus are inadequate for capturing subtle term semantics. In the following, we introduce the adaptive clustering and local embedding modules in TaxoGen for addressing these two challenges.

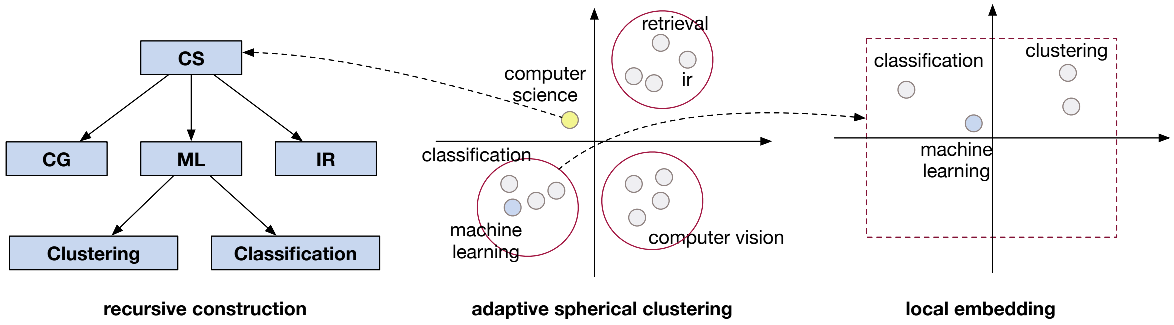


Figure 2.3: An overview of TaxoGen. It uses term embeddings to construct the taxonomy in a top-down manner, with two novel components for ensuring the quality of the resursive process: (1) an adaptive clustering module that allocates terms to proper topic nodes; and (2) a local embedding module for learning term embeddings on topic-relevant documents.

2.4 ADAPTIVE TERM CLUSTERING

The adaptive clustering module in TaxoGen is designed to split a coarse topic C into fine-grained ones. It is based on the spherical K-means algorithm [21], which groups a given set of term embeddings into K clusters such that the terms in the same cluster have similar embedding directions. Our choice of the spherical K-means algorithm is motivated by the effectiveness of the cosine similarity [63] in quantifying the similarities between word embeddings. The center direction of a topic acts as a semantic focus on the unit sphere, and the member terms of that topic falls around the center direction to represent a coherent semantic meaning.

2.4.1 Spherical Clustering for Topic Splitting

Given a coarse topic C, a straightforward idea for generating the sub-topics of C is to directly apply spherical K-means to C, such that the terms in C are grouped into K clusters to form C's sub-topics. Nevertheless, such a straightforward strategy is problematic because not all the terms in C should be allocated into the child topics. For example, in Figure 2.3, when splitting the root topic of computer science, terms like 'computer science' and 'cs' are general — they do not belong to any specific child topics but instead should remain in the parent. Furthermore, the existence of such general terms makes the clustering process more challenging. As such general terms can co-occur with various contexts in the corpus, their embeddings tend to fall on the boundaries of different sub-topics. Thus, the clustering structure for the sub-topics is blurred, making it harder to discover clear sub-topics.

Motivated by the above, we propose *an adaptive clustering module* in TaxoGen. As shown in Figure 2.3, the key idea is to iteratively identify general terms and refine the sub-topics after pushing general terms back to the parent. Identifying general terms and refining child topics are two operations that can mutually enhance each other: excluding the general terms in the clustering process can make the boundaries of the sub-topics clearer; while the refined sub-topics boundaries enable detecting additional general terms.

Algorithm 2.1 shows the process for adaptive spherical clustering. As shown, given a parent topic C, it first puts all the terms of C into the sub-topic term set C_{sub}. Then it iteratively identifies general terms and refines the sub-topics. In each iteration, it computes the representativeness score of a term t for the sub-topic S_k, and excludes t if its representativeness is smaller than a threshold δ. After pushing up general terms, it re-forms the sub-topic term set C_{sub} and prepares for the next spherical clustering operation. The iterative process terminates when no more general terms can be detected, and the final set of sub-topics $S_1, S_2, \ldots, S_K$ are returned.

Algorithm 2.1: Adaptive clustering for topic splitting.

```
Input: A parent topic $C$; the number of sub-topics $K$; the term representativeness
       threshold $\delta$.
Output: $K$ sub-topics of $C$.
$C_{sub} \leftarrow C$;
while True do
    $S_1, S_2, \ldots, S_K \leftarrow$ SPHERICAL-KMEANS($C_{sub}, K$);
    for $k$ from 1 to $K$ do
        for $t \in S_k$ do
            $r(t, S_k) \leftarrow$ representativeness of term $t$ for $S_k$;
            if $r(t, S_k) < \delta$ then
                $S_k \leftarrow S_k - \{t\}$;
    $C'_{sub} \leftarrow S_1 \cup S_2 \cup \ldots \cup S_K$;
    if $C'_{sub} = C_{sub}$ then
        Break;
    $C_{sub} \leftarrow C'_{sub}$;
Return $S_1, S_2, \ldots, S_K$;
```

2.4.2 Identifying Representative Terms

In Algorithm 2.1, the key question is how to measure the representativeness of a term t for a sub-topic S_k. While it is tempting to measure the representativeness of t by its closeness to the center of S_k in the embedding space, we find such a strategy is unreliable: general terms may also fall close to the cluster center of S_k, which renders the embedding-based detector inaccurate.

Our insight for addressing this problem is that, a representative term for S_k should appear frequently in S_k but not in the sibling topics of S_k. We hence measure term representativeness using the documents that belong to S_k. Based on the cluster memberships of terms, we first use the TF-IDF scheme to obtain the documents belonging to each topic S_k. With these S_k-related documents, we consider the following two factors for computing the representativeness of a term t for topic S_k:

- **Popularity**: A representative term for S_k should appear frequently in the documents of S_k.
- **Concentration**: A representative term for S_k should be much more relevant to S_k compared to the sibling topics of S_k.

To combine the above two factors, we notice that they should have conjunctive conditions, namely a representative term should be both popular and concentrated for S_k. Thus we define the representativeness of term t for topic S_k as

$$r(t, S_k) = \sqrt{pop(t, S_k) \cdot con(t, S_k)} \tag{2.1}$$

where $pop(t, S_k)$ and $con(t, S_k)$ are the popularity and concentration scores of t for S_k. Let $\mathcal{D}_k$ denotes the documents belonging to S_k, we define $pop(t, S_k)$ as the normalized frequency of t in $\mathcal{D}_k$:

$$pop(t, S_k) = \frac{\log(tf(t, \mathcal{D}_k) + 1)}{\log tf(\mathcal{D}_k)},$$

where $tf(t, \mathcal{D}_k)$ is number of occurrences of term t in $\mathcal{D}_k$, and $tf(\mathcal{D}_k)$ is the total number of tokens in $\mathcal{D}_k$.

To compute the concentration score, we first form a pseudo document D_k for each subtopic S_k by concatenating all the documents in $\mathcal{D}_k$. Then we define the concentration of term t on S_k based on its relevance to the pseudo document D_k:

$$con(t, S_k) = \frac{\exp(rel(t, D_k))}{1 + \sum\limits_{1 \le j \le K} \exp(rel(t, D_j))},$$

where $rel(p, D_k)$ is the BM25 relevance of term t to the pseudo document D_k.

Example 2.1. *Figure 2.3 shows the adaptive clustering process for splitting the computer science topic into three sub-topics: computer graphics (CG), machine learning (ML), and information retrieval (IR). Given a sub-topic, for example ML, terms (*e.g.*, 'clustering', 'classificiation') that are popular and concentrated in this cluster receive high representativeness scores. In contrast, terms (*e.g.*, 'computer science') that are not representative for any sub-topics are considered as general terms and pushed back to the parent.*

2.5 ADAPTIVE TERM EMBEDDING

2.5.1 Distributed Term Representations

The recursive taxonomy construction process of TaxoGen relies on term embeddings, which encode term semantics by learning fixed-size vector representations for the terms. We use the SkipGram model [63] for learning term embeddings. Given a corpus, SkipGram models the relationship between a term and its context terms in a sliding window, such that the terms that share similar contexts tend to have close embeddings in the latent space. The result embeddings can well capture the semantics of different terms and been demonstrated useful for various NLP tasks.

Formally, given a corpus $\mathcal{D}$, for any token t, we consider a sliding window centered at t and use W_t to denote the tokens appearing in the context window. Then we define the log-probability of observing the contextual terms as

$$\log p(W_t|t) = \sum_{w \in W_t} \log p(w|t) = \sum_{w \in W_t} \log \frac{\mathbf{v}_t \mathbf{v}'_w}{\sum\limits_{w' \in V} \mathbf{v}_t \mathbf{v}'_{w'}}$$

where $\mathbf{v}_t$ is the embedding for term t, $\mathbf{v}'_w$ is the contextual embedding for the term w, and V is the vocabulary of the corpus $\mathcal{D}$. Then the overall objective function of SkipGram is defined over all the tokens in $\mathcal{D}$, namely

$$L = \sum_{t \in \mathcal{D}} \sum_{w \in W_t} \log p(w|t),$$

and the term embeddings can be learned by maximizing the objective with stochastic gradient descent and negative sampling [63].

2.5.2 Learning Local Term Embeddings

However, when we use the term embeddings trained on the entire corpus $\mathcal{D}$ for taxonomy construction, one drawback is that these global embeddings have limited discriminative power at lower levels. Let us consider the term 'reinforcement learning' in Figure 2.3. In the entire corpus $\mathcal{D}$, it shares a lot of similar contexts with the term 'machine learning', and thus has an embedding close to 'machine learning' in the latent space. The proximity with 'machine learning' makes it successfully assigned into the machine learning topic when we are splitting the root topic. Nevertheless, as we move down to split the machine learning topic, the embeddings of 'reinforcement learning' and other machine learning terms are entangled together, making it difficult to discover sub-topics for machine learning.

Therefore, we propose the local embedding module to enhance the discriminative power of term embeddings at lower levels of the taxonomy. For any topic C that is not the root topic, we learn local term embeddings for splitting C. Specifically, we first create a sub-corpus $\mathcal{D}_C$ from $\mathcal{D}$ that is relevant to the topic C. To obtain the sub-corpus $\mathcal{D}_C$, we employ the following two strategies: (1) *Clustering-based.* We derive the cluster membership of each document $d \in \mathcal{D}$ by aggregating the cluster memberships of the terms in d using TF-IDF weight. The documents that are clustered into topic C are collected to form the sub-corpus $\mathcal{D}_C$. (2) *Retrieval-based.* We compute the embedding of any document $d \in \mathcal{D}$ using TF-IDF weighted average of the term embeddings in d. Based on the obtained document embeddings, we use the mean direction of the topic C as a query vector to retrieve the top-M closest documents and form the sub-corpus $\mathcal{D}_C$. In practice, we use the first strategy as the main one to obtain $\mathcal{D}_C$, and apply the second strategy for expansion if the clustering-based subcorpus is not large enough. Once the sub-corpus $\mathcal{D}_C$ is retrieved, we apply the SkipGram model to the sub-corpus $\mathcal{D}_C$ to obtain term embeddings that are tailored for splitting the topic C.

Example 2.2. *Consider Figure 2.3 as an example, when splitting the machine learning topic, we first obtain a sub-corpus $\mathcal{D}_{ml}$ that is relevant to machine learning. Within $\mathcal{D}_{ml}$, terms reflecting general machine learning topics such as 'machine learning' and 'ml' appear in a large number of documents. They become similar to stopwords and can be easily separated from more specific terms. Meanwhile, for those terms that reflect different machine learning sub-topics (*e.g.*, 'classifcation' and 'clustering'), they are also better separated in the local embedding space. Since the local embeddings are trained to preserve the semantic information for topic-related documents, different terms have more freedom to span in the embedding space to reflect their subtle semantic differences.*

2.6 EXPERIMENTAL EVALUATION

In this section, we evaluate the empirical performance of TaxoGen.

2.6.1 Experimental Setup

Datasets

We use two real-life corpora in our experiments[1]:

1. DBLP contains around 1,889,656 titles of computer science papers from the areas of information retrieval, computer vision, robotics, security & network, and machine learning. From those paper titles, we use an existing NP chunker to extract all the noun phrases and then remove infrequent ones to form the term set, resulting in 13,345 distinct terms;

2. SP contains 94,476 paper abstracts from the area of signal processing. Similarly, we extract all the noun phrases in those abstracts to form the term set and obtain 6,982 different terms.

Compared Methods

We compare TaxoGen with the following methods that are capable of generating topic taxonomies in an unsupervised way:

1. HLDA (hierarchical Latent Dirichlet Allocation) [10] is a non-parametric hierarchical topic model. It models the probability of generating a document as choosing a path from the root to a leaf and sampling words along the path. We apply HLDA for topic-level taxonomy construction by regarding each topic in HLDA as a topic.

2. HPAM (hierarchical Pachinko Allocation Model) is a state-of-the-art hierarchical topic model [64]. Different from TaxoGen that generates the taxonomy recursively, HPAM takes all the documents as its input and outputs a pre-defined number of topics at different levels based on the Pachinko Allocation Model.

3. HClus (hierarchical clustering) uses hierarchical clustering for taxonomy construction. We first apply the SkipGram model on the entire corpus to learn term embeddings, and then use spherical k-means to cluster those embeddings in a top-down manner.

[1] The code and data are available at https://github.com/franticnerd/taxogen/.

4. NoAC is a variant of TaxoGen without the adaptive clustering module. In other words, when splitting one coarse topic into fine-grained ones, it simply performs spherical clustering to group parent terms into child topics.

5. NoLE is a variant of TaxoGen without the local embedding module. During the recursive construction process, it uses the global embeddings that are learned on the entire corpus throughout the construction process.

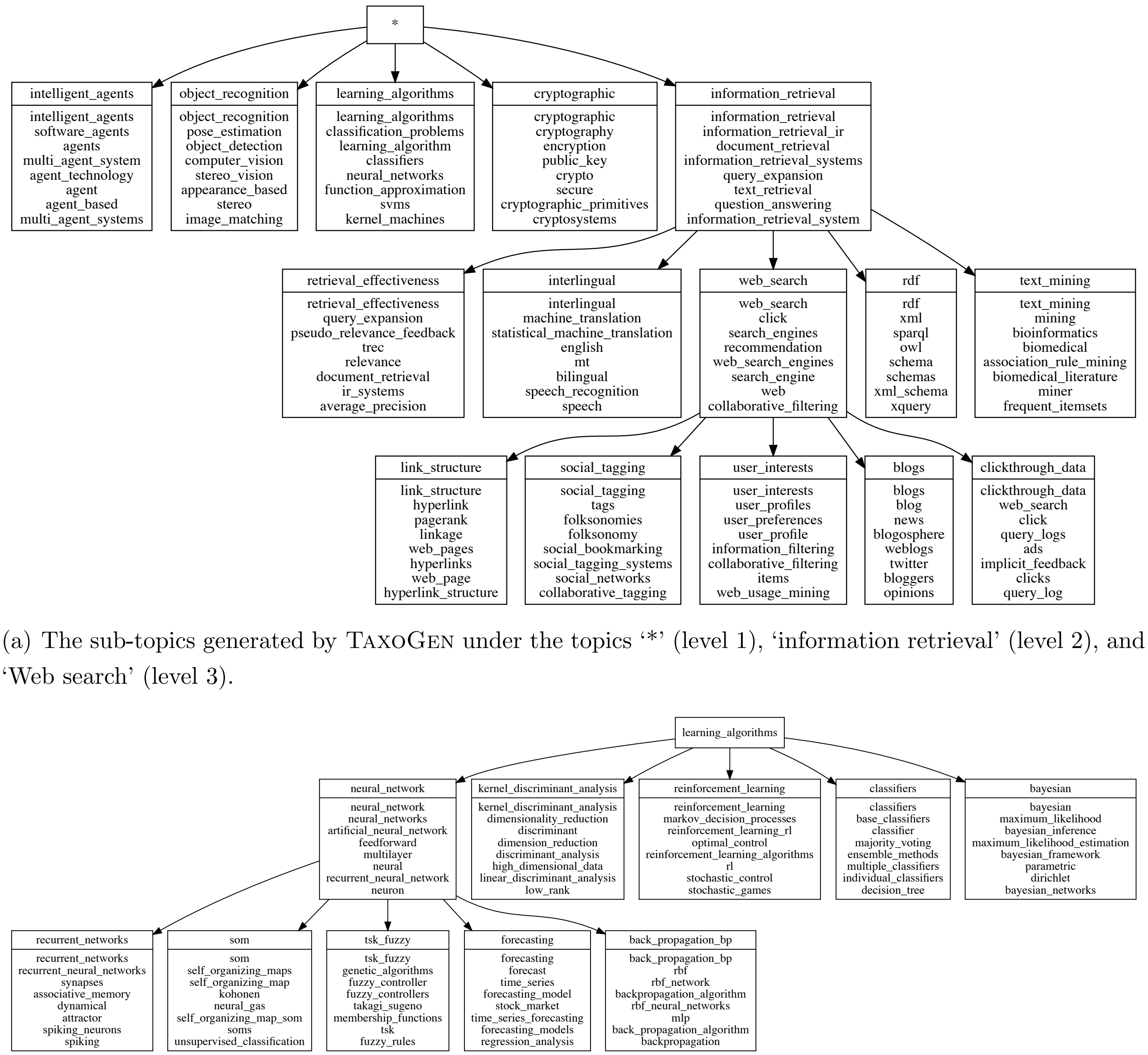
Parameter Settings

We use the methods to generate a four-level taxonomy on DBLP and a three-level taxonomy on SP. There are two key parameters in TaxoGen: the number K for splitting a coarse topic and the representativeness threshold δ for identifying general terms. We set $K = 5$ as we found such a setting matches the intrinsic taxonomy structures well on both DBLP and SP. For δ, we set it to 0.25 on DBLP and 0.15 on SP after tuning, because we observed such a setting can robustly detect general terms that belong to parent topics at different levels in the construction process.

For HLDA, it involves three hyper-parameters: (1) the smoothing parameter α over level distributions; (2) the smoothing parameter γ for the Chinese Restaurant Process; and (3) the smoothing parameter η over topic-word distributions. We set $\alpha = 0.1, \gamma = 1.0, \eta = 1.0$. Under such a setting, HLDA generates a comparable number of topics with TaxoGen on both datasets. The method HPAM requires to set the mixture priors for super- and subtopics. We find that the best values for these two priors are 1.5 and 1.0 on DBLP and SP, respectively. The remaining three methods (HClus, NoAC, and NoLE) have a subset of the parameters of TaxoGen, and we set them to the same values as TaxoGen.

2.6.2 Qualitative Results

In this subsection, we demonstrate the topic taxonomies generated by different methods on DBLP. We apply each method to generate a four-level taxonomy on DBLP, and each parent topic is split into five child topics by default (except for HLDA, which automatically determines the number of child topics based on the Chinese Restaurant Process).



(a) The sub-topics generated by TaxoGen under the topics '*' (level 1), 'information retrieval' (level 2), and 'Web search' (level 3).

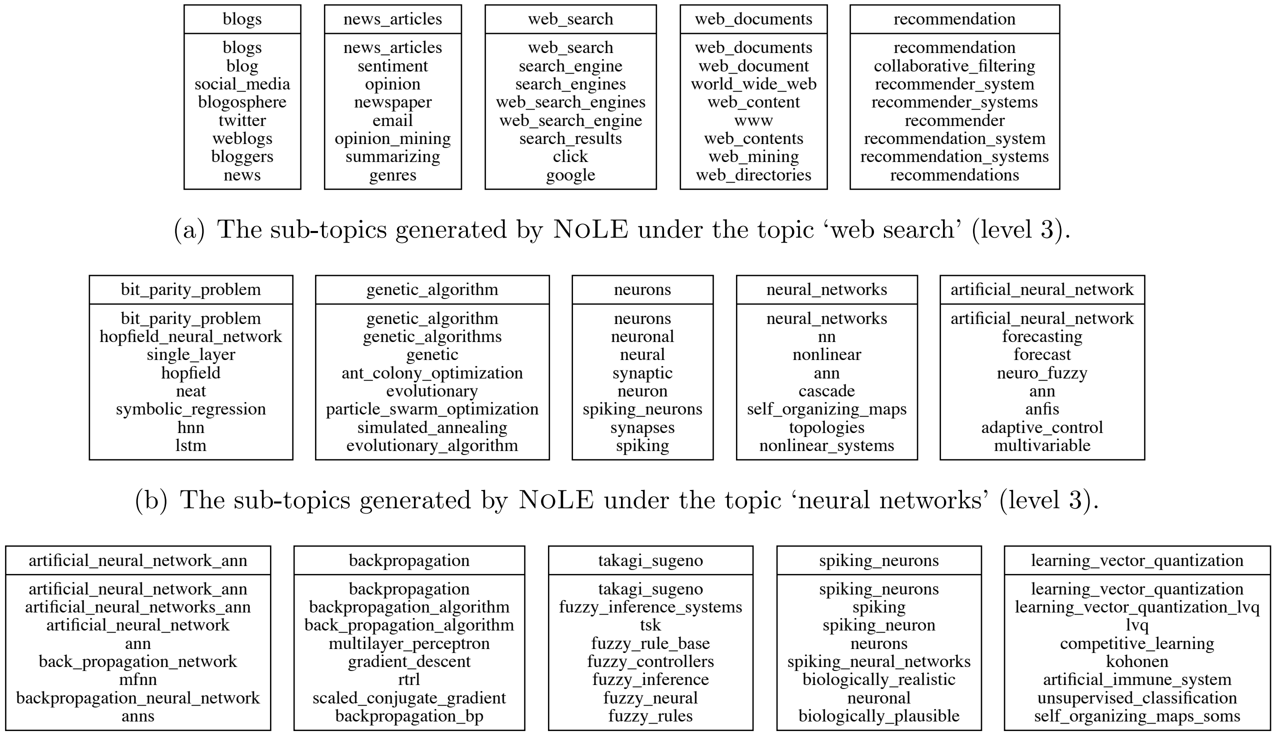
(b) The sub-topics generated by TaxoGen under the topics 'learning algorithms' (level 2) and 'neural network' (level 3).

Figure 2.4: Parts of the taxonomy generated by TaxoGen on the DBLP dataset. For each topic, we show its label and the top-eight representative terms generated by the ranking function of TaxoGen. All the labels and terms are returned by TaxoGen automatically without manual selection or filtering.

Figure 2.4 shows parts of the taxonomy generated by TaxoGen. As shown in Figure 2.4(a), given the DBLP corpus, TaxoGen splits the root topic into five sub-topics: 'intelligent agents', 'object recognition', 'learning algorithms', 'cryptographic', and 'information retrieval'. The labels for those topics are generated automatically by selecting the term that is most representative for a topic (Equation 2.1). We find those labels are of good quality and precisely summarize the major research areas covered by the DBLP corpus. The only minor flaw for the five labels is 'object recognition', which is too specific for the computer vision area. The reason is probably because the term 'object recognition' is too popular in the titles of computer vision papers, thus attracting the center of the spherical cluster towards itself.

In Figure 2.4(a) and 2.4(b), we also show how TaxoGen splits level-two topics 'information retrieval' and 'learning algorithms' into more fine-grained topics. Taking 'information retrieval' as an example: (1) at level three, TaxoGen can successfully find major areas in information retrieval: retrieval effectiveness, interlingual, Web search, rdf & xml query, and text mining; (2) at level four, TaxoGen splits the Web search topic into more fine-grained problems: link analysis, social tagging, recommender systems & user profiling, blog search, and clickthrough models. Similarly for the machine learning topic (Figure 2.4(b)), TaxoGen can discover level-three topics like 'neural network' and level-four topic like 'recurrent neural network'. Moreover, the top terms for each topic are of good quality — they are semantically coherent and cover different aspects and expressions of the same topic.

We have also compared the taxonomies generated by TaxoGen and other baseline methods, and found that TaxoGen offers clearly better taxonomies from the qualitative perspective. Due to the space limit, we only show parts of the taxonomies generated by NoAC and NoLE to demonstrate the effectiveness of TaxoGen. As shown in Figure 2.5(a), NoLE can also find several sensible child topics for the parent topic (*e.g.*, 'blogs' and 'recommender system' under 'Web search'), but the major disadvantage is that a considerable number of the child topics are false positives. Specifically, a number of parent-child pairs ('web search' and 'web search', 'neural networks' and 'neural networks') actually represent the same topic instead of true hypernym-hyponym relations. The reason behind is that NoLE uses global term embeddings at all levels, and thus the terms for different semantic granularities have close embeddings and hard to be separated at lower levels. Such a problem also exists for NoAC, but with a different reason: NoAC does not leverage adaptive clustering to push up the terms that belong to the parent topic. Consequently, at fine-grained levels, terms that have different granularities are all involved in the clustering step, making the clustering boundaries less clear compared to TaxoGen. Such qualitative results clearly show the advantages of TaxoGen over the baseline methods, which are the key factors that leads to the performance gaps between them in our quantitative evaluation.



(a) The sub-topics generated by NoLE under the topic 'web search' (level 3).

(b) The sub-topics generated by NoLE under the topic 'neural networks' (level 3).

(c) The sub-topics generated by NoAC under the topic 'neural network' (level 3).

Figure 2.5: Example topics generated by NoLE and NoAC on the DBLP dataset. Again, we show the label and the top-eight representative terms for each topic.

Table 2.1 further compares global and local term embeddings for similarity search tasks. As shown, for the given two queries, the top-five terms retrieved with global embeddings (*i.e.*, the embeddings trained on the entire corpus) are relevant to the queries, yet they are semantically dissimilar if we inspect them at a finer granularity. For example, for the query 'information extraction', the top-five similar terms cover various areas and semantic granularities in the NLP area, such as 'text mining', 'named entity recognition', and 'natural language processing'. In contrast, the results returned based on local embeddings are more coherent and of the same semantic granularity as the given query.

2.6.3 Quantitative Analysis

In this subsection, we quantitatively evaluate the quality of the constructed topic taxonomies by different methods. The evaluation of a taxonomy is a challenging task, not only because there are no ground-truth taxonomies for our used datasets, but also that the quality of a taxonomy should be judged from different aspects. In our study, we consider the

Table 2.1: Similarity searches on DBLP for: (1) Q1 = 'pose_estimation'; and (2) Q2 = 'information_extraction'. For both queries, we use cosine similarity to retrieve the top-five terms in the vocabulary based on global and local embeddings. The local embedding results for 'pose_estimation' are obtained in the 'object_recognition' sub-topic, while the results for 'information_extraction' are obtained in the 'learning_algorithms' sub-topic.

Query	**Global Embedding**	**Local Embedding**
Q1	pose_estimation	pose_estimation
	single_camera	camera_pose_estimation
	monocular	dof
	d_reconstruction	dof_pose_estimation
	visual_servoing	uncalibrated
Q2	information_extraction	information_extraction
	information_extraction_ie	information_extraction_ie
	text_mining	ie
	named_entity_recognition	extracting_information_from
	natural_language_processing	question_anwering_qa

following aspects for evaluating a topic-level taxonomy:

- **Relation Accuracy** aims at measuring the portions of the true positive parent-child relations in a given taxonomy.
- **Term Coherency** aims at quantifying how semantically coherent the top terms are for a topic.
- **Cluster Quality** examines whether a topic and its siblings form quality clustering structures that are well separated in the semantic space.

We instantiate the evaluations of the above three aspects as follows. First, for the relation accuracy measure, we take all the parent-child pairs in a taxonomy and perform user study to judge these pairs. Specifically, we recruited 10 doctoral and post-doctoral researchers in Computer Science as human evaluators. For each parent-child pair, we show the parent and child topics (in the form of top-five representative terms) to at least three evaluators, and ask whether the given pair is a valid parent-child relation. After collecting the answers from the evaluators, we simply use majority voting to label the pairs and compute the ratio of true positives. Second, to measure term coherency, we perform a term intrusion user study. Given the top five terms for a topic, we inject into these terms a fake term that is randomly chosen from a sibling topic. Subsequently, we show these six terms to an evaluator and ask which one is the injected term. Intuitively, the more coherent the top terms are, the more likely an evaluator can correctly identify the injected term, and thus we compute the ratio of correct instances as the term coherency score. Finally, to quantify cluster quality, we use the Davies-Bouldin (DB) Index measure: For any cluster C, we first compute the similarities between C and other clusters and assign the largest value to C as its cluster similarity. Then the DB index is obtained by averaging all the cluster similarities [20]. The smaller the DB index is, the better the clustering result is.

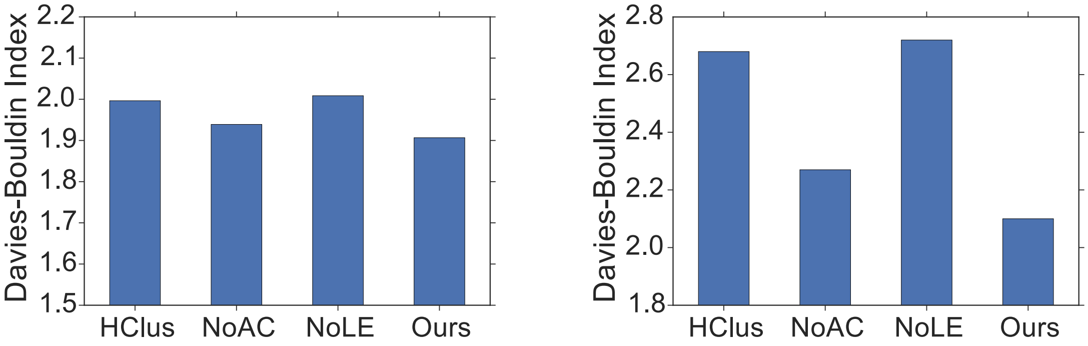
Table 2.2 shows the relation accuracy and term coherency of different methods. As shown, TaxoGen achieves the best performance in terms of both measures. TaxoGen significantly outperforms topic modeling methods as well as other embedding-based baseline methods. Comparing the performance of TaxoGen, NoAC, and NoLE, we can see both the adaptive clustering and the local embedding modules play an important role in improving the quality of the result taxonomy: the adaptive clustering module can correctly push background terms back to parent topics; while the local embedding strategy can better capture subtle semantic differences of terms at lower levels. For both measures, the topic modeling methods (HLDA and HPAM) perform significantly worse than embedding-based methods, especially on the short-document dataset DBLP. The reason is two-fold. First, HLDA and HPAM make stronger assumptions on document-topic and topic-term distributions, which may not fit the empirical data well. Second, the representative terms of topic modeling methods are selected purely based on the learned multinomial distributions, whereas embedding-based methods perform distinctness analysis to select terms that are more representative.

Table 2.2: Relation accuracy and term coherency of different methods on the DBLP and SP datasets.

	Relation Accuracy		Term Coherency	
Method	DBLP	SP	DBLP	SP
HPAM	0.109	0.160	0.173	0.163
HLDA	0.272	0.383	0.442	0.265
HClus	0.436	0.240	0.467	0.571
NoAC	0.563	0.208	0.35	0.428
NoLE	0.645	0.240	0.704	0.510
TaxoGen	**0.775**	**0.520**	**0.728**	**0.592**

Figure 2.6 shows the DB index of all the embedding-based methods. TaxoGen achieves the smallest DB index (the best clustering result) among these four methods. Such a phenomenon further validates the fact that both the adaptive clustering and local embedding

modules are useful in producing clearer clustering structures: (1) The adaptive clustering process gradually identifies and eliminates the general terms, which typically lie in the boundaries of different clusters; (2) The local embedding module is capable of refining term embeddings using a topic-constrained sub-corpus, allowing the sub-topics to be well separated from each other at a finer granularity.



(a) DB index on DBLP.

(b) DB index on SP.

Figure 2.6: The Davies-Bouldin index of embedding-based methods on DBLP and SP.

2.7 SUMMARY

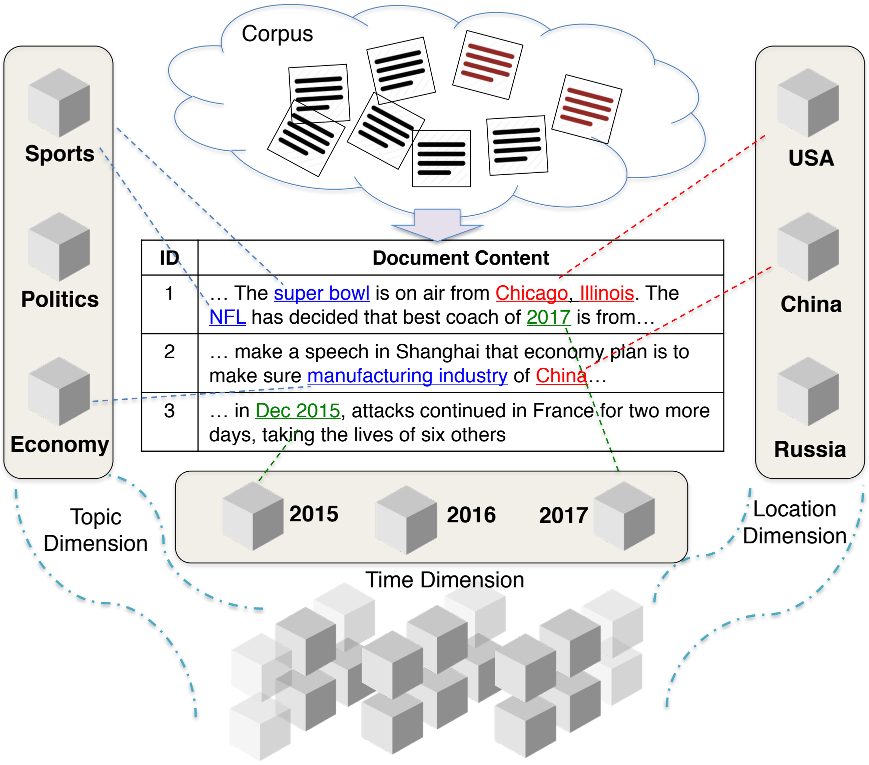
In this chapter, we studied the problem of constructing topic taxonomies from text, which can serve as an essential ingredient for defining the schema for each cube dimension. Our proposed method TaxoGen relies on term embedding and spherical clustering to construct a topic taxonomy in a recursive way. It consists of an adaptive clustering module that allocates terms to proper levels when splitting a coarse topic, as well as a local embedding module that learns term embeddings to maintain strong discriminative power at lower levels. In our experiments, we have demonstrated that both two modules are useful in improving the quality of the resultant taxonomy, which renders TaxoGen advantages over state-of-the-art hierarchical topic models and hierarchical clustering methods for topic taxonomy construction.

CHAPTER 3: DOCUMENT ALLOCATION WITHOUT LABELED DATA

In the previous chapter, we studied the taxonomy generation problem for discovering the schema of text cube. In this chapter we proceed to study the document allocation problem, which aims at allocating the documents into the multi-dimensional cube.

3.1 OVERVIEW

Given a text corpus $\mathcal{D}$ and a defined cube schema $\mathcal{C}$, the document allocation task aims to allocate the documents in $\mathcal{D}$ into the right cells in $\mathcal{C}$. Figure 3.1 shows an example on a news corpus. Let $\mathcal{C}$ be a pre-defined cube schema with three dimensions: topic, location, and time. The document allocation task is to assign each news article in the given corpus into a proper cube cell (*e.g.*, ⟨Sports, 2017, USA⟩), by choosing one label along each dimension to best match the textual content of the article. Obviously, document allocation is an indispensible step for organizing unstructured text documents into a multi-dimensional cube structure.



ID	Document Content
1	… The super bowl is on air from Chicago, Illinois. The NFL has decided that best coach of 2017 is from…
2	… make a speech in Shanghai that economy plan is to make sure manufacturing industry of China…
3	… in Dec 2015, attacks continued in France for two more days, taking the lives of six others

Figure 3.1: Text cube construction on a news corpus with three dimensions: topic, location and time. Each document needs to be assigned with one label in each of the three dimensions.

Document allocation is a multi-dimensional categorization problem in nature and closely related to document classification [4, 77, 96, 82]. Nevertheless, it has two unique challenges that prevent existing document classification methods from being applied: (1) The first challenge is *the lack of labeled training data.* The success of prevailing document classification methods largely relies on sufficient labeled document-label pairs to train reliable classifiers. For text cube construction, it is costly to manually annotate a large number of documents for classification, given that every document has to be assigned with multiple labels; (2) The second is to *extract discriminative features for different dimensions.* Existing document classification methods typically extract a set of lexical features, or learn distributed representations for textual units (words, sentences, or documents) to derive document representations. Either way, each document is represented as one fixed feature vector. In text cube construction, however, the categorization tasks along different dimensions often require different information from the same document. Continuing the news corpus example in Figure 3.1, the location dimension may favor location-indicative terms such as "Chicago" and "China" as features, while the topic dimension may favor semantics-telling ones such as "super bowl" and "economy". Existing text categorization methods derive fixed document representations and are dimension-agnostic. As a result, irrelevant terms are overemphasized in the representation, which often hurts the categorization performance.

We propose Doc2Cube, a method that allocates documents into a text cube in an unsupervised way. Regarding label names as a small set of labeled seed terms, the key idea behind Doc2Cube is to learn learn *dimension-aware* embeddings for documents and labels for categorization. That is, it learns multiple, instead of one, representations for each document, by selectively focusing on discriminative terms for different dimensions and propagating the information from the seeds to other terms.

To learn dimension-aware embeddings for document categorization, Doc2Cube first constructs a tripartite graph to encode the correlations among labels, terms, and documents. As shown in Figure 3.2, the initial graph links each document with all the terms appearing in it and meanwhile links each label to its surface name. It then iteratively refines the graph structure and derives quality embeddings of labels, terms, and documents to uncover their inter-type similarities. During the iterative embedding process, Doc2Cube features two novel components to obtain discriminative joint embeddings: *document focalization* and *label expansion.*

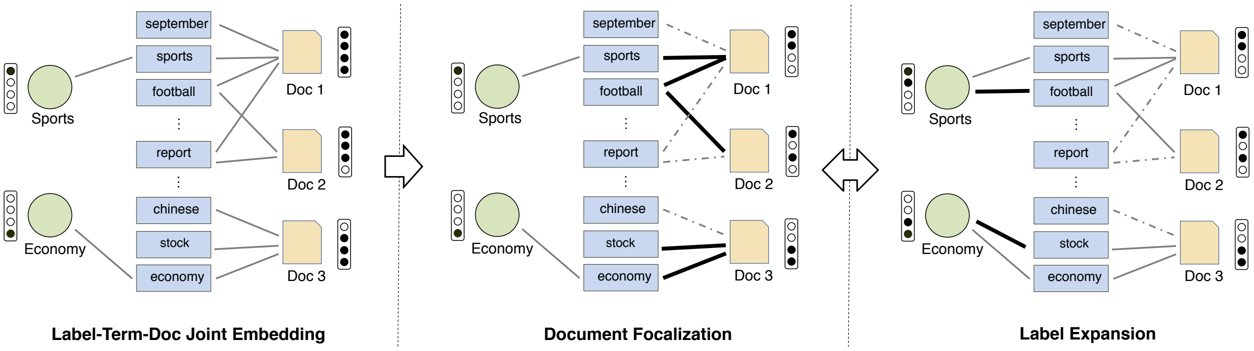


Figure 3.2: A toy example of dimension-aware joint embedding framework on the *topic* dimension. In document focalization, the background term ("report") along with the indiscriminative words ("september" and "chinese") are less emphasized for the *topic* dimension. In label expansion, more topic-indicative words ("football" and "stock") are expanded and labeled.

The document focalization component gradually sparsifies the term-document sub-graph by emphasizing discriminative terms. As shown in Figure 3.2, a document is initially connected with all the terms appearing in it. The resultant document embedding is *over-represented* in the sense that many terms indiscriminative to the current dimension are encoded. To address this issue, Doc2Cube iteratively estimates the discriminativeness of terms for each cube dimension, and emphasizes discriminative ones to generate tailored document embeddings. As such, one document can have multiple representations—each tailored for one cube dimension by highlighting truly discriminative information.

The label expansion component iteratively densifies the label-term subgraph to address the label sparsity problem. As shown in Figure 3.2, as each label is only connected to its surface name in the beginning, the initial label embedding is *under-represented* because many other relevant terms are overlooked. To tackle this issue, Doc2Cube computes the correlations between labels and terms along different dimensions, and iteratively links each label with positively correlated terms. In this way, the information is propagated from label names to other semantically relevant terms for alleviating label sparsity.

Our contributions in this chapter can be summarized as follows:

1. We propose an unsupervised method for document allocation. It does not require excessive labeled data, but simply leverages the surface names of different labels to achieve effective text categorization along different cube dimensions.

2. We propose a novel dimension-aware joint embedding algorithm. It learns dimension-aware embeddings by focusing on discriminative terms and propagating information from label names to other terms to alleviate label sparsity.

3. We have performed extensive experiments using two real-life datasets. The results show that our method generates high-quality embeddings and significantly outperforms state-of-the-art methods.

3.2 RELATED WORK

In this section, we examine related work in two areas: text cube analysis and text categorization.

3.2.1 Data Warehousing and Text Cube

Data warehouse and online analytical processing techniques [34, 15] have been demonstrated as a powerful tool for multidimensional data analytics. However, most existing literature on data warehousing are constrained to structured data like relational tables.

There have been several studies on extending data warehousing and online analytical processing techniques to unstructured data. Lin *et al.* [54] proposed to compute aggregation measures in a text cube. They assumed the text documents have been organized in a neat multi-dimensional structure and studied how to efficiently compute different aggregation measures in the multi-dimensional space. Since then, text cube analysis has drawn attention from the database and data mining communities [69, 106, 107, 23, 86]. Specifically, R-Cube [69] was proposed where users can specify an analysis portion by supplying some keywords and a set of cells are extracted based on relevance. TopCell and TEXplorer were proposed [23, 107] to support keyword-based ranking of text cube cells and facilitate interactive exploration of a text cube. A number of multi-dimensional analytical platforms [62, 84] were also developed to support end-to-end textual analytics. However, all these studies focus on the text analytics tasks, assuming the cube is already constructed by data providers. The text cube construction task, which aims at organizing massive text documents into a cube for multidimensional analysis, has remained largely overlooked.

3.2.2 Text Classification

Text cube construction is closely related to text categorization. Below, we review existing literature on both supervised and unsupervised text classification.

Supervised Text Classification

Prevailing text categorization methods take a supervised approach. Relying on a sufficient amount of document-label training pairs, they learn reliable classifiers that are capable of predicting the label of any new document, including SVM [41], decision tree [4, 77], and neural networks [96]. Different from supervised text classification, the text cube construction problem does not involve excessive labeled data, but only a text corpus and a pre-defined cube schema. Such a setting makes our problem challenging and existing supervised methods inapplicable.

Unsupervised Text Classification

There are unsupervised or weakly-supervised approaches for text categorization. Ko *et al.* [44] used heuristic rules to generate training data, but the curated labels often need considerable feature engineering efforts to ensure the quality. OHLDA [33, 17] applies topic model with given labels to generate document classifiers, while leveraging external knowledge from *Wikipedia* to represent labels. The recently developed dataless classification methods [82] also use Wikipedia to perform explicit semantic analysis of labels and documents to derive vector representations. The common limitation of OHLDA and dataless models is their dependency on external knowledge bases. They suffer from limited performance if the given corpus is closed-domain or has limited coverage by external knowledge bases.

3.3 PRELIMINARIES

3.3.1 The Document Allocation Problem

Text cube [54] is a data model that enables multi-dimensional and multi-granular text analysis. Given a text corpus $\mathcal{D}$, the text cube for $\mathcal{D}$ is a multi-dimensional data structure. The multiple dimensions, which reveal important aspects (*e.g.*, topic, location, time) of the corpus $\mathcal{D}$, uniquely define the schema of the text cube. Each document $d \in \mathcal{D}$ lies in one multi-dimensional cube cell to characterize the textual content of the document from multiple aspects. Formally, we define the concepts of *cube dimension* as follows:

Definition 3.1 (Cube Dimension)**.** *A cube dimension is defined as* $\mathcal{L} = \{l_1, l_2, \ldots, l_{|\mathcal{L}|}\}$*, where* $l_i \in \mathcal{L}$ *is a categorical label in this dimension.*

Consider Figure 3.1 as an example. There are three cube dimensions for the given corpus: (1) $\mathcal{L}_{topic}$ representing the topic aspect; (2) $\mathcal{L}_{loc}$ representing the location aspect; (3) $\mathcal{L}_{time}$

representing the time aspect. Then for each article, it should be associated with one label for each of the three dimensions, *e.g.*, label "Economy" for $\mathcal{L}_{topic}$, label "China" for $\mathcal{L}_{loc}$, and label "2017" for $\mathcal{L}_{time}$. The labels from different dimensions partition the space into cube cells.

Definition 3.2 (Cube Cell). *Given n cube dimensions, $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n$, a cube cell c is defined as a n-dimensional tuple $(l_1, l_2, \ldots, l_n)$, where l_i $(1 \leq i \leq n)$ is a label in dimension L_i.*

Definition 3.3 (Text Cube). *A text cube for a text corpus $\mathcal{D}$ is a n-dimensional structure $\mathcal{C} = (\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n)$, where $\mathcal{L}_i$ is the i-th cube dimension. Each document $d \in \mathcal{D}$ resides in a cube cell $(l_{t_1}, \ldots, l_{t_n})$ in $\mathcal{C}$, where l_{t_i} is the label of d in dimension $\mathcal{L}_i$.*

We study the problem of allocating a text corpus $\mathcal{D}$ into a text cube $\mathcal{C}$. In tradition data cube literature [15], this process is also called *cube instantiation* or *cube loading*. We formally define this problem in the following.

Problem 3.1 (Document Allocation). *Let $\mathcal{C}$ be a n-dimensional text cube with dimensions $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_n$, and $\mathcal{D}$ be a corpus of text documents. For any document $d \in \mathcal{D}$, the text cube construction problem is to allocate d into a n-dimensional cell in $\mathcal{C}$. This is equivalent to assigning n labels $l_{t_1}, \ldots, l_{t_n}$ for d, where label $l_{t_i} \in \mathcal{L}_i$ represents the category of d in dimension $\mathcal{L}_i$.*

3.3.2 Doc2Cube: Document Allocation via Dimension-Aware Joint Embedding

The major challenge for applying document classification methods is that there are no labeled documents for training reliable classifiers. Instead, one needs to perform document categorization along different dimensions using only label names and document content. Our method Doc2Cube uses label names to form a small set of seed labeled terms, and use them as weak supervision signals for document categorization. At the high level, Doc2Cube is an approach that learns distributed representations of labels, terms, and documents. Instead of using bag-of-words as document representation, it learns low-dimensional document embeddings by discovering the correlations among terms.

As shown in Figure 3.2, Doc2Cube initially constructs a tripartite *label-term-document* graph to encode the relationships among labels, terms, and documents along different dimensions, and embeds them into the same latent space. While the initial embeddings encode the seed information and the occurrences of terms in documents, they suffer from two drawbacks: (1) the document embeddings are *over-represented* in the sense that many terms indiscriminative to the current dimension are encoded; and (2) the label embeddings are *under-represented* because many other relevant terms are overlooked. To address the above challenges, Doc2Cube features two novel components for learning discriminative joint embeddings in an iterative fashion: (1) the *document focalization* component that emphasizes different terms for different dimensions, thus deriving dimension-aware document representations; and (2) the *label expansion* component that propagates information from label names to other terms for alleviating label scarcity.

3.4 GRAPH-BASED JOINT EMBEDDING

In this section, we describe the joint label-term-document embedding step. For a given dimension $\mathcal{L}$, it first constructs a *Label-Term-Document* tripartite graph (Section 3.4.1) and then embeds different data types into the same latent space (Section 3.4.2).

3.4.1 Label-Term-Document Graph

To model the relationships among labels, terms and documents, we construct an *Label-Term-Document (L-T-D)* correlation graph. Since different dimensions have different label spaces, we construct an L-T-D graph for each dimension separately. As shown in Figure 3.2, for each cube dimension, there are three different node types: labels, terms, and documents. The initial graph G_{LTD} is designed to capture two types of relationships: (1) the seed information between label names and terms; and (2) the occurrence information between terms and documents. Hence, we induce two different edge types to encode these relationships: label-term edges and document-term edges. The resultant L-T-D graph is a heterogeneous tripartite graph defined as follows.

Definition 3.4 (L-T-D Graph). *The L-T-D graph for a dimension $\mathcal{L}$ is a tripartite graph $G_{LTD} = (V_{LTD}, E_{LTD})$. The node set V_{LTD} contains all the labels in $\mathcal{L}$, terms in $\mathcal{T}$, and documents in $\mathcal{D}$. The edge set E_{LTD} consists of two types of edges: (1) E_{TL} is a set of edges between labels and terms. There is an edge between term t_i and label l_j if and only if they strictly match each other, and the weight $w^{TL}_{i,j}$ is set to 1; (2) E_{TD} is a set of edges between terms and documents. There is an edge between term t_i and document d_j if t_i occurs in d_j, and the edge weight $w^{TD}_{i,j}$ is set to* $\log(1 + \mathit{count}(t_i, d_j))$.

3.4.2 Learning Joint Embeddings

The L-T-D graph G_{LTD} encodes the information from seed terms as well as the co-occurrence relationships between terms and documents. Based on the constructed L-T-D graph, we proceed to learn initial vector representations of labels, terms, and documents. This is achieved by embedding all the nodes in the L-T-D graph into a D-dimensional vector space, such that their structural proximities in the graph are preserved. Here, D is a parameter that specifies the dimensionality of the embedding space, *e.g.*, $D = 200$.

The L-T-D graph G_{LTD} is a tripartite graph between labels, terms, and documents. We design the graph embedding task to preserve the information from both the label-term edges E_{TL} and the term-document edges E_{TD}. For this purpose, we define the probability of observing a term i given a label j as follows:

$$p(\mathbf{u}_i^T|\mathbf{u}_j^L) = \frac{\exp(\mathbf{u}_i^T \cdot \mathbf{u}_j^L)}{\sum_{i' \in \mathcal{T}} \exp(\mathbf{u}_{i'}^T \cdot \mathbf{u}_j^L)}, \tag{3.1}$$

where $\mathbf{u}_i^T$ and $\mathbf{u}_j^L$ are the D-dimensional embeddings of term i and label j, respectively. Similarly, we define the probability of observing a term i given a document j as follows:

$$p(\mathbf{u}_i^T|\mathbf{u}_j^D) = \frac{\exp(\mathbf{u}_i^T \cdot \mathbf{u}_j^D)}{\sum_{i' \in \mathcal{T}} \exp(\mathbf{u}_{i'}^T \cdot \mathbf{u}_j^D)}. \tag{3.2}$$

Now given the L-T-D graph G_{LTD}, we learn the embeddings of labels, terms, and documents by collectively preserving the structures of the two bipartite graphs E_{TL} and E_{TD}. This is achieved by minimizing the following objective function:

$$\mathcal{O}_{ltd} = \mathcal{O}_{lt} + \mathcal{O}_{td}, \tag{3.3}$$

where

$$\begin{aligned} \mathcal{O}_{lt} &= - \sum_{(i,j) \in E_{TL}} w_{i,j}^{TL} \log p(\mathbf{u}_i^T|\mathbf{u}_j^L), \\ \mathcal{O}_{td} &= - \sum_{(i,j) \in E_{TD}} w_{i,j}^{TD} \log p(\mathbf{u}_i^T|\mathbf{u}_j^D). \end{aligned}$$

The above objective function is expensive to optimize due the large amount of terms in the vocabulary. To efficiently learn the joint embeddings, we use the negative sampling strategy [63] with stochastic gradient descent (SGD) for optimizing Equation 3.3.

3.5 DIMENSION-AWARE EMBEDDING LEARNING

In this section, we present the dimension-aware embedding updating step. Taking the joint embeddings as initialization, the updating step iteratively derives dimension-aware document embeddings by focusing on discriminative terms for each dimension, and expands the initial labeled seed terms to address label sparsity.

3.5.1 Measuring Term Discriminativeness

Although the joint embeddings capture the co-occurrence information among labels, terms, and documents, the resultant embeddings suffer from two problems. First, the document embedding is fixed for all the dimensions. In text cube construction, different dimensions require different representation for the same document. For instance, the location dimension may favor terms that captures location-related information, such as “new york”, while topic dimension may favor terms that captures topical information, such as “super bowl” and “economic growth”. Second, the scarcity of labeled terms makes label embeddings not comprehensive enough to cover the semantics of the target category. For example, with the provided seeds, the label “Sports” is only linked to the term “sports”. However, the scope of “Sports” is quite broad, covering information such as “nba”, “nfl”, and “soccer”. Consequently, the initial joint embeddings over-represent documents while under-represent labels.

The key to tackling the above two problems is to estimate each term's discriminative power *w.r.t.* a dimension and a label. The computed discriminative scores can address the over-represented document embedding problem by emphasizing discriminative terms and understating indiscriminative ones. In the mean time, for the under-represented label embedding problem, the discriminative scores of terms allow for expanding each label to highly relevant terms. In what follows, we define the *label-focal score* and the *dimension-focal score* of a term t and describe how we compute these two measures.

Label-Focal Score

The label-focal score of a term t *w.r.t.* a label l in dimension $\mathcal{L}$, denoted as $f(t, l)$, aims at quantifying the discriminative power of the term t for the label l. The higher $f(t, l)$ is, the more exclusively the term t belongs to the label l.

Our strategy for measuring the label-focal score $f(t, l)$ is to leverage the documents containing t to derive the distribution of term t over all the labels in dimension $\mathcal{L}$. Specifically, with the document embedding matrix $\mathbb{U}^{\mathcal{D}}$ and the label embedding matrix $\mathbb{U}^{\mathcal{L}}$, we first compute the label-document similarity matrix as:

$$\mathbf{R}^{(\mathcal{DL})} = \mathbb{U}^{\mathcal{D}} {\mathbb{U}^{\mathcal{L}}}^{\mathrm{T}}. \tag{3.4}$$

In the above, $\mathbf{R}^{(\mathcal{DL})}$ is a $|\mathcal{D}| \times |\mathcal{L}|$ matrix that gives the similarities between documents and labels in the embedding space. Combining it with the term-document subgraph, we are able to further compute the similarities between labels and terms. Specifically, let $\mathbf{A}^{(\mathcal{TD})}$ be the adjacency matrix for the term-document subgraph in G_{LTD}, we compute the term-label similarities as:

$$\mathbf{R}^{(\mathcal{TL})} = \mathbf{A}^{(\mathcal{TD})} \mathbf{R}^{(\mathcal{DL})}, \tag{3.5}$$

where $\mathbf{R}^{(\mathcal{TL})}$ is a $|\mathcal{T}| \times |\mathcal{L}|$ matrix keeping the similarities between terms and labels. Base on $\mathbf{R}^{(\mathcal{TL})}$, we apply row-wise softmax function to derive the probability distribution of each term over the labels. Finally, we define the *label-focal score* $f(t_i, l_j)$ as the probability of assigning term t_i to label l_j. Namely,

$$f(t_i, l_j) = \mathbf{R}_{ij}^{(\mathcal{TL})}. \tag{3.6}$$

Dimension-Focal Score

We proceed to define the dimension-focal score of a term. The dimension-focal score of a term t_i *w.r.t.* dimension $\mathcal{L}$, denoted as $f(t_i, \mathcal{L})$, aims to quantify how discriminative the term t_i is for the categorization task along dimension $\mathcal{L}$. The higher $f(t_i, \mathcal{L})$ is, the more useful term t_i is for deciding the label in dimension $\mathcal{L}$.

We measure the dimension-focal score $f(t_i, \mathcal{L})$ based on the distribution of term t_i over all the labels in dimension $\mathcal{L}$. Recall that the matrix $\mathbf{R}^{(\mathcal{TL})}$ gives the label distribution of term t_i. We compute its normalized KL-divergence from the uniform distribution of t_i over all the labels as the dimension-focal score. Formally, the *dimension-focal score* $f(t_i, \mathcal{L})$ is given by:

$$f(t_i, \mathcal{L}) = \frac{\sum_{j=0,\cdots,|\mathcal{L}|} \mathbf{R}_{ij}^{(\mathcal{TL})} \log |\mathcal{L}| \mathbf{R}_{ij}^{(\mathcal{TL})}}{\log |\mathcal{L}|}, \tag{3.7}$$

where $\log |\mathcal{L}|$ is a normalization term.

3.5.2 Document Focalization

The *document focalization* component uses the dimension-focal scores of terms to address the *over-represented* problem of document embeddings. The rationale is that the fixed document representation encodes the information from all the terms in the vocabulary, even those that are not relevant to the categorization task in the target dimension. With dimension-focal scores, it becomes possible to emphasize discriminative terms while understating irrelevant ones. Consider Figure 3.2 as an example. As shown, for the topic dimension, the first document is connected to topical terms such as "football" and "sports", as well as time-indicative terms like "september" and background terms like "report". Those irrelevant terms in the document can act as background noise and make the categorization task more difficult. Document focalization remedies this problem by emphasizing discriminative terms and generating dimension-tailored document representations, *e.g.*, lowering the weights of "september" and "report" for that document.

To obtain dimension-tailored document embeddings, we use the dimension-focal scores to re-weigh the term-document matrix $\mathbf{A}^{(\mathcal{TD})}$, and compute the weighted average of term embeddings. Formally, we update the document embedding matrix $\mathbb{U}^{\mathcal{D}}$ as:

$$\mathbb{U}^{\mathcal{D}} = \left(\mathbf{A}^{(\mathcal{TD})} \circ \Big[f_{\mathcal{L}} \cdots f_{\mathcal{L}} \Big]_{|\mathcal{T}|\times|\mathcal{D}|}\right)^{\mathrm{T}} \mathbb{U}^{\mathcal{T}} \tag{3.8}$$

where $\circ$ is the Hadamard product between two matrices; and $f_{\mathcal{L}}$ is a length-$|\mathcal{T}|$ vector representing the dimension-focal scores of all the terms along dimension $\mathcal{L}$. In this formula, the dimension-focal score of each term places a penalty in the range of $[0,1]$ on the original weight in the matrix $\mathbf{A}^{(\mathcal{TD})}$. The document embedding is then an aggregation of term embeddings with penalized weights. The higher a term's dimension-focal score is, the more it is emphasized when computing the document embedding.

Observing from Equation 3.4 and 3.8, it is apparent that the computations of the focal scores $f_{\mathcal{L}}$ and the document embeddings $\mathbb{U}^{\mathcal{D}}$ are dependent on each other. These two measures can mutually enhance each other: (1) better document representations lead to more accurate labeling of the documents and thus better estimations of term focal scores; and (2) more accurate focal scores surface terms that are important to the dimension and result in more discriminative document embeddings. Consequently, we design an iterative process that updates $\mathbf{R}^{(\mathcal{DL})}$, $f_{\mathcal{L}}$ and $\mathbb{U}^{\mathcal{D}}$ alternatively until they stablize. We will describe the iterative process shortly.

3.5.3 Label Expansion

The *label expansion* component is designed to solve the *under-represented* problem of label embeddings. The intuition behind it is to link each label with other positively correlated terms in addition to its surface name. For example, in Figure 3.2, it is reasonable to expand the label "Sports" to the term "football", and the label "Economy" to the term "stock". As such, the label-term subgraph is enriched and the obtained label representations encode the semantics of relevant terms more comprehensively.

To ensure the quality of the expanded terms, we consider two factors: (1) the label-focal score of a term; and (2) the popularity of a term. The label-focal score is critical to determining the correlations between a term and the considered label. However, we observe that only using the label-focal score could link the label to many low-quality terms during the label expansion process. This is because many terms that have high discriminative power are infrequent in the corpus. Expanding labels to them not only covers few extra documents, but also suffers from their inadequately-trained embeddings. Hence, we design the expansion criterion by combining the label-focal score and the term popularity. Given a term t_i and a label l_j, we compute the expansion score of term t_i for label l_j as:

$$e(t_i, l_j) = f(t_i, l_j) \cdot \frac{\log 1 + df(t_i)}{\log 1 + |\mathcal{D}|} > \eta \tag{3.9}$$

where $df(t_i)$ is the document frequency of term t_i. The second term thus reflects the normalized popularity of term t_i. In Equation 3.9, $\eta > 0$ is a pre-defined threshold for label expansion. Any term-label pairs with the expansion scores higher than η are connected and the adjacency matrix $\mathbf{A}^{(\mathcal{LT})}$ is updated accordingly. After the expansion, we compute the label embedding as:

$$\mathbb{U}^{\mathcal{L}} = \mathbf{A}^{(\mathcal{LT})}\mathbb{U}^{\mathcal{T}}. \tag{3.10}$$

Since the label expansion process changes label embeddings, the label-focal scores of terms will be updated according to the newly computed $\mathbf{R}^{(\mathcal{DL})}$ and $\mathbf{R}^{(\mathcal{TL})}$. As label-focal scores are updated, a new label expansion operation could further benefit generating high-quality label embeddings. We design an iterative process to perform label expansion and focal score computation in turn, which will be described shortly.

3.6 THE OVERALL DOC2CUBE ALGORITHM

In this section, we put different pieces together and summarize the entire procedure of Doc2Cube for text cube construction. There are three major steps in Doc2Cube: (1) joint embedding of labels, terms, and documents; (2) dimension-aware embedding updating; and (3) label assignment. Algorithm 3.1 sketches the overall process of Doc2Cube. As shown, given the corpus, we first build the L-T-D tripartite graph and compute the joint embeddings of labels, terms, and documents (lines 2 - 8). Then we iteratively update the embeddings based on Algorithm 3.2 to derive dimension-aware document and label embeddings (line 9). Finally, we assign the max-scoring label to each document for the target dimension (line 10 - 11). The label assignment step is achieved by directly measuring the cosine similarity between label embedding and document embedding.

Algorithm 3.1: The overall procedure of Doc2Cube.

```
Input:  $\mathcal{D}$: a corpus of text documents
        $\mathcal{T}$: the vocabulary of terms in $\mathcal{D}$
        $\mathcal{L}_1, \cdots, \mathcal{L}_n$: the label sets for the $n$ dimensions
        $K$: the number of negative samples in SGD
        $M$: the maximum number of samples in embedding
Output: The labels for the documents in $\mathcal{D}$.
1   for $\mathcal{L}$ in $\mathcal{L}_1, \cdots, \mathcal{L}_n$ do
        // L-T-D graph construction
2       Construct $G_{LTD}$ using $\mathcal{D}, \mathcal{T}, \mathcal{L}$;
        // Embedding learning
3       Randomly initialize $\mathbb{U}^{\mathcal{L}}$, $\mathbb{U}^{\mathcal{T}}$ and $\mathbb{U}^{\mathcal{D}}$ ;
4       for $i = 1 : M$ do
5           Sample an edge $e \in E_{TL}$ and $K$ negative edges;
6           Update $\mathbb{U}^{\mathcal{T}}$ and $\mathbb{U}^{\mathcal{L}}$;
7           Sample an edge $e \in E_{TD}$ and $K$ negative edges;
8           Update $\mathbb{U}^{\mathcal{T}}$ and $\mathbb{U}^{\mathcal{D}}$;
        // Embedding updating
9       $\mathbb{U}^{\mathcal{D}}$, $\mathbb{U}^{\mathcal{L}}$ = Embed_Update($G_{LTD}$, $\mathbb{U}^{\mathcal{L}}$, $\mathbb{U}^{\mathcal{D}}$, $\mathbb{U}^{\mathcal{T}}$);
        // Construction using embeddings
10      for $d_i$ in $\mathcal{D}$ do
11          label($d_i$) = $\text{argmax}_{l_j \in \mathcal{L}} \cos(\mathbf{u}_i^D, \mathbf{u}_j^L)$;
```

Algorithm 3.2 presents the iterative embedding updating process for document and label embeddings. Starting with the initial embeddings for labels ($\mathbb{U}^{\mathcal{L}}$), terms ($\mathbb{U}^{\mathcal{T}}$), and documents ($\mathbb{U}^{\mathcal{D}}$), we iteratively perform document focalization and label expansion to obtain more discriminative dimension-aware embeddings. In the document focalization component (lines 2 - 5), we compute the dimension-focal scores of terms, and update the document embeddings according to Equation 3.8; while in the label expansion component (lines 6 - 8), we compute the label-focal scores of terms, and update the label embeddings according to Equation 3.10.

Algorithm 3.2: Dimension-Aware Embedding Updating.

```
Input: $\mathbb{U}^{\mathcal{L}}$, $\mathbb{U}^{\mathcal{D}}$, $\mathbb{U}^{\mathcal{T}}$: initial embeddings of labels, docs and terms.
       $\mathbf{A}^{(\mathcal{L}\mathcal{T})}$: the adjacency matrix for the label-term subgraph
       $\mathbf{A}^{(\mathcal{T}\mathcal{D})}$: the adjacency matrix for the term-document subgraph.
       $T$: the number of iterations for updating
Output: The updated embeddings of labels and documents.
for $iter = 1 : T$ do
    // Document focalization
    Compute $\mathbf{R}^{(\mathcal{T}\mathcal{L})}$ by Equation 3.4 and 3.5;
    for $t_i$ in $\mathcal{T}$ do
        $f(t_i, \mathcal{L}) = \frac{\sum_{j=0,\cdots,|\mathcal{L}|} \mathbf{R}_{ij}^{(\mathcal{T}\mathcal{L})} \log |\mathcal{L}| \mathbf{R}_{ij}^{(\mathcal{T}\mathcal{L})}}{\log |\mathcal{L}|}$
    // Update document embeddings
    $\mathbb{U}^{\mathcal{D}} = \left( \mathbf{A}^{(\mathcal{T}\mathcal{D})} \circ \left[ f_{\mathcal{L}} \cdots f_{\mathcal{L}} \right]_{|\mathcal{T}| \times |\mathcal{D}|} \right)^{T} \mathbb{U}^{\mathcal{T}}$;
    // Label expansion
    Compute $e(t, l)$ for all term-label pairs by Equation 3.9;
    Update $\mathbf{A}^{(\mathcal{L}\mathcal{T})}$ for all $e(t, l) > \eta$;
    // Update label embeddings
    $\mathbb{U}^{\mathcal{L}} = \mathbf{A}^{(\mathcal{L}\mathcal{T})} \mathbb{U}^{\mathcal{T}}$
```

Time Complexity. The total time cost of DOC2CUBE involves two parts: (1) the initial joint embeddings; and (2) the iterative updating. For the first part, along each dimension, DOC2CUBE needs to sample M edges for graph embedding. For each sampled edge, DOC2CUBE generates K negative samples to update the D-dimensional embeddings. The time cost is thus $O(nMKD)$. For the second part, DOC2CUBE performs T iterations for updating the embeddings. In each iteration, computing the focal scores takes $O(n \cdot |\mathcal{T}| \cdot |\mathcal{D}| \cdot |\mathcal{L}|_{max} \cdot D)$ time where $|\mathcal{L}|_{max}$ is the maximum cardinality of the label set for all the dimensions. Once the focal scores are computed, DOC2CUBE updates the embeddings with time complexity $O(n \cdot |\mathcal{T}| \cdot |\mathcal{D}| \cdot |\mathcal{L}|_{max} \cdot D)$. The overall time complexity of DOC2CUBE is $O(nMKD + nT \cdot |\mathcal{T}| \cdot |\mathcal{D}| \cdot |\mathcal{L}|_{max} \cdot D)$. Note that the variables n, K, D, T, and $|\mathcal{L}|_{max}$ are usually small in practice.

3.7 EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of DOC2CUBE. We design the experiments to answer the following questions: (1) Can DOC2CUBE achieve better performance compared with existing methods? (2) How effective are the document focalization and label expansion components? and (3) Is DOC2CUBE fast enough to process large-scale text corpora?

3.7.1 Experimental Setup

Dataset

We use the following two real datasets in our experiments:

- The first dataset, named NYT, is a collection of New York Times articles. We crawled 13,080 articles using New York Time API[1] in 2015. The articles in the corpus cover 29 topics and 5 countries, and each article contains exactly one topic label and one country label. Accordingly, two dimensions are involved for constructing a text cube for the NYT corpus: *Topic* and *Location*. The annotations of different articles along these two dimension are used as ground truth. Before applying different methods on this dataset, we use an existing phrase mining tool[2] to segment each article into phrases. Furthermore, we remove all the stopwords and the phrases that appear less than 10 times.

- Our second dataset, Yelp, is a collection of business reviews from the Yelp Data Challenge[3]. We aim to predict two attributes for each review: (1) business category; and (2) created location. Due to the long-tail nature of the raw dataset, we preprocess it by selecting the five most popular categories and states to form the label spaces, and choose the reviews falling in those top five categories and states. After preprocessing, we obtain 128,868 Yelp reviews in total.

Baselines

To demonstrate the effectiveness of Doc2Cube, we compare it with multiple baselines that can perform document categorization in an unsupervised or semi-supervised way:

- **IR** [73] treats each label as a keyword query and performs categorization based on the BM25 retrieval model. Using BM25, the label that achieves the highest query relevance is assigned to the considered document.

- **IR + Expansion (IR+QE)** [74, 22] extends the IR method by expanding label names using *Word2Vec* [63] and using the expanded term set as queries.

[1] http://developer.nytimes.com/

[2] https://github.com/shangjingbo1226/SegPhrase

[3] https://www.yelp.com/dataset/challenge

- **Word2vec (W2V)** [63] first learns vector representations for all the terms in a given corpus, and then derives label and document representations by aggregating their member terms. Finally, the most similar label for a document is assigned based on cosine similarity.

- **Word2vec + Focalization (W2V+DF)** extends W2V with our document focalization component. Instead of simply aggregating term embeddings for document representation, we leverage term dimension-focal scores to compute document representations.

- **Paragraph2vec (P2V)** [48] directly learns vector representations of documents, by embedding documents and terms into the same semantic space.

- **Semi-Supervised Topic Model (Semi-TM)** [56] extends the PLSA model [**?**] by using labels as guidance and forcing the learned topics to align with the provided labels.

- **Dataless Classification (Dataless)** [82, 14, 17] is an unsupervised algorithm that utilizes Wikipedia as external knowledge base. It leverages Wikipedia and Explicit Semantic Analysis (ESA) to derive vector representations of labels and documents.

- **PTE** [83] is a semi-supervised method that jointly embeds documents, terms, and labels into the same latent space and directly uses the embeddings for categorization.

Besides the above baseline methods, we also design two ablation algorithms to evaluate the separate effects of document focalization and label expansion for Doc2Cube:

- **D2C-DF** updates document embeddings for each dimension using document focalization. However, the label embeddings are not updated with the label expansion component.

- **D2C-LE** updates label embeddings iteratively with the label expansion component. However, it does not include document focalization for deriving dimension-aware document embeddings.

Evaluation Protocol

For our used datasets, there are two dimensions for each corpus, and every document has one label along each dimension. To evaluate the performance of different methods, we use them to allocate all the documents in the corpus, and measure the F1 scores along different dimensions.

We set the parameters of different methods as follows. There are three major parameters in Doc2Cube: (1) the latent embedding dimension D; (2); the number of iterations for embedding updating T; and (3) the correlation threshold for label expansion η. After tuning, we set these parameters as the following on both datasetes: $D = 100, T = 3$ and $\eta = 0.8$. We will also show the performance of Doc2Cube when these parameters vary. For the baseline methods, we set the embedding dimensions for W2V and PTE to 100 to ensure fair comparison with Doc2Cube; we set the number of topics to 20 for Semi-TM; and we set the number of Wikipedia concepts to 500 for Dataless.

3.7.2 Effectiveness Evaluation

In this subsection, we demonstrate the effectiveness of different methods, and also study the effects of different parameters on their performance.

Performance Comparison

In the first set of experiments, we compare the effectiveness of different methods. As shown in Table 3.1, we report the micro-F1 and macro-F1 scores of all the methods along different dimensions. One can observe that Doc2Cube outperforms all the baselines in both dimensions on NYT and Yelp. Specifically, Semi-TM is the strongest baseline along the topic and location dimensions on NYT (Semi-TM does not perform as well on Yelp, possibly because of the shortness nature of Yelp reviews). However, Doc2Cube outperforms Semi-TM by more than 16.2% in the topic dimension and 37.3% in the location dimension. On the Yelp dataset, Doc2Cube again outperforms the strongest baseline (W2V+DF and Semi-TM) by 22.4% and 4.5% along the business category and the location dimensions, respectively.

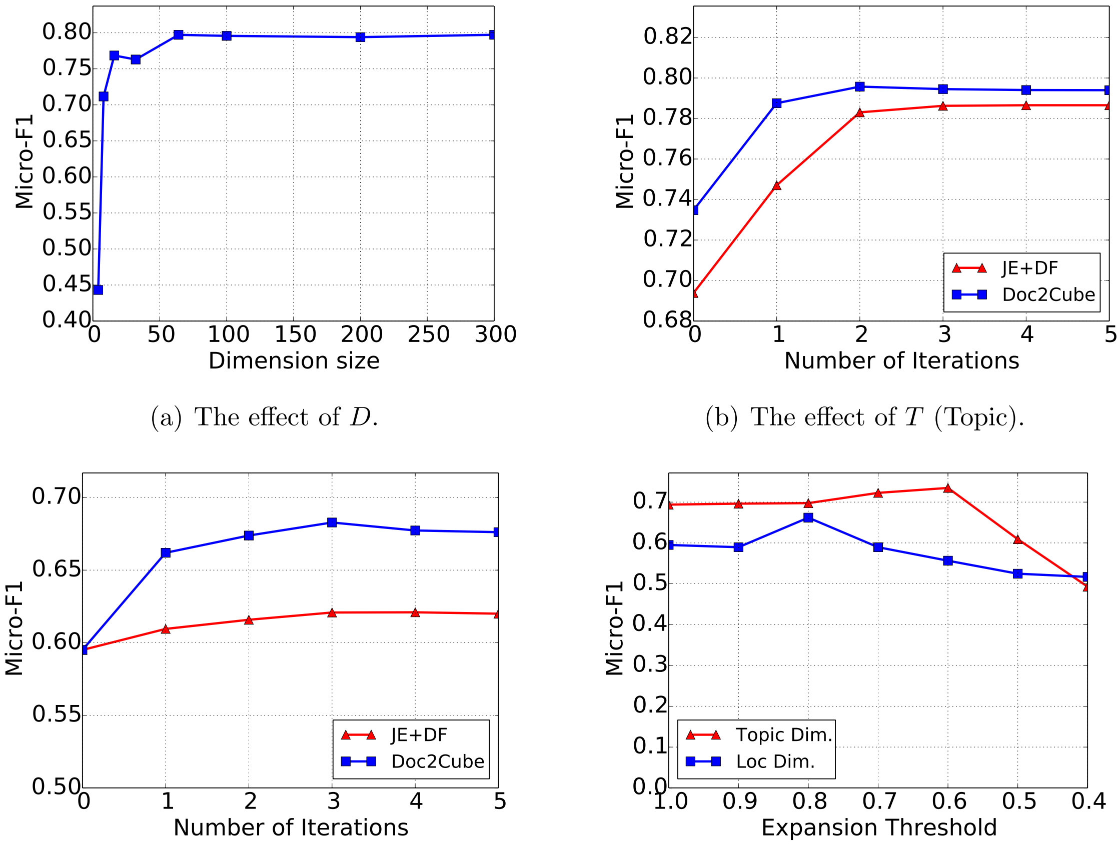
Comparing the absolute performance of different methods on NYT and Yelp, we observe that all the methods perform better on NYT. This phenomenon is reasonable as the average document length on NYT is much larger than Yelp, allowing different methods to capture more discriminative signals for the categorization tasks. Meanwhile, the performance of all the methods is significantly lower in the location dimension on Yelp. Our investigations into the data reveal that many Yelp reviews do not mention any location-indicative information, which is the major reason explaining the relatively lower performance in the location dimension compared to the business category dimension.

Table 3.1: The performance of different methods on the NYT and Yelp datasets. The NYT dataset involves two dimensions: Topic and Location; the Yelp dataset also involves two dimensions: Business Category and Location. For each dimension, we measure the micro-F1 and macro-F1 scores of different methods for categorization.

	NYT				Yelp			
	Topic		Location		Business Category		Location	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
IR	0.3963	0.4520	0.4615	0.517	0.2957	0.3669	0.0547	0.3111
IR+QE	0.4112	0.4744	0.4722	0.4726	0.3276	0.3726	0.0779	0.2806
W2V	0.5928	0.3891	0.5226	0.3598	0.4980	0.4635	0.1915	0.2530
W2V+DF	0.6100	0.3981	0.5446	0.4156	0.5129	0.5257	0.2392	0.2532
P2V	0.6079	0.4018	0.3337	0.3511	0.1920	0.3752	0.1766	0.2421
Dataless	0.5882	0.3724	0.5	0.4362	0.1463	0.1733	0.1080	0.1981
Semi-TM	0.6845	0.5407	0.5704	0.4588	0.2105	0.1876	0.3645	0.1990
PTE	0.6938	0.4992	0.595	0.4695	0.4459	0.4387	0.2505	0.2465
D2C-DF	0.7863	0.5235	0.6208	0.5635	0.6059	0.5707	0.3508	0.3010
D2C-LE	0.7347	0.5081	0.6619	0.5415	0.5261	0.5164	0.2939	0.2894
Doc2Cube	**0.7957**	**0.5414**	**0.6828**	**0.5986**	**0.6279**	**0.6037**	**0.3811**	**0.3165**

From Table 3.1, one can clearly observe the necessity of learning dimension-aware embeddings to achieve good performance across all the dimensions. We can see while certain dimension-agnostic methods (*e.g.*, W2V, P2V) can achieve reasonably good performance in the topic dimension, their performance drops drastically in the location dimension. In contrast, Doc2Cube achieves strong performance on both the topic and location dimensions, which validates the benefits of our design of learning dimension-aware document representations.

Comparing the different ablations of Doc2Cube, we can observe the benefits of the document focalization and label expansion components. For example, on the NYT dataset, the inclusion of document focalization (D2C-DF v.s. PTE) improves the micro-F1 score from ∼0.69 to ∼0.78 in the topic dimension; and the inclusion of label expansion (D2C-LE v.s. PTE) improves the micro-F1 score from ∼0.69 to ∼0.73. The reason is that, document focalization identifies a set of terms that are highly discriminative to the target dimension and leads to higher-quality document representations, while label expansion connects each label with a comprehensive set of relevant terms and thus addresses label sparsity. Interestingly, by applying document focalization (W2V+DF) and label expansion (IR+QE) on baseline methods, we also observed considerable performance gains along different dimensions. Such a phenomenon further demonstrates the effectiveness of document focalization and label expansion. The effects of document focalization and label expansion can be similarly observed on the Yelp dataset.



(a) The effect of D.

(b) The effect of T (Topic).

(c) The effect of T (Location).

(d) The effect of threshold η.

Figure 3.3: The effects of different parameters on the performance of Doc2Cube on the NYT dataset: (a) the embedding dimensionality D; (b, c) the number of iterations T; and (d) the similarity threshold η.

Effects of Parameters

In this section, we study the effects of different parameters on the performance of Doc2Cube, including: 1) the dimensionality D for joint embedding; 2) the number of iterations T in the iterative embedding updating procedure; and 3) the similarity threshold η for label expansion.

Figure 3.3(a) shows the effect of the embedding dimensionality D. As shown, when D increases from 4 to 300, the performance of our method Doc2Cube first increases rapidly, and then stabilizes after the dimensionality is larger than 64. This phenomenon is intuitive, because a larger embedding dimensionality D leads to a more expressive model that captures latent semantics better.

Figure 3.3(b) and 3.3(c) show the effects of the number of iterations T for embedding updating. Along both the topic and location dimensions, we observe that the performance improves rapidly in the first two iterations and then gradually stablizes after three iterations. The first iteration computes the *dimension-focal scores* using the doc-label similarity matrix $\mathbf{R}^{(\mathcal{DL})}$ that is derived from initial embeddings. It is capable of identifying most non-focal terms and down-voting them to generate dimension-tailored representations. However, the latter iterations are also useful as the document-label similarity matrix $\mathbf{R}^{(\mathcal{DL})}$ keeps being refined.

Figure 3.3(d) shows the effect of the label expansion threshold η, which controls the stopping criteria of label expansion. A bigger η imposes a stricter condition when connecting the label to relevant terms. As shown in Figure 3.3(d), as η varies from 1.0 to 0.4, the micro-F1 scores for both dimensions first increase and then decrease rapidly. This phenomenon is reasonable. When η is large, a slightly smaller η can include more terms to enrich the semantics of label embeddings. However, when η is too small, noisy terms that are not very correlated with the label could be included and deteriorate the performance.

3.7.3 Case Study

In this subsection, we perform a set of case studies. We are first interested in examining the computed dimension-focal scores of different terms on the NYT dataset. For this purpose, we pick five terms in the vocabulary and show their dimension-focal scores in the topic and location dimensions in Table 3.2. From the results, we can see that: (1) The first two terms, “economic growth” and “soccer”, both have very high focal scores in the topic dimension but low scores in the location dimension. This is intuitive as these two terms are quite topic-indicative but do not naturally reflect the location of a given article. In the joint embedding procedure, these terms are emphasize when generating topic-aware representations and de-emphasized when generating location-aware representations. (2) The terms “beijing” and “new york state” are only discriminative for the location dimension. These terms do not carry topical semantics but are very useful signals for deciding the locations of news events. (3) There are also terms that have high focal scores in both the topic and location dimensions, such as “chinese consumer”. It makes sense as one can easily tell the topics and locations of news articles from such terms.

We proceed to demonstrate the empirical results of the label expansion component. In Table 3.3, we choose four labels in each of the topic and location dimensions and show the label expansion results in three rounds. The results clearly show why the label expansion is useful. Starting from the surface name of a label, Doc2Cube is capable of discovering other terms that are highly correlated with the label and include them for generating label embeddings. For example, for the label “movies” in the topic dimension, Doc2Cube iter-

Table 3.2: The dimension-focal scores of different terms in the topic and location dimension on NYT.

	Topic	Location
economic growth	0.972	0.223
soccer	0.883	0.096
beijing	0.245	0.681
new york state	0.166	0.788
chinese consumer	0.999	0.994

atively discovers correlated terms such as "films", "director", and "hollywood". Similarly, in the location dimension, Doc2Cube expands the label "China" by including terms like "chinese", "beijing" and "shanghai". One can imagine that, although many documents describing "China" may not explicitly use the term "china", the label expansion component will enrich the semantic coverage of the label "China" and give high scores to those using "chinese", "beijing" and "shanghai". Such a property effectively reduces label sparsity and improves the text cube construction performance.

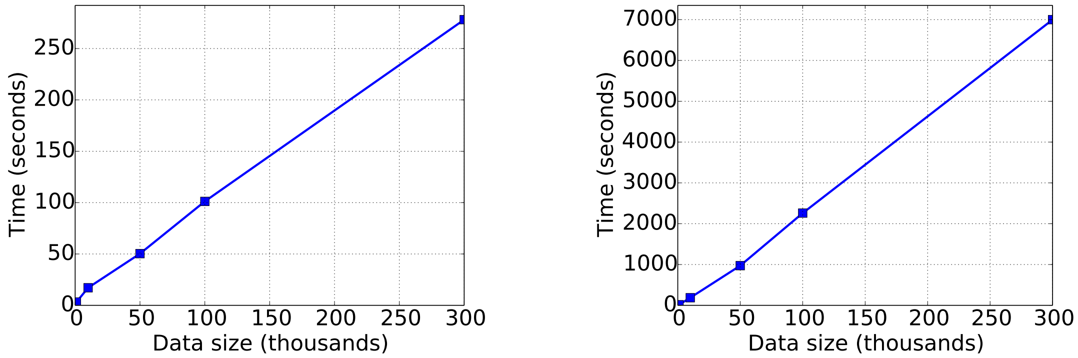
Table 3.3: The label expansion results for four example labels in the topic and location dimensions on the NYT dataset.

Round	**Topic**			
Seed	*movies*	*baseball*	*tennis*	*business*
#1	films	inning	wimbleldon	company
#2	director	hits	french open	chief executive
#3	hollywood	pitch	grand slam	industry
Round	**Location**			
Seed	*brazil*	*Australia*	*China*	*Spain*
#1	brazilian	sydney	chinese	madrid
#2	san paulo	australian	shanghai	barcelona
#3	confederations cup	melbourne	beijing	la liga

3.7.4 Scalability Study

Finally, we study the scalability of Doc2Cube. We take the NYT dataset and use random sampling to generate multiple corpora with different sizes. The running time of Doc2Cube mainly consists of two parts: (1) generating initial joint embeddings of labels, terms, and documents; and (2) iterative updating for deriving dimension-aware embeddings.

We measure the running time of these two different steps and show the results in Figure 3.4. As shown, when the corpus size increases from 1,000 to 300,000, the running time of both joint embedding and iterative updating increases roughly linearly.



(a) The running time of the joint embedding step.

(b) The running time of dimension-aware updating step.

Figure 3.4: The efficiency of Doc2Cube on differently sized corpora sampled from NYT.

3.8 SUMMARY

In this chapter, we proposed a method that automatically allocates documents into a multi-dimensional cube structure, by selecting the most appropriate label along each dimension. Departed from existing text classification methods, our proposed method Doc2Cube require no labeled training data but only the surface names of labels. It leverages label names as weak supervision signals and iteratively performs dimension-aware joint embedding of labels, terms, and documents to uncover their semantic similarities. It features a document focalization component that learns dimension-aware document representations by selectively focusing on discriminative terms; as well as a label expansion component that propagates information from label names to other terms for alleviating label sparsity. Our experiments validate the effectiveness of Doc2Cube and its advantages over a comprehensive set of unsupervised and semi-supervised text classification methods for document allocation.

CHAPTER 4: CROSS-DIMENSION PREDICTION IN CUBE SPACE

In the previous part, we have presented algorithms that organize unstructured text data into a multi-dimensional cube structure, by discovering the taxonomic structure for each dimension (Chapter 2) and assigning documents with the most appropriate label along each dimension (Chapter 3).

The multi-dimensional and multi-granular cube structure enables users to flexibly identify relevant data with declarative queries. This, however, is merely a first step in turning unstructured text data into multi-dimensional knowledge. Text data in their raw forms are often noisy, yet what people need are the patterns hidden in the data which are useful for decision making. In this part, we proceed to investigate how to discover multi-dimensional knowledge in the cube space. The high-level purpose of this part is to mine user-selected data from the cube to distill useful multi-dimensional knowledge. In the following two chapters, we will study two important problems: (1) cross-dimension prediction—how to make predictions across dimension? (2) abnormal event detection—how to detect abnormal events in a multi-dimensional cube cell?

To instantiate the above two problems, we assume a three-dimensional 'topic-location-time' cube structure and study how to acquire multi-dimensional knowledge from it. As these three dimensions are fundamental factors underlying human activities, such a cube structure serve as a good proxy for the above two problems. That being said, our developed algorithms can be easily extended to general settings for cube-based multi-dimensional knowledge discovery. Specifically, we study the spatiotemporal activity prediction problem and spatiotemporal event detection problem in the subsequent two chapters. First in Chapter 4, we investigate the spatiotemporal activity prediction problem and propose a semi-supervised multimodal embedding method that makes accurate cross-dimension predictions. Then in Chapter 5, we study the spatiotemporal event detection problem and propose a method that combines multimodal embedding and latent variable model to identify abnormal spatiotemporal events under limited supervision.

4.1 OVERVIEW

Spatiotemporal activity prediction aims at making accurate predictions across three dimensions: location, time, and topic. As shown in Figure 4.1, given a text message and a timestamp, can we predict where the message is created? Conversely, given a location and a specific time, can we predict what are the popular keywords around the location at that time point? Spatiotemporal activity prediction serves as a good proxy for cross-dimension prediction, because answering such questions require modeling the correlations across different dimensions (topic, location, time) and making predictions across them.

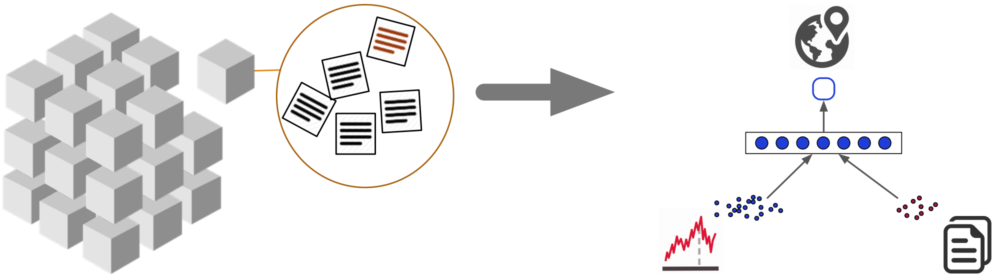


Figure 4.1: An illustration of cross-dimension prediction in a topic-location-time cube.

State-of-the-art methods for this problem employ latent variable models [81, 97, 43]. Specifically, they extend classic topic models such as Latent Dirichlet Allocation by assuming each latent topic variable generates not only textual keywords but also locations and timestamps. The predictive performance of such generative models can be poor in practice. The major reason is because they impose distributional assumptions for the latent topics (*e.g.*, defining the spatial distribution of each topic as Gaussian). Although such assumptions simplify model inference with parameterization, they may not fit real-life data well and are sensitive to noise. Meanwhile, such generative models cannot easily scale up to large data sets.

We propose CrossMap, a multimodal embedding method for spatiotemporal activity prediction. Different from existing generative models, CrossMap models spatiotemporal activities via multimodal embedding—which maps elements from different dimensions (location, time, topic) into the same space with their cross-dimension correlations well preserved. As shown in Figure 4.2, if two elements are highly correlated (*e.g.*, the JFK airport region and the keyword 'flight'), their representations in the latent space tend be close. Compared with existing generative models, the multimodal embedding does not impose any distributional assumptions, and incurs much lower computational cost in the learning process.

To learn quality multimodal embeddings, CrossMap employs a novel semi-supervised learning paradigm. In a considerable number of records, the users explicitly specify the point-of-interest (POI) to indicate their activity categories (*e.g.*, outdoor, shop). The category information can serve as clean and well-structured knowledge, which allows us to better separate the elements with different semantics in the latent space. Our designed semi-supervised paradigm thus leverages such clean category information to guide representation learning to generate better-quality embeddings.

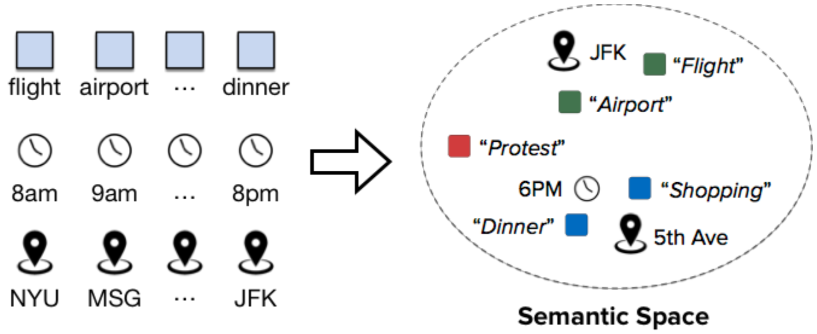


Figure 4.2: An illustration of multimodal embedding for cross-dimension prediction. Items from different dimensions (*e.g.*, location, time, text) are mapped into the same latent space with their correlations preserved. Their representations in the latent space are used for cross-dimension prediction.

Furthermore, in many applications for spatiotemporal activity modeling, the records may arrive continuously instead of being given in one batch. We will show that CrossMap can be easily extended into an online version, which can emphasize more recent records to further improve the performance.

To summarize, our contributions in this chapter include the following:

1. We propose a multimodal embedding method for spatiotemporal activity modeling. Different from existing generative models, CrossMap directly embeds elements from all the dimensions into a low-dimensional vector space to preserve their inter-type interactions. Such a multimodal embedding framework does not impose any distributional assumptions, and incurs much lower computational cost in the learning process.

2. We propose a semi-supervised learning paradigm for learning multimodal embeddings. By linking given records with external knowledges sources (*e.g.*, Wikipedia, POI database), the semi-supervised paradigm effectively incorporates external knowledge, which can serve as guidance to learn quality multimodal embeddings that separate different semantics in the latent space.

3. We propose techniques that perform online updating of CrossMap in situations where new records arrive continuously. Specifically, we explore two strategies: the first imposes life-decaying weights on the records such that recent records are emphasized; while the second treats previous embeddings as prior knowledge, and employs a constrained optimization procedure to obtain updated embeddings. These strategies lead to recency-aware predictive models that further improve the performance of CrossMap.

We evaluate CrossMap on three large-scale social media data sets. Our experiments demonstrate CrossMap outperforms state-of-the-art spatiotemporal activity prediction methods significantly.

4.2 RELATED WORK

State-of-the-art models for spatiotemporal activity modeling [81, 43, 97, 39, 99, 61, 88] adopt latent variable models by extending topic models. Notably, Sizov *et al.* [81] extend LDA [11] by assuming each latent topic has a multinomial distribution over text, and two Gaussians over latitudes and longitudes. They later extend the model to find topics that have complex and non-Gaussian distributions [43]. Yin *et al.* [97] extend PLSA [38] by assuming each region has a normal distribution that generates locations, as well as a multinomial distribution over the latent topics that generate text. While the above models are designed to detect global geographical topics, Hong *et al.* [39] and Yuan *et al.* [99] introduce the user factor in the modeling process such that users' individual-level preferences can be inferred. Our work resembles the studies [81, 43, 97] more because we also model global-level spatiotemporal activities instead of individual-level preferences. That said, our approach for spatiotemporal activity modeling is fundamentally different from these studies. Instead of using latent variable models to bridge different dimension, our method directly maps items from different dimension into the same latent space to preserve their correlations. Such a multimodal embedding method is able to capture cross-dimension correlations in a more direct and scalable way.

4.3 PRELIMINARIES

4.3.1 Problem Description

Let $\mathcal{R}$ be a corpus of activity records in a three-dimensional topic-location-time cube. Each record $r \in \mathcal{R}$ is defined by a tuple $\langle t_r, l_r, m_r \rangle$ where: (1) l_r is a two-dimensional vector that represents the user's location when r is created; (2) t_r is the creating time[1]; and (3) m_r is a bag of keywords denoting the text message of r.

We aim to use a large amount of activity records to model people's activities in the spatiotemporal space. As there are three different dimensions (*i.e.*, location, time, and text) that are intertwined, an effective spatiotemporal activity model should accurately capture

[1] We convert the raw time to the range of [0, 86400] by calculating its offset (in second) w.r.t. 12:00am.

their cross-dimension correlations. Given any two of the three dimensions, the activity model is expected to predict the remaining one. Specifically: (1) What are the typical activities occurring at a specific location and time? (2) Given an activity and time, where does this activity usually take place? and (3) Given an activity and a location, when does the activity usually happen?

4.3.2 CrossMap: Cross-Dimension Prediction via Multimodal Embedding

An effective spatiotemporal activity model should accurately capture the cross-dimension correlations between location, time, and text. For this purpose, existing models [81, 97, 43] assume latent states that generate multi-dimensional observations according to pre-defined distributions (*e.g.*, assuming the location follows Gaussian). Nevertheless, the distributional assumptions may not fit the real data well. For example, beach-related activities are usually distributed along coastlines that have complex shapes, and cannot be well modeled by a Gaussian distribution. Further, learning such generative models are usually time-consuming. Hence, can we capture the cross-dimension correlations more directly?

We develop a joint embedding module to effectively and efficiently capture the cross-dimension correlations between location, time, and text. Different from existing generative models that use latent states to indirectly bridge different data types, our embedding procedure directly captures the cross-dimension correlations by mapping all the items into a common Euclidean space.[2]

A natural design for learning such multimodal embeddings is to use the reconstruction-based strategy: it considers every record as a multi-dimensional relation, and learns the embeddings to maximize the likelihood of observing the given records. However, to learn better quality multimodal embeddings, we observe that a considerable number of records can be linked with external knowledge. For instance, many tweets explicitly specify the points-of-interests (POIs). The category information (*e.g.*, outdoor, shop) of those records, which is clean and well-structured, can serve as useful signals to distinguish different semantics. We thus regard those categories as labels, and design a semi-supervised paradigm to guide the learning of multimodal embeddings.

Figure 4.3 shows the framework of CrossMap. At a high level, CrossMap aims to learn the embeddings L, T, W, and C where: (1) L is the embeddings for regions; (2)

[2] While the keywords can serve as natural embedding elements for the textual part, it is infeasible to embed every location and timestamp as the space and time are continuous. We thus map each timestamp to some hour in a day and use the mapped hour as a basic temporal element, and hence have 24 possible temporal elements in total. Similarly, we partition the geographical space into equal-size regions and consider each region as a basic spatial element.

T is the embeddings for hours; (3) W is the embeddings for keywords; and (4) C is the embeddings for categories. Take L as an example. Each element $\mathbf{v}_l \in L$ is a D-dimensional ($D > 0$) vector, which represents the embedding for region l. As shown, it adopts a semi-supervised paradigm for multimodal embedding. 1) For an unlabeled record r_u, we optimize the embeddings L, T, W for the task of recovering the attributes in r_u; and 2) For a labeled record r_l, we optimize the embeddings L, T, W, C for not only attribute recovery but also activity classification. In such a process, the embeddings of the regions, hours, and keywords are shared across the two tasks, while the category embeddings are specific to the activity classification task. In this way, the semantics of activity categories are propagated from labeled records to unlabeled ones, thereby better separating the elements with different semantics in the latent space.

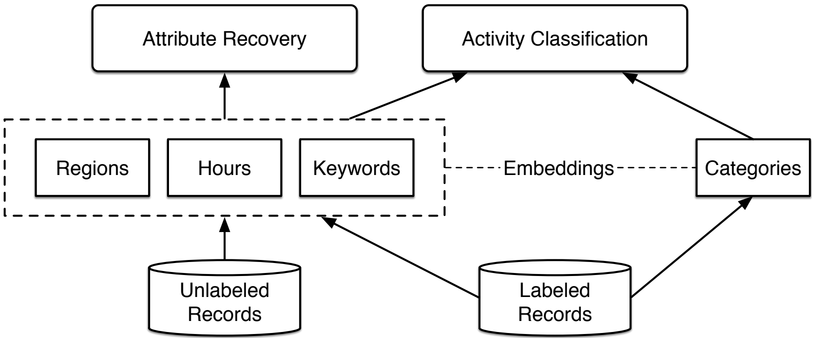


Figure 4.3: The semi-supervised multimodal embedding framework of CrossMap.

Furthermore, we propose strategies that update the embeddings learned by CrossMap in an online manner. When a collection $\mathcal{R}_\Delta$ of new records arrive, our goal is to update the embeddings (L, T, W, C) to accommodate the information contained in $\mathcal{R}_\Delta$. While it is tempting to use $\mathcal{R}_\Delta$ to learn the embeddings from scratch, such an idea not only incurs unnecessary computational overhead, but also leads to overfitting of the new data. To address this issue, we propose two online learning strategies, which effectively incorporates the new records while largely preserving the information encoded in the previous embeddings.

4.4 SEMI-SUPERVISED MULTIMODAL EMBEDDING

In this section, we describe the semi-supervised multimodal embedding module that maps all spatial, temporal, and textual items into a common Euclidean space. Here, a spatial item is a spatial region, a temporal item is a temporal period, and a textual item is a keyword. As shown in Figure 4.3, our semi-supervised multimodal embedding algorithm learns their representations under a multi-task learning setting. By jointly optimizting an unsupervised reconstruction task and a supervised classification task, our algorithm leverages external knowledge to guide the embedding learning process. In what follows, we describe the unsupervised and supervised tasks in Section 4.4.1 and 4.4.2, and then given the optimization procedure in 4.4.3.

4.4.1 The Unsupervised Reconstruction Task

The unsupervised reconstruction task aims at preserving the correlations observed in the given records. The key principle here is to learn the embeddings L, T, W such that the observed relationships among location, time, and text can be reconstructed. We thus define the unsupervised task as an attribute reconstruction task: learn the embeddings L, T, W such that each attribute of a record r can be maximally recovered, assuming the other attributes of r are already observed.

Given a record r, for any attribute $i \in r$ with type X (location, time, or keyword), we model the likelihood of observing i as

$$p(i|r_{-i}) = \exp(s(i, r_{-i})) / \sum_{j \in X} \exp(s(j, r_{-i})),$$

where r_{-i} is the set of all the attributes in r except i, and $s(i, r_{-i})$ is the similarity score between i and r_{-i}.

In the above, the key is how to define $s(i, r_{-i})$. A natural idea is to average the embeddings of the attributes in r_{-i}, and compute $s(i, r_{-i})$ as $s(i, r_{-i}) = \mathbf{v}_i^{\mathrm{T}} \sum_{j \in r_{-i}} \mathbf{v}_j / |r_{-i}|$, where $\mathbf{v}_i$ is the embedding for attribute i. Nevertheless, such a simple definition fails to consider the continuities of the space and time. Take the spatial continuity as an example. According to the first law of geography: *everything is related to everything else, but near things are more related than distant things.* To achieve spatial smoothness, two spatial items that are close to each other should be considered correlated instead of independent. We thus introduce *spatial smoothing* and *temporal smoothing* to capture the spatiotemporal continuities. With the smoothing technique, CrossMap not only maintains local consistency of neighboring regions and periods, but also alleviates data sparsity.

Figure 4.4 illustrates the spatial and temporal smoothing processes. As shown, for each region l, we introduce a pseudo region $\hat{l}$. The embedding of $\hat{l}$ is the weighted average of the embeddings of l and l's neighboring regions, namely

$$\mathbf{v}_{\hat{l}} = (\mathbf{v}_l + \alpha \sum_{l_n \in \mathcal{N}_l} \mathbf{v}_{l_n}) / (1 + \alpha |\mathcal{N}_l|),$$

where $\mathcal{N}_l$ is the set of l's neighboring regions, and α is a constant for spatial smoothing. Similarly, for each period t, we introduce a pseudo period $\hat{t}$, whose embedding is the weighted average of the embeddings of t and t's neighboring periods:

$$\mathbf{v}_{\hat{t}} = (\mathbf{v}_t + \beta \sum_{t_n \in \mathcal{N}_t} \mathbf{v}_{t_n})/(1 + \beta|\mathcal{N}_t|),$$

where $\mathcal{N}_t$ is the set of t's neighboring periods, and β is a temporal smoothing constant. In practice, we find that setting $\alpha = 0.1$ and $\beta = 0.1$ usually leads to satisfactory performance of the model.

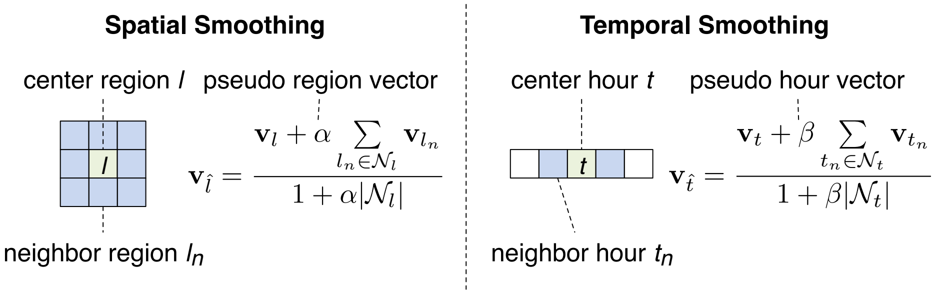

Spatial Smoothing
center region *l*
pseudo region vector
$\mathbf{v}_{\hat{l}} = \frac{\mathbf{v}_l + \alpha \sum_{l_n \in \mathcal{N}_l} \mathbf{v}_{l_n}}{1 + \alpha|\mathcal{N}_l|}$
l
neighbor region *l_n*
Temporal Smoothing
center hour *t*
pseudo hour vector
$\mathbf{v}_{\hat{t}} = \frac{\mathbf{v}_t + \beta \sum_{t_n \in \mathcal{N}_t} \mathbf{v}_{t_n}}{1 + \beta|\mathcal{N}_t|}$
t
neighbor hour *t_n*

Figure 4.4: Spatial and temporal smoothing. For each region (period), we combine it with its neighboring regions (periods) to generate a pseudo region (period).

In addition to the above pseudo region and period embeddings, we also introduce pseudo keyword embeddings for notational ease. Given r_{-i}, its pseudo keyword embedding is defined as:

$$\mathbf{v}_{\hat{w}} = \sum_{w \in \mathcal{N}_w} \mathbf{v}_w / |\mathcal{N}_w|,$$

where $\mathcal{N}_w$ is the set of keywords in r_{-i}. With these pseudo embeddings, we define a smoothed version of $s(i, r_{-i})$ as $s(i, r_{-i}) = \mathbf{v}_i^{\mathrm{T}} \mathbf{h}_i$, where

$$\mathbf{h}_i = \begin{cases} (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/3, & \text{if } i \text{ is a keyword,} \\ (\mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/2, & \text{if } i \text{ is a region,} \\ (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{w}})/2, & \text{if } i \text{ is an period.} \end{cases}$$

Let $\mathcal{R}_{\cup}$ be a collection of records for learning the spatiotemporal activity modeling. The final loss function for the attribute recovery task is simply the negative log-likelihood of observing all the attributes of the records in $\mathcal{R}_{\cup}$:

$$J_{\mathcal{R}_{\cup}} = -\sum_{r \in \mathcal{R}_{\cup}} \sum_{i \in r} \log p(i|r_{-i}). \tag{4.1}$$

4.4.2 The Supervised Classification Task

The supervised classification task leverages external knowledge to guide the multimodal embedding process. After linking with knowledge bases to derive activity category information for records that mention point-of-interest names, we obtain a subset of records $\mathcal{R}_{\cup}$ that become labeled. Now the objective of the supervised classification task learn the embeddings such that the activity categories of those labeled records in $\mathcal{R}_{\cup}$ can be correctly predicted. Let r be a labeled record with category c. The basic intuition is to make c's embedding close to the constituent attributes in r. Based on this intuition, we model the probability of classifying r into category c as:

$$p(c|r) = \exp(s(c, r)) / \sum_{c' \in C} \exp(s(c', r)).$$

For the similarity score $s(c, r)$, we define it in a smoothed way similar to the attribute recovery task. That is, $s(c, r) = \mathbf{v}_c^{\mathrm{T}} \mathbf{h}_r$, where $\mathbf{h}_r = (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/3$.

The objective function of the activity classification task is then the negative log-likelihood of predicting the activities categories for the records in $\mathcal{R}_{\cup}$:

$$J'_{\mathcal{R}_{\cup}} = - \sum_{r \in \mathcal{R}_{\cup}} \log p(c|r). \tag{4.2}$$

4.4.3 The Optimization Procedure

Under the multi-task learning setting (Figure 4.3), we jointly optimize the unsupervised objective $J_{\mathcal{R}_{\cup}}$ and the supervised objective $J'_{\mathcal{R}_{\cup}}$. For efficient optimization, we use stochastic gradient descent (SGD) and negative sampling [63]. Let us first consider the unsupervised loss $J_{\mathcal{R}_{\cup}}$. At each time, we use SGD to sample a record r and an attribute $i \in r$. With negative sampling, we randomly select K negative attributes that have the same type with i but do not appear in r, then the loss function for the selected samples becomes:

$$J_r = - \log \sigma(s(i, r_{-i})) - \sum_{k=1}^{K} \log \sigma(-s(k, r_{-i})),$$

where $\sigma(\cdot)$ is the sigmoid function. The updating rules for $\mathbf{v}_i$, $\mathbf{v}_k$ and $\mathbf{h}_i$ can be obtained by taking the derivatives of J_r:

$$\begin{aligned}
\frac{\partial J_r}{\partial \mathbf{v}_i} &= (\sigma(s(i, r_{-i})) - 1)\mathbf{h}_i, \\
\frac{\partial J_r}{\partial \mathbf{v}_k} &= \sigma(s(i, r_{-i}))\mathbf{h}_i, \\
\frac{\partial J_r}{\partial \mathbf{h}_i} &= (\sigma(s(i, r_{-i})) - 1)\mathbf{v}_i + \sum_{k=1}^{K} \sigma(s(k, r_{-i}))\mathbf{v}_k.
\end{aligned}$$

For any attribute j in $\mathbf{h}_i$, we have $\partial L/\partial \mathbf{v}_j = \partial L/\partial \mathbf{h}_i \cdot \partial \mathbf{h}_i/\partial \mathbf{v}_j$, as $\mathbf{h}_i$ is linear in j, the term $\partial \mathbf{h}_i/\partial \mathbf{v}_j$ is convenient to calculate.

The supervised loss $J'_{\mathcal{R}_\cup}$ can again be efficiently optimized with SGD and negative sampling. In specific, given the labeled record r with the positive category c, we randomly pick a negative category c' satisfying $c' \neq c$. Then the loss function for r in the activity classification task becomes:

$$J_r = -\log \sigma(s(c, r)) - \log \sigma(-s(c', r)).$$

Similar to the derivation in the attribute recovery task, the updating rules of the attributes and categories can be easily obtained by taking the derivatives of J_r and then applying SGD.

4.5 ONLINE UPDATING OF MULTIMODAL EMBEDDING

In this section, we describe the online learning procedures for CrossMap. Given a collection of newly records $\mathcal{R}_\Delta$, the goal is to update the multimodal embeddings L, W, T to capture the information in $\mathcal{R}_\Delta$. The key issue in the above online learning framework is, how to update the embeddings with the goal of effectively incorporating the information in $\mathcal{R}_\Delta$ without overfitting it? We develop two different strategies for this problem: one is life-decaying learning, and the other is constraint-based learning. In what follows, we first describe the details of those two strategies in Section 4.5.1 and 4.5.2. Then we analyze their space and time complexities in Section 4.5.3.

4.5.1 Life-Decaying Learning

Our first strategy, called life-decaying learning, assigns different weights to the records in the data stream such that more recent records receive higher weights. Specifically, for any record r that has appeared in the stream, we set its weight as:

$$w_r = e^{-\tau a_r},$$

where $\tau > 0$ is a decaying parameter, and a_r is r's age with regard to the current time. The general philosophy of such a weighing scheme is to emphasize the recent records and highlight the up-to-date observations of urban activities. On the other hand, the old records in the stream are not completely ignored, they have smaller weights but are still involved in model training to prevent overfitting.

Practically, it is infeasible to store all the records seen so far on account of the massive size of the data stream. For tackling this issue, we maintain a continuously updating buffer $\mathcal{B}$, as shown in Figure 4.5. The buffer $\mathcal{B}$ consists of m buckets $B_0, B_1, \ldots, B_{m-1}$, where all the buckets have the same time span ΔT. For each bucket $B_i (0 \leq i < m)$, we assign an exponentially decaying weight $e^{-\tau i}$ to it, where the weight represents the percentage of samples that we preserve for the respective time span. In other words, the most recent bucket B_0 holds the complete set of records within its time span, the next bucket B_1 holds $e^{-\tau}$ of the corresponding records, and so on. When a new collection of records $\mathcal{R}_\Delta$ arrive, the buffer $\mathcal{B}$ is updated to accommodate $\mathcal{R}_\Delta$. The new records $\mathcal{R}_\Delta$ are fully stored in the most recent bucket B_0. For each other bucket $B_i (i > 0)$, the records in its predecessor B_{i-1} are downsampled with rate $e^{-\tau}$ and then moved into B_i.

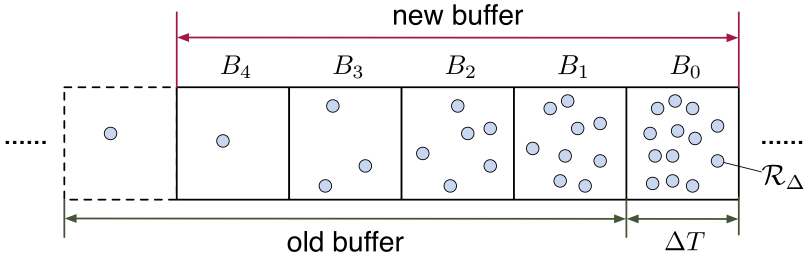


Figure 4.5: Maintaining a buffer $\mathcal{B}$ for life-decaying learning. For any bucket B_i, $e^{-\tau i}$ of the records falling in B_i's time span are preserved for model updating. When new records arrive, $\mathcal{B}$ is updated based on downsampling and shifting.

Algorithm 4.1 sketches the learning procedure of CrossMap with the life-decaying strategy. As shown, when a collection $\mathcal{R}_\Delta$ of new records arrive, we first shift the records from B_{i-1} to B_i by downsampling (lines 1-2), and store $\mathcal{R}_\Delta$ into B_0 in full (lines 3). Once the buffer $\mathcal{B}$ is updated, we randomly sample records from $\mathcal{B}$ (line 4-7) to update the embeddings. First, for any record r, we consider the attribute recovery task and update the embeddings L, T, and W such that the attributes of r can be correctly recovered. Second, if r is labeled, we further update L, T, W, and C such that r can be classified into the correct activity category. Such a process is repeated over $\mathcal{R}_{\cup}$ for a number of epochs before the updated embeddings of L, T, W, and C are output.

Algorithm 4.1: Life-decaying learning of CrossMap.

```
Input: The previous embeddings $L$, $T$, $W$, and $C$.
       A buffer of $m$ buckets $\mathcal{B} = \{B_0, B_1, \ldots, B_{m-1}\}$.
       A collection $\mathcal{R}_{\Delta}$ of new records.
Output: The updated buffer $\mathcal{B}$ and embeddings $L$, $T$, $W$, and $C$.
// Downsampling with rate $e^{-\tau}$.
for $i$ from 1 to $n$ do
    $B_i \leftarrow e^{-\tau}$-downsampled records from $B_{i-1}$;
$B_0 \leftarrow \mathcal{R}_{\Delta}$ ;
$\mathcal{R}_{\cup} \leftarrow \mathcal{B}_{m-1} \cup \mathcal{B}_{m-2} \ldots \cup \mathcal{B}_0$;
for epoch from 1 to $N$ do
    for $i$ from 1 to $|\mathcal{R}_{\Delta}|$ do
        $r \leftarrow$ Randomly sample a record from $\mathcal{R}_{\cup}$;
        // for labeled and unlabeled records
        Update $L$, $T$, and $W$ for recovering $r$'s attributes;
        // for only labeled records
        if $r$ is labeled then
            Update $L$, $T$, $W$, and $C$ for classifying $r$'s activity;
Return $\mathcal{B}$, $L$, $T$, $W$, and $C$;
```

4.5.2 Constraint-Based Learning

The life-decaying strategy relies on the buffer $\mathcal{B}$ to keep old records besides $\mathcal{R}_{\Delta}$, thereby incorporating the information in $\mathcal{R}_{\Delta}$ without overfitting. However, maintaining $\mathcal{B}$ could incur additional space and time overhead. To avoid such overhead, we propose our second strategy named constraint-based learning. The key is to to accommodate the new records $\mathcal{R}_{\Delta}$ by fine-tuning the previous embeddings. During the fine-turning process, we impose the constraint that the updated embeddings do not deviate much from the previous ones. In this way, CrossMap generates embeddings that are optimized for $\mathcal{R}_{\Delta}$ while respecting the prior knowledge encoded in previous embeddings. Algorithm 4.2 sketches the constraint-based learning procedure of CrossMap. As shown, when a collection $\mathcal{R}_{\Delta}$ of new records arrive, we directly use them to update the embeddings for a number of epochs, where the updating for both attribute recovery and activity classification is performed under constraints.

Algorithm 4.2: Constraint-based learning of CrossMap.

```
Input: The previous embeddings $L$, $T$, $W$, and $C$.
       A collection $\mathcal{R}_\Delta$ of new records.
Output: The updated embeddings $L$, $T$, $W$, and $C$.
for epoch from 1 to $N$ do
    Randomly shuffle the records in $\mathcal{R}_\Delta$;
    foreach $r \in \mathcal{R}_\Delta$ do
        Update $L$, $T$, and $W$ for constrained attribute recovery;
        if $r$ is labeled then
            Update $L$, $T$, $W$, and $C$ for constrained activity classification;
Return $L$, $T$, $W$, and $C$;
```

Let us first examine the constraint-based attribute recovery task. Given the new records $\mathcal{R}_\Delta$ and their attributes, our goal is still to recover the attributes of $\mathcal{R}_\Delta$, but now we add a regularization term in the objective to ensure the result embeddings can retain the previous embeddings. In formal, we design the objective function for attribute recovery as:

$$J_{\mathcal{R}_\Delta} = -\sum_{r \in \mathcal{R}_\Delta} \sum_{i \in r} \log p(i|r_{-i}) + \lambda \sum_{i \in L,T,W,C} \|\mathbf{v}_i - \mathbf{v}'_i\|^2,$$

where $\mathbf{v}_i$ is the updated embedding of attribute i, and $\mathbf{v}'_i$ is i's previous embedding learnt before the arrival of $\mathcal{R}_\Delta$. In the above objective function, it is important to note the regularization term $\sum_{i \in L,T,W,C} \|\mathbf{v}_i - \mathbf{v}'_i\|^2$. It prevents the updated embeddings from deviating drastically from the previous embeddings. The value of λ ($\lambda \geq 0$) plays an important role in controlling the regularization strength. When $\lambda = 0$, the embeddings are purely optimized for fitting $\mathcal{R}_\Delta$; when $\lambda = \infty$, the learning process completely ignore the new records and all the embeddings remain unchanged.

We still combine stochastic gradient descent (SGD) and negative sampling to optimize the above objective function. Consider a record r and an attribute $i \in r$. With negative sampling, we randomly select a set of K negative attributes N_i^-, then the objective for the selected samples is:

$$J_r = -\log \sigma(s(i, r_{-i})) - \sum_{k \in N_i^-} \log \sigma(-s(k, r_{-i})) + \lambda \sum_{i \in \{r\} \cup N_i^-} \|\mathbf{v}_i - \mathbf{v}'_i\|^2.$$

The updating rules for different attributes can be easily obtained by taking the derivatives of J_r. Taking attribute i as an example, the corresponding derivative and updating rule are given by

$$\frac{\partial J_r}{\partial \mathbf{v}_i} = (\sigma(s(i, r_{-i})) - 1)\mathbf{h}_i + 2\lambda(\mathbf{v}_i - \mathbf{v}'_i),$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \eta(1 - \sigma(s(i, r_{-i})))\mathbf{h}_i - 2\eta\lambda(\mathbf{v}_i - \mathbf{v}'_i),$$

where η is the learning rate for SGD.

By examining the updating rule for i, we can see the constraint-based strategy enjoys two attractive properties: 1) It tries to make i's embedding close to the average embedding (*i.e.*, $\mathbf{h}_i$) of the other attributes in r. Especially when the current embeddings do not produce high similarity score between i and r_i, *i.e.*, $s(i, r_{-i})$ is small, the updating takes an aggressive step to push $\mathbf{v}_i$ close to $\mathbf{h}_i$; and 2) With the term $-2\eta\lambda(\mathbf{v}_i - \mathbf{v}'_i)$, the learnt embeddings are constrained to preserve the information encoded in the previous embeddings. In specific, if the learnt embedding $\mathbf{v}_i$ deviates from the previous embedding $\mathbf{v}'_i$ too much, the updating rule would subtract the difference to some extent and drag $\mathbf{v}_i$ towards $\mathbf{v}'_i$.

We proceed to examine the activity classification task under the constraint-based strategy. The overall objective is to maximize the log-likelihood of predicting the activities categories for $\mathcal{R}_\Delta$ while minimizing the deviation from the previous embeddings. Using SGD, for any record r with activity category c, we generate a negative category c', and then define the objective as

$$J_r = -\log\sigma(s(c, r)) - \log\sigma(-s(c', r)) + \lambda \sum_{c \in \{c, c'\}} \|\mathbf{v}_c - \mathbf{v}'_c\|^2.$$

Again, the updating rules for the different variables in the above objective can be easily obtained by taking the derivatives of J_r, we omit the details here to save space.

4.5.3 Complexity Analysis

Space complexity. With either life-decaying learning or constraint-based learning, we need to maintain the embeddings of all the regions, periods, keywords, and categories. Let D be the dimension of the latent space. Then the space cost for maintaining those embeddings is $O(D(|L| + |T| + |W| + |C|))$, where $|L|$, $|T|$, $|W|$, and $|C|$ are the numbers of regions, periods, keywords, and categories, respectively. In addition, both strategies need to keep a collection of training records. For the constraint-based learning, the space cost of this part is $O(|\mathcal{R}_{max}|)$ where $|\mathcal{R}_{max}|$ is the maximum number of new records that arrive at one time. The life-decaying learning strategy needs to keep the new records as well as some old ones. As it imposes exponentially decaying sampling rates on the buckets, the space cost for maintaining those records is

$$O(|\mathcal{R}_{max}|(1 + e^{-\tau} + \ldots + e^{-(m-1)\tau})) = O(|\mathcal{R}_{max}|\frac{1 - e^{-m\tau}}{1 - e^{-\tau}}).$$

Time complexity. We first analyze the time complexity of the constraint-based learning strategy. Examining Algorithm 4.2, one can see that the constraint-based strategy needs to go over $\mathcal{R}_\Delta$ for N epochs and process every record in $\mathcal{R}_\Delta$ exactly once in each epoch. Hence, the time complexity is $O(NDM^2|\mathcal{R}_{max}|)$, where M is the maximum number of attributes in any record. Since N and D are fixed beforehand, and M is usually sufficiently small, CrossMap scales roughly linearly with $\mathcal{R}_\Delta$. Similarly, the time complexity of the life-decaying strategy is derived as $O(NDM^2|\mathcal{R}_{max}|+|\mathcal{R}_\cup|)$, where $|\mathcal{R}_\cup| = |\mathcal{R}_{max}|(1-e^{-m\tau})/(1-e^{-\tau})$.

4.6 EXPERIMENTS

In this section, we empirically evaluate CrossMap to examine the following questions about it: (1) Can it better capture the correlations between regions, periods, and activities compared with existing methods? (2) How is the performance of online learning modules? and (3) Are the learnt embeddings useful for downstream applications?

4.6.1 Experimental Setup

Data Sets

Our experiments are based on the following three real-life data sets:

1. The first dataset, called LA, contains ∼1.10 million geo-tagged tweets published in Los Angeles. We crawled the LA data set by monitoring the Twitter Streaming API[3] during 2014.08.01 – 2014.11.30 and continuously gathering the geo-tagged tweets in the bounding box of LA. In addition, we crawled all the POIs in LA through Foursquare's public API[4]. We are able to link ∼0.11 million of the crawled tweets to the POI database and assign them to one of the following categories: Food, Shop & Service, Travel & Transport, College & University, Nightlife Spot, Residence, Outdoors & Recreation,

[3] https://dev.twitter.com/streaming/overview

[4] https://developer.foursquare.com/

Arts & Entertainment, Professional & Other Places. We preprocessed the raw data as follows. For the text part, we removed user mentions, URLs, stopwords, and the words that appear less than 100 times in the corpus. For the space and time, we partitioned the LA area into small grids with size 300m*300m, and broke the one-day period into 24 one-hour windows.

2. The second dataset, called NY, is also collected from Twitter and then linked with Foursquare. It consists of ∼1.20 million geo-tagged tweets in New York City during 2014.08.01 - 2014.11.30, and we are able to link ∼0.10 million of them with Foursquare POIs. The preprocessing steps are the same as LA.

3. The third dataset, called 4SQ, is collected from Foursquare. It consists of around 0.7 million Foursquare checkins posted in New York during 2010.08 - 2011.10. This dataset is mainly used to evaluate the performance of CrossMap for the downstream task of activity classification. Similarly, we removed user mentions, URLs, stopwords, and the words that appear less than 100 times in the corpus.

Baselines

We compare our proposed CrossMap model with the following baseline methods:

- LGTA [97] is a geographical topic model that assumes a number of latent spatial regions — each described by a Gaussian. Meanwhile, each region has a multinomial distribution over the latent topics that generate keywords.

- MGTM [43] is a state-of-the-art geographical topic model based on the multi-Dirichlet process. It is capable of finding geographical topics with non-Gaussian distributions, and does not require a pre-specified number of topics.

- Tensor [35] builds a 4-D tensor to encode the co-occurrences among location, time, text, and category. It then factorizes the tensor to obtain low-dimensional representations of all the elements.

- SVD first constructs the co-occurrence matrices between each pair of location, time, text, and category, and then performs Singular Value Decomposition on the matrices.

- TF-IDF constructs the co-occurrence matrices between each pair of location, time, text, and category. It then computes the tf-idf weight for each entry in the matrix by treating rows as documents and columns as words.

Similar to our CrossMap method, Tensor, SVD, and TF-IDF also rely on space and time partitioning to obtain regions and time periods. We use the same partitioning granularity for those methods to ensure fair comparison. Besides them, we also implement a weakened variant of CrossMap to validate the effectiveness of the semi-supervised paradigm: CrossMap-Unsupervised is a variant of CrossMap that does not leverages the category information as distant supervision. In other words, CrossMap-Unsupervised only trains the embeddings in an unsupervised fashion. Besides CrossMap-Unsupervised, for the two online learning version of CrossMap, we refer to the life-decaying one as CrossMap-OL-Decay, and the constraint-based one as CrossMap-OL-Cons.

Parameter Settings

There are five major parameters in CrossMap: 1) the latent embedding dimension D; 2) the number of epochs N; 3) the SGD learning rate η; 4) the spatial smoothing constant α; and 5) the temporal smoothing constant β. By default, we set $D = 300$, $N = 50$, $\eta = 0.01$, and $\alpha = \beta = 0.1$.

Meanwhile, for the two online learning variants of CrossMap, life-decaying and constraint-based strategies, there are a few additional parameters. The life-decaying strategy has its specific parameters, the decaying rate τ and the number of buckets m; and the constraint-based strategy also has its own parameter, the regularization strength λ. We set their default values to $\tau = 0.01$, $m = 500$, and $\lambda = 0.3$.

In LGTA, there are two major parameters, the number of regions R, and the number of latent topics Z. After careful tuning, we set $R = 300$ and $Z = 10$. MGTM is a non-parametric method that involves several hyper-parameters. We set the hyper-parameters following the original paper [43]. For Tensor and SVD, we set the latent dimension as $D = 300$ to compare with CrossMap fairly.

Evaluation Tasks and Metrics

In our quantitative studies, we investigate two types of spatiotemporal activity prediction tasks. The first is to *predict locations for a given textual query.* Specifically, recall that each record reflects a user's activity with three attributes: a location l_r, a timestamp t_r, and a bag of keywords m_r. In the location prediction task, the input is the timestamp t_r and the keywords m_r, and the goal is to accurately pinpoint the ground-truth location from a pool of candidates. We predict the location at two different granularities: 1) coarse-grained region prediction is to predict the ground-truth region that r falls in; and 2) fine-grained POI prediction is to predict the ground-truth POI that r corresponds to. Note that fine-grained POI prediction is only evaluated on the tweets that have been linked with Foursquare. The second task is to *predict activities for a given location query*. In this task, the input is the timestamp t_r and the location l_r, and the goal is to pinpoint the ground-truth activities at two different granularities: 1) coarse-grained category prediction is to predict the ground-truth activity category of r. Again, such a coarse-grained activity prediction is performed only on the tweets that have been linked with Foursquare; and 2) fine-grained keyword prediction is to predict the ground-truth message m_r from a candidate pool of messages.

To summarize, we study four cross-dimension prediction sub-tasks in total: 1) region prediction; 2) POI prediction; 3) category prediction; and 4) keyword prediction. For each prediction task, we generate a candidate pool by mixing the ground truth with a set of M negative samples. Take region prediction as an example. Given the ground-truth region l_r, we mix l_r with M randomly chosen regions. Then we try to pinpoint the ground truth from the size-$(M+1)$ candidate pool by ranking all the candidates. Intuitively, the better a model captures the patterns underlying people's activities, the more likely it ranks the ground truth to top positions. We thus use Mean Reciprocal Rank (MRR) to quantify the effectiveness of a model. Given a set Q of queries, the MRR is defined as: $\text{MRR} = (\sum_{i=1}^{|Q|} 1/\text{rank}_i)/|Q|$, where rank_i is the ranking of the ground truth for the i-th query.

We describe the ranking procedures of different methods as follows. Again consider region prediction as an example. For CrossMap, we compute the average cosine similarity of each candidate region to the observed elements (time and keywords), and rank them in the descending order of the similarity; for LGTA and MGTM, we compute the likelihood of observing each candidate given the keywords, and rank the candidates by likelihood; for Tensor and SVD, we use the decompositions to reconstruct densified co-occurrence tensor and matrices, and then predict the tensor/matrix entries to rank the candidates; for TF-IDF, we rank the candidates by computing average tf-idf similarities.

4.6.2 Quantitative Comparison

Table 4.1 and 4.2 report the quantitative results of different methods for location and activity predictions, respectively. As shown, on all of the four sub-tasks, CrossMap and its variants achieve much higher MRRs than the baseline methods. Compared with the two geographical topic models (LGTA and MGTM), CrossMap yields as much as 62% performance improvement for location prediction, and 83% for activity prediction. There are three factors for explaining the performance gap: (1) Neither LGTA nor MGTM models the time factor, and thus fails to leverage the time information for prediction; (2) CrossMap emphasizes recent records to capture up-to-date spatiotemporal activities, while LGTA and MGTM work in batch and treat all training instances equally; and (3) Instead of using generative models, CrossMap directly maps different data types into a common space to capture their correlations more directly.

Tensor, SVD, and TF-IDF have better performance than LGTA and MGTM by modeling time and category, yet CrossMap still outperforms them by large margins. Interestingly, TF-IDF turns out to be a strong baseline, demonstrating the effectiveness of the tf-idf similarity for the prediction tasks. SVD and Tensor can effectively recover the co-occurrence matrices and tensor by filling in the missing values. However, the raw co-occurrence seems a less effective relatedness measure for location and activity prediction.

Table 4.1: The MRRs of different methods for location prediction. For each test tweet, we assume its timestamp and keywords are observed, and perform location prediction at two granularities: 1) *region prediction* retrieves the ground-truth region; and 2) *POI prediction* retrieves the ground-truth POI (for Foursquare-linked tweets).

Method	Region prediction		POI prediction	
	LA	NY	LA	NY
LGTA	0.3583	0.3544	0.5889	0.5674
MGTM	0.4007	0.391	0.5811	0.553
Tensor	0.3592	0.3641	0.6672	0.7399
SVD	0.3699	0.3604	0.6705	0.7443
TF-IDF	0.4114	0.4605	0.719	0.776
CrossMap-Unsupervised	0.5373	0.5597	0.7845	0.8508
CrossMap	0.5586	0.5632	0.8155	0.8712
CrossMap-OL-Cons	0.5714	0.5864	0.8311	**0.8896**
CrossMap-OL-Decay	**0.5802**	**0.5898**	**0.8473**	0.885

Comparing the variants of CrossMap, we see clear performance gaps between CrossMap-Unsupervised and CrossMap, particularly for the category prediction task. The major difference between CrossMap-Unsupervised and CrossMap is that, CrossMap-Unsupervised just treats category descriptions as keywords, while CrossMap uses activity categories as labels to guide embedding. This phenomenon shows the semi-supervised paradigm indeed helps propagate external category knowledge into the embedding process to generate high-quality multimodal embeddings.

CrossMap-OL-Decay and CrossMap-OL-Cons achieve even better prediction performance than CrossMap. Although the three variants all use semi-supervised training, CrossMap treats all the training instances equally whereas the other two work online and emphasize recent instances more. This fact verifies that there are notable evolutions un-

Table 4.2: The MRRs of different methods for activity prediction. For each test tweet, we assume its location and timestamp are observed, and predict activities at two granularities: 1) *category prediction* predicts the ground-truth category (for Foursquare-linked tweets); and 2) *keyword prediction* retrieves the ground-truth message.

Method	Category prediction		Keyword prediction	
	LA	NY	LA	NY
LGTA	0.4409	0.4527	0.3392	0.3425
MGTM	0.4587	0.464	0.3501	0.343
Tensor	0.8635	0.7988	0.4004	0.3744
SVD	0.8556	0.7826	0.4098	0.3728
TF-IDF	0.9137	0.8259	0.5236	0.4864
CrossMap-Unsupervised	0.6225	0.5874	0.5693	0.5538
CrossMap	0.9056	0.8993	0.5832	0.5793
CrossMap-OL-Cons	0.92	0.8964	0.6097	0.5887
CrossMap-OL-Decay	**0.9272**	**0.9026**	**0.6174**	**0.5928**

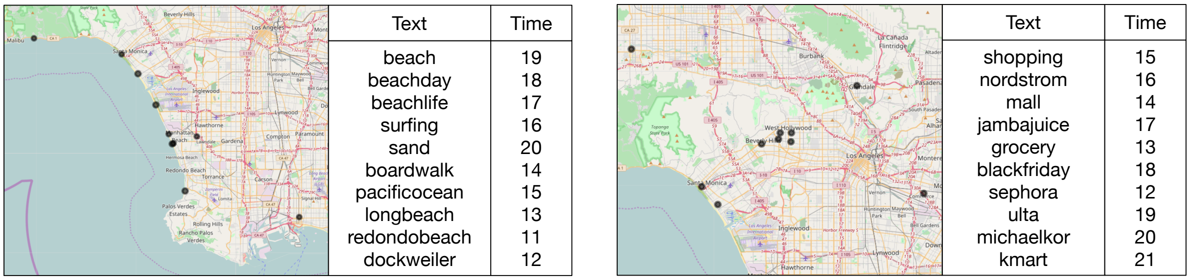
derlying people's activities in the four-month time period, and the recency-aware nature of CrossMap-OL-Decay and CrossMap-OL-Cons effectively captures such evolutions to better suit users' prediction needs. Finally, examining the performance of CrossMap-OL-Decay and CrossMap-OL-Cons, we find that the life-decaying learning strategy performs slightly better than the constraint-based one in practice, but at the cost of extra space and time overhead.

4.6.3 Case Studies

In this subsection, we perform a set of case studies to examine how well CrossMap makes predictions across dimensions, and whether CrossMap can capture the dynamic evolutions of spatiotemporal activities. Specifically, we perform one-pass training of CrossMap on LA and NY, and launch a bunch of queries at different stages. For each query, we retrieve the top-10 most similar elements with different types from the entire search space.

Textual Queries

Figure 4.6(a) and 4.6(b) show the results when we query with the keywords 'beach' and 'shopping'. One can see the retrieved items in each type are quite meaningful: (1) For the query 'beach', the top locations mostly fall in famous beach areas in Los Angeles; the top keywords reflect people's activities on the beach, such as 'sand' and 'boardwalk'; the top time slots are in the late afternoon, which are indeed good time to enjoy the beach life. (2) For the query 'shopping', the retrieved locations are at popular malls and outlets in Los Angeles; the keywords (*e.g.*, 'nordstrom', 'mall', 'blackfriday') are either brand names or shopping-related nouns; and the time slots are mostly around 3pm in the afternoon, matching people's real-life shopping patterns intuitively.



Text	Time
beach	19
beachday	18
beachlife	17
surfing	16
sand	20
boardwalk	14
pacificocean	15
longbeach	13
redondobeach	11
dockweiler	12

(a) Query = 'beach'

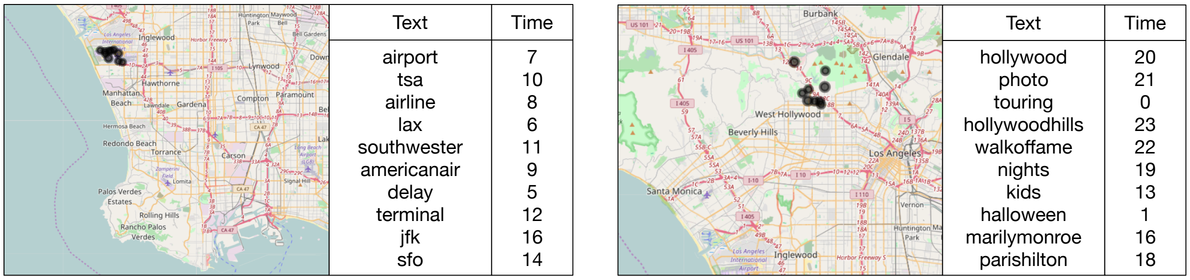
Text	Time
shopping	15
nordstrom	16
mall	14
jambajuice	17
grocery	13
blackfriday	18
sephora	12
ulta	19
michaelkor	20
kmart	21

(b) Query = 'shopping'

Figure 4.6: Two textual queries and the top ten results returned by CrossMap.

Spatial Queries

Figure 4.7(a) and 4.7(b) show the results for two spatial queries: (1) the location of the LAX airport; and (2) the location of Hollywood. Again, we can see the retrieved top spatial, temporal, and textual items are closely related to airport and Hollywood, respectively. For instance, given the query at LAX, the top keywords are all meaningful concepts that reflect flight-related activities, such as 'airport', 'tsa', and 'airline'.



Text	Time
airport	7
tsa	10
airline	8
lax	6
southwester	11
americanair	9
delay	5
terminal	12
jfk	16
sfo	14

(a) Query = '(33.9424, -118.4137)' (LAX Airport)

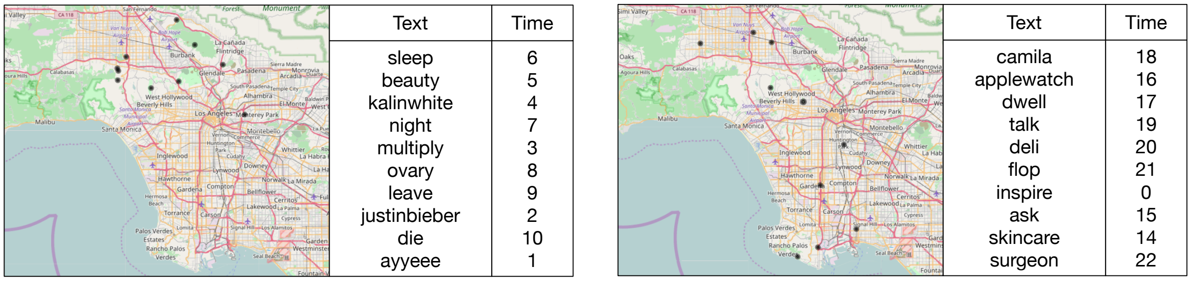
Text	Time
hollywood	20
photo	21
touring	0
hollywoodhills	23
walkoffame	22
nights	19
kids	13
halloween	1
marilymonroe	16
parishilton	18

(b) Query = '(34.0928, -118.3287)' (Hollywood)

Figure 4.7: Two spatial queries and the top ten results returned by CrossMap.

Temporal Queries

Figure 4.8(a) and 4.8(b) show the results when we query with two timestamps: 6am and 6pm. We find the results in each list make practical sense (*e.g.*, keywords like 'sleep' are ranked high for the query '6am'), but are less coherent compared with those of spatial and textual queries. This phenomenon is reasonable, as people's activities in the same time slot could vary greatly. For instance, it is common that people have different activities at 6pm, ranging from having food to shopping and working. Therefore, the temporal signal alone cannot easily determine people's activities or locations.



Text	Time
sleep	6
beauty	5
kalinwhite	4
night	7
multiply	3
ovary	8
leave	9
justinbieber	2
die	10
ayyeee	1

(a) Query = '6am'

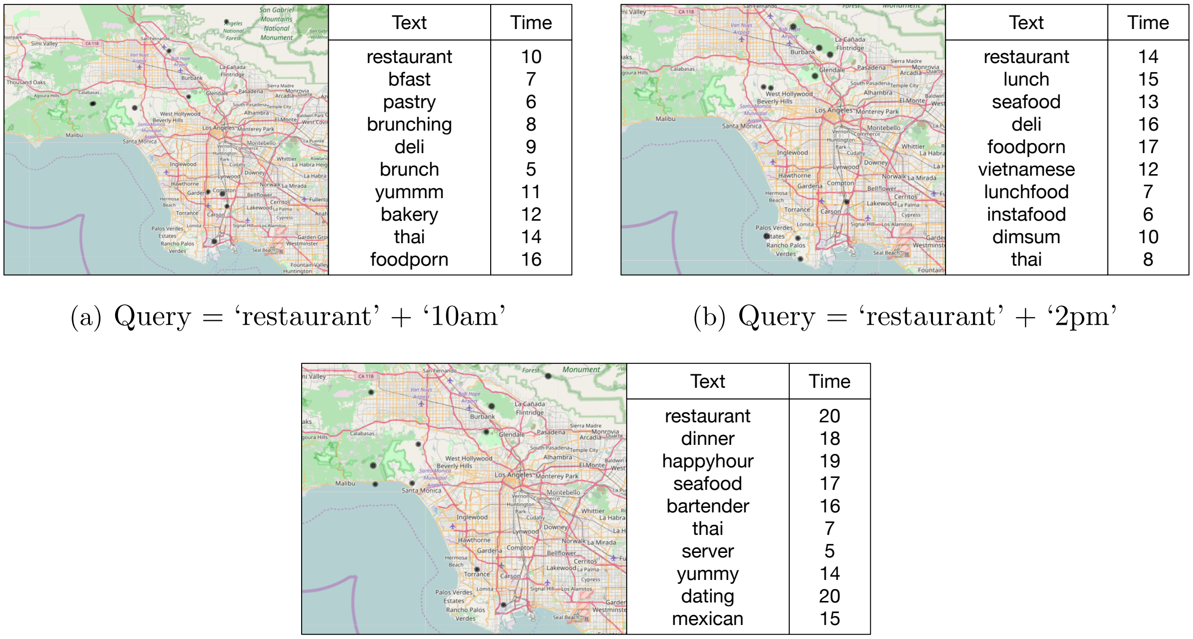
Text	Time
camila	18
applewatch	16
dwell	17
talk	19
deli	20
flop	21
inspire	0
ask	15
skincare	14
surgeon	22

(b) Query = '6pm'

Figure 4.8: Two temporal queries and the top ten results returned by CrossMap.

Temporal-Textual Queries

Figure 4.9(a), 4.9(b), and 4.9(c) show some temporal-textual queries to demonstrate the temporal dynamics of urban activities. As we fix the query keyword as 'restaurant' and vary the time, the retrieved items change obviously. Examining the top keywords, we can see the query '10am' leads to many breakfast-related keywords in the list, such as 'bfast' and 'brunch'. In contrast, the query '2pm' retrieves many lunch-related ones while '8pm' retrieves dinner-related ones. Also, the top locations for '10am' and '2pm' mostly fall in working areas, while the ones for '8pm' distribute more in residential areas. Those results clearly show that the time factor plays an important role in determining people's activities, and CrossMap effectively captures such subtle dynamics. Our spatial-temporal and spatial-textual queries lead to similar observations, we omit them to save space.



Text	Time
restaurant	10
bfast	7
pastry	6
brunching	8
deli	9
brunch	5
yummm	11
bakery	12
thai	14
foodporn	16

(a) Query = ‘restaurant’ + ‘10am’

Text	Time
restaurant	14
lunch	15
seafood	13
deli	16
foodporn	17
vietnamese	12
lunchfood	7
instafood	6
dimsum	10
thai	8

(b) Query = ‘restaurant’ + ‘2pm’

Text	Time
restaurant	20
dinner	18
happyhour	19
seafood	17
bartender	16
thai	7
server	5
yummy	14
dating	20
mexican	15

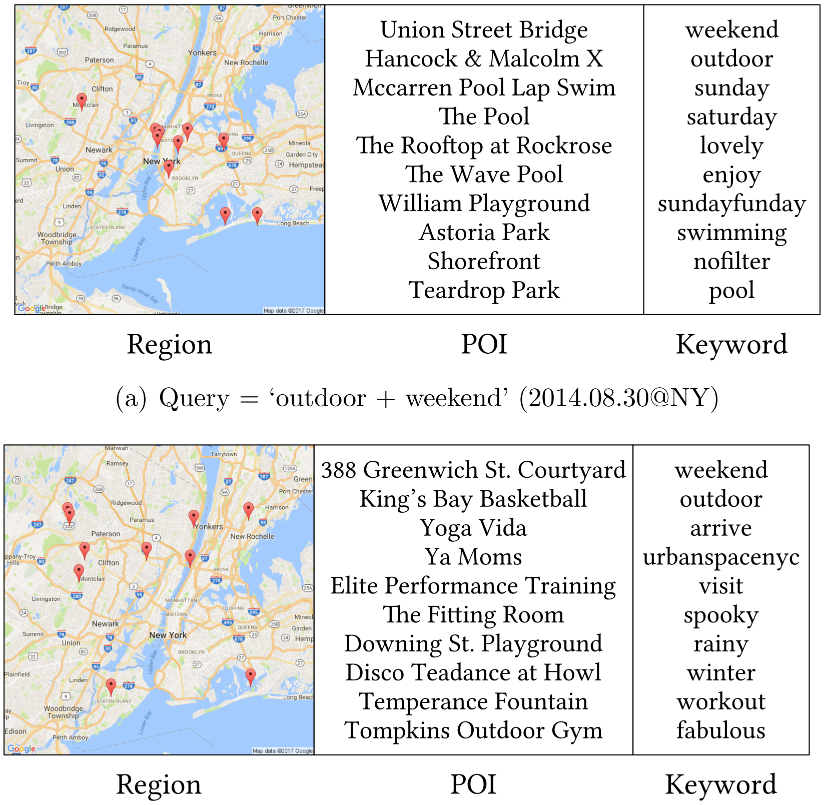
(c) Query = ‘restaurant’ + ‘8pm’

Figure 4.9: Three temporal-textual queries and the top ten results returned by CrossMap.

Dynamic Queries

In this subsection, we examine how the online versions of CrossMap can capture the dynamic evoluations of spatiotemporal activities. Figure 4.10(a) and 4.10(b) show the results for the query ‘outdoor + weekend’ issued on NY for two different days. Interestingly, the results obtained for the two days both relate to ‘outdoor’, but exhibit clear evolutions. While the results for 2014.08.30 contain many swimming-related activities, those for 2014.10.30 are mostly fitness venues. Based on such phenomena, one can clearly see that CrossMap captures not only cross-dimension correlations but also the temporal evolutions underlying spatiotemporal activities.

Figure 4.11(a) and 4.11(b) illustrate the evolutions of two spatial queries: 1) the Metlife Stadium; and 2) the Universal Studio. Again, we can see the results well match the query location and meanwhile reflect activity dynamics clearly. For the Metlife Stadium query, the top keywords evolve from concert-related ones to football-related ones. It is because the NFL season opens in early September, and people start visiting the stadium to watch the games of the Giants and the Jets. For the Universal Studio query, we intentionally include Halloween and Thanksgiving in the query days. In such a setting, we find the latter two lists contain holiday-specific keywords, verifying the capability of CrossMap for capturing



POI	Keyword
Union Street Bridge	weekend
Hancock & Malcolm X	outdoor
Mccarren Pool Lap Swim	sunday
The Pool	saturday
The Rooftop at Rockrose	lovely
The Wave Pool	enjoy
William Playground	sundayfunday
Astoria Park	swimming
Shorefront	nofilter
Teardrop Park	pool

(a) Query = 'outdoor + weekend' (2014.08.30@NY)

POI	Keyword
388 Greenwich St. Courtyard	weekend
King's Bay Basketball	outdoor
Yoga Vida	arrive
Ya Moms	urbanspacenyc
Elite Performance Training	visit
The Fitting Room	spooky
Downing St. Playground	rainy
Disco Teadance at Howl	winter
Temperance Fountain	workout
Tompkins Outdoor Gym	fabulous

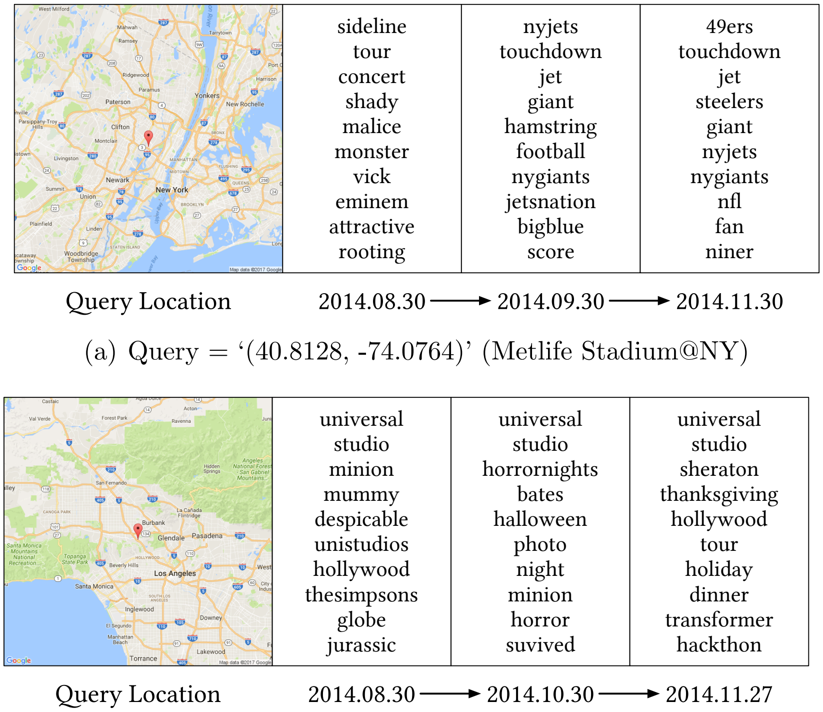
(b) Query = 'outdoor + weekend' (2014.10.30@NY)

Figure 4.10: Illustrative cases demonstrating how CrossMap captures dynamic evoluations. Figure 4.10(a) and 4.10(b) are textual queries issued on different days (*i.e.*, the dates in bracket). For each query, we use the trained model on the issuing day to retrieve ten most similar regions (the markers in the map denote the region centers), POIs, and keywords, based on cosine similarities of the embeddings.

the most recent activity patterns.

4.6.4 Effects of Parameters

In this subsection, we study the effects of different parameters on the performance of CrossMap. Figure 4.12(a) and 4.12(b) show the effects of the latent dimension D and the number of epochs N. Since the trends are very similar for fine-grained and coarse-grained prediction tasks, we omit the results for POI prediction and category prediction for clarity. As shown in Figure 4.12(a), the MRRs of both methods keep increasing with D and gradually converge. This phenomenon is expected because a larger D leads to a more expressive model that can capture latent semantics more accurately. From Figure 4.12(b), one can see as N increases, the performance of CrossMap also increases first and



Query Location	2014.08.30	2014.09.30	2014.11.30
	sideline	nyjets	49ers
	tour	touchdown	touchdown
	concert	jet	jet
	shady	giant	steelers
	malice	hamstring	giant
	monster	football	nyjets
	vick	nygiants	nygiants
	eminem	jetsnation	nfl
	attractive	bigblue	fan
	rooting	score	niner

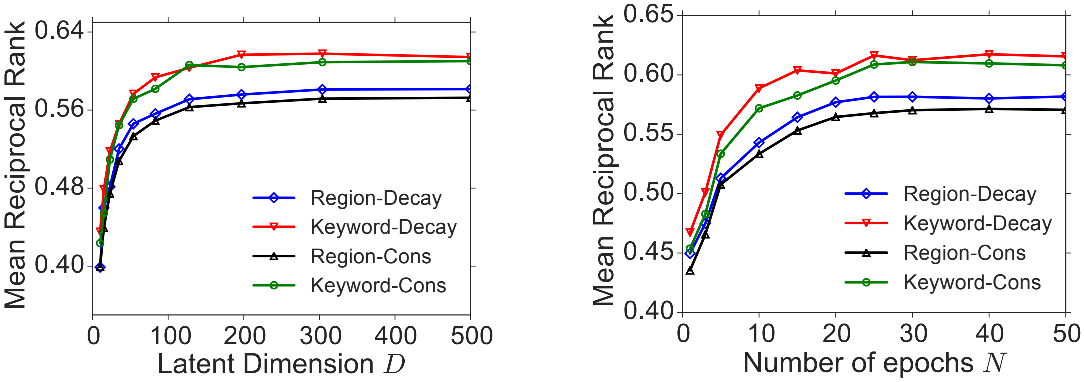
(a) Query = '(40.8128, -74.0764)' (Metlife Stadium@NY)

Query Location	2014.08.30	2014.10.30	2014.11.27
	universal	universal	universal
	studio	studio	studio
	minion	horrornights	sheraton
	mummy	bates	thanksgiving
	despicable	halloween	hollywood
	unistudios	photo	tour
	hollywood	night	holiday
	thesimpsons	minion	dinner
	globe	horror	transformer
	jurassic	suvived	hackthon

(b) Query = '(34.1381, -118.3534)' (Universal Studio@LA)

Figure 4.11: Two spatial queries at the Metlife Stadium and Universal Studio. For each query, we retrieve ten most similar keywords on different days.

finally becomes stable: when N is small, the updated embeddings do not incorporate the new information sufficiently; when N is large, both the life-decaying and constraint-based strategies can effectively prevent CrossMap from overfitting the new records.

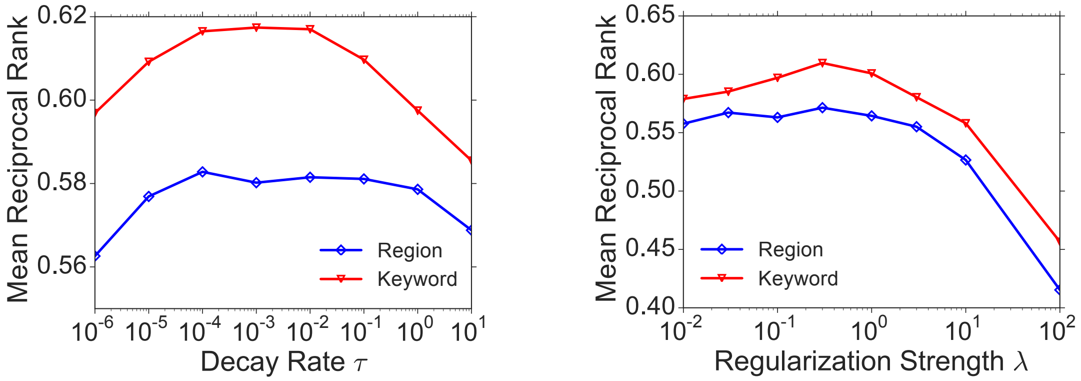


(a) Effect of D.

(b) Effect of N.

Figure 4.12: Parameter study on LA. Figure 4.12(a) and 4.12(b) show the effects of the latent dimension D and the number of epochs N on CrossMap-OL-Decay and CrossMap-OL-Cons.

Figure 4.13(a) and 4.13(b) depict the effects of τ and λ on the performance of the two online learning strategies, respectively. As shown, for life-decaying learning, its performance first increases with τ, then becomes stable, and finally deteriorates. The reason is two-fold: 1) a too small τ makes the buffer contain too many old records in the history, thus diluting the most recent information; and 2) a too large τ leads to a buffer that contains only recent records, making the result model suffer from overfitting. The effect of λ on the constraint-based learning is similar. A too large λ causes underfitting of the new records, while a too small λ causes overfitting. Besides the above parameters, we have also studied the effects of the smoothing parameters α and β, and found that the performance of CrossMap varied no more than 3% when α and β are set to the range $[0.05, 0.5]$, thus we omit the results to save space.



(a) Effect of τ.

(b) Effect of λ.

Figure 4.13: Parameter study on LA. Figure 4.13(a) shows the effect of the decaying rate τ on CrossMap-OL-Decay. Figure 4.13(b) shows the effect of the regularization strength λ on CrossMap-OL-Cons.

4.6.5 Downstream Application

We choose activity classification as an example application to demonstrate the usefulness of the multimodal embeddings learnt by CrossMap. In 4SQ, each checkin belongs to one of the following nine categories: Food, Shop & Service, Travel & Transport, College & University, Nightlife Spot, Residence, Outdoors & Recreation, Arts & Entertainment, Professional & Other Places. We use those categories as activity labels, and learn classifiers to predict the label for any given check-in. After random shuffling, we use 80% checkins for training, and the rest 20% for testing. Given a checkin r, any of the methods introduced in Section 4.6.1 (including CrossMap) can obtain three vector representations for the location, time, and text message; we concatenate the three vectors as the feature vector of a checkin.

After feature transformation, we train a multi-class logistic regression for each method. We measure the classification performance of each method with the Micro-F1 metric and report the results in Figure 4.14. As shown, CrossMap outperform other methods significantly. Even with a simple linear classification model, the absolute F1 score can reach as high as 0.843. Such results show that the embeddings obtained by CrossMap can well distinguish the semantics of different categories. Figure 4.15 further verifies this fact. Therein, we choose three categories and use t-SNE [59] to visualize the feature vectors. One can observe that the learnt embeddings of CrossMap result in much clearer inter-class boundaries compared to **LGTA**.

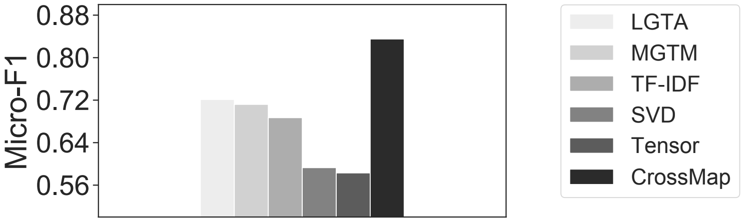
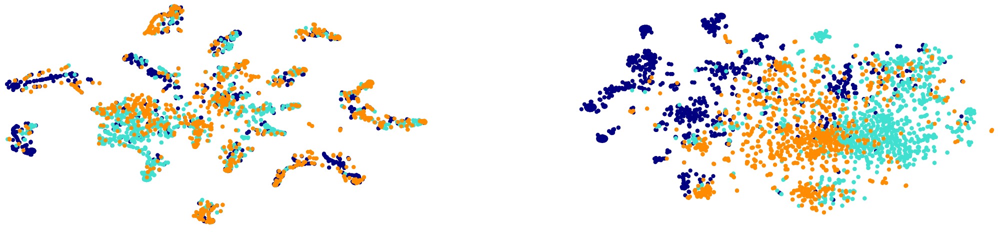


Figure 4.14: Activity classification performance on 4SQ.



(a) LGTA

(b) CrossMap

Figure 4.15: Visualizing the feature vectors generated by LGTA and CrossMap for three activity categories: 'Food' (cyan), 'Travel & Transport' (blue), and 'Residence' (orange). The feature vector of each 4SQ checkin is mapped to a 2D point with t-SNE [59].

4.7 SUMMARY

In this chapter, we have studied the problem of spatiotemporal activity prediction, which serves as a proxy for cross-dimension prediction in the cube space. Towards this end, we proposed CrossMap, a semi-supervised multimodal embedding method. CrossMap embeds items from different dimensions into the same latent space, while leveraging external knowledge as guidance with a semi-supervised paradigm. Further, we proposed strategies that allow CrossMap to learn from continuous data and emphasize the most recent records. Our experiments on real data have shown the effectiveness of the semi-supervised multimodal embedding paradigm and the proposed online learning strategies.

CHAPTER 5: EVENT DETECTION IN CUBE SPACE

In this chapter, we study how to extract abnormal events in the cube space. As mentioned earlier, we examine an instantiated 'topic-location-time' cube and focus on detecting abnormal spatiotemporal events in cube cells. We will describe our method that discovers spatiotemporal events accurately by combining latent variable models and multimodal embeddings.

5.1 OVERVIEW

A spatiotemporal event (*e.g.*, protest, crime, disaster) is an abnormal activity bursted in a local area and within specific duration while engaging a considerable number of participants. Detecting spatiotemporal events at their onsets is in pressing need for many applications. For example, in disaster control, it is highly important to build a real-time disaster detector that constantly monitors a geographical region. By sending out timely alarms when emergent disasters outbreak, the detector can help people take timely actions to alleviate huge life and economic losses. Another example is public order maintaining. For local governments, it is desirable to monitor people's activities in the city and know about social unrests (*e.g.*, protest, crime) as soon as possible. With a detector that discovers social unrests upon their onsets, the government can respond timely to prevent severe social riots.

Figure 5.1 illustrated the workflow of spatiotemporal event detection in the cube space. As shown, from the cube structure, the user can select chunks of unstructured data by specifying queries along multiple dimensions, *e.g.*, ⟨*, USA, 2017⟩, ⟨Entertainment, Japan, 2018⟩. From the user-selected data, the spatiotemporal event detector aims at extracting abnormal multi-dimensional events. Note that the cube structure and the event detector are tightly coupled instead of being independent. By virtue of the cube structure, the users can specify query conditions, which allow the detector to determine what constitute "abnormal" patterns with respect to the user-specified contexts.

Detecting abnormal spatiotemporal events in the cube space is by no means a trivial task. It has two unique challenges that largely limit the performance of existing methods: 1) *Capturing anomaly in a multi-dimensional space.* Existing event detection methods rely on heuristic ranking functions to select the top-K bursty events [6, 3, 76, 3, 93, 42]. An abnormal spatiotemporal event, however, may not be bursty in the multi-dimensional space. For example, when a protest occurs at the 5th Avenue, there can be only a few people discussing about this event on Twitter. The key challenge is to distinguish abnormal events

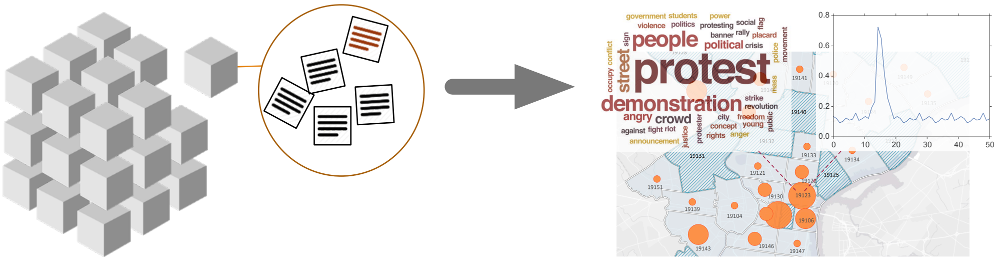


Figure 5.1: An illustration of spatiotemporal event detection in the cube space.

(*e.g.*, protest in the 5th Avenue) from routine activities (*e.g.*, shopping in the 5th Avenue) by jointly modeling multiple factors; 2) *Fast online detection.* When a spatiotemporal event outbreaks, our goal is to report the event instantly to allow for timely actions. Hence, it is desirable to continuously monitor the massive text stream and report spatiotemporal events on the fly. Such a requirement renders existing batch-wise detection methods [16, 46, 91] undesirable.

We propose TrioVecEvent, a method that combines multimodal embeddings and latent variable models for accurate online spatiotemporal event detection. The foundation of TrioVecEvent is the multimodal embedding learner that maps all the regions, periods, and keywords into the same space with their correlations preserved, which we have described in the previous chapter. If two items are highly correlated (*e.g.*, 'Pats' and 'Patriots', or the 5th Avenue region and the keyword 'shopping'), their representations in the latent space tend be close. Such multimodal embeddings not only allow us to capture the subtle semantic similarities between records, but also serve as background knowledge by revealing the typical keywords in different regions and periods.

Built upon the multimodal embeddings, TrioVecEvent employs a two-step scheme to achieve high detection accuracy. First, *it performs online clustering to divide the records in the query window into coherent geo-topic clusters.* We develop a novel Bayesian mixture model that jointly models the record locations in the Euclidean space and the semantic embeddings in the spherical space. The model can generate quality candidates to ensure a high coverage of the underlying events. Second, *it extracts a set of discriminative features for accurate candidate classification.* Based on the multimodal embeddings, we design features that can well characterize spatiotemporal events, which enable pinpointing true positives from the candidate pool with only a small amount of training data. Compared with existing top-K candidate selection schemes, the classification-based candidate filtering not only frees us from designing heuristic ranking functions, but also eliminates the inflexibility of rigid top-K selection. Furthermore, as the query window shifts continuously, TrioVecEvent does not need to detect the spatiotemporal events in the new window from scratch, but just needs to update the previous results with little cost to enable fast online detection.

Our main contributions are summarized as follows:

1. We propose a novel Bayesian mixture clustering model that finds geo-topic clusters as candidate events. It generates quality geo-topic clusters without specifying the number of clusters a priori, and continuously updates the clustering results as the query window shifts. The clustering model is novel in that it for the first time combines two powerful techniques: representation learning and graphical models. The former can well encode the semantics of unstructured text, while the latter is good at expressing the complex structural correlations among different factors.

2. We design an effective candidate classifier that judges whether each candidate is indeed a spatiotemporal event. Relying on the multimodal embeddings, we extract a set of discriminative features for the candidates, which enable identifying multi-dimensional anomaly with a small amount of training data.

3. We have performed extensive experiments on large-scale geo-tagged tweet streams. Our effectiveness studies based on crowdsourcing show that TrioVecEvent improves the detection precision of the state-of-the-art method by a large margin. Meanwhile, TrioVecEvent demonstrates excellent efficiency, making it suitable to be deployed for monitoring large-scale text streams in practice.

5.2 RELATED WORK

In this section, we review existing work related to event detection, including: (1) bursty event detection; (2) spatiotemporal event detection; and (3) event forecasting.

5.2.1 Bursty Event Detection

A larger number of methods have been proposed for extracting globa events that are bursty in the entire data stream. Generally, existing global event detection approaches can be classified into two categories: *document-based* and *feature-based*. Document-based approaches [6, 3, 76] consider each document as a basic unit. They group similar documents into clusters and then find the bursty ones as events. For instance, Allan *et al.* [6] perform online clustering and use a similarity threshold to determine whether a new document should form a new topic or be merged into an existing one; Aggarwal *et al.* [3] also detect events via online clustering, but with a similarity measure that considers both tweet content relevance and user proximity; Sankaranarayanan *et al.* [76] train a Naïve Bayes filter to obtain news-related tweets and cluster them based on TF-IDF similarity. Feature-based approaches [36, 60, 93, 42, 49] identify a set of bursty features (*e.g.*, keywords) and cluster them to form events. Various techniques for extracting bursty features have been proposed, such as Fourier transform [36], Wavelet transform [93], and phrase-based burst detection [49, 30]. For example, Fung *et al.* [28] model feature occurrences with binomial distribution to extract bursty features; He *et al.* [36] construct the time series for each feature and perform Fourier Transform to identify bursts; Weng *et al.* [93] use wavelet transform and auto-correlation to measure word energy and extract high-energy words; Li *et al.* [49] segment each tweet into meaningful phrases and extract bursty phrases based on frequency; Giridhar *et al.* [30] extract an event as a group of tweets that contain at least one pair of bursty keywords. The above methods are all designed for detecting globally bursty events. A spatiotemporal event, however, is usually bursty in a local region instead of the entire stream. Hence, directly applying these methods to our problem can miss many spatiotemporal events.

5.2.2 Spatiotemporal Event Detection

Spatiotemporal event detection has been receiving increasing research interest in the past few years [26, 71, 75, 16, 46, 25, 1]. Watanabe *et al.* [91] and Quezada *et al.* [71] extract location-aware events in the social media, but their focus is on geo-locating the tweets/events. Sakaki *et al.* [75] achieve real-time earthquake detection, by training a classifier to judge whether an incoming tweet is earthquake-related. Li *et al.* [52] detect crime and disaster events (CDE) with a self-adaptive crawler for CDE-related tweets. Our work differs from these studies in that we aim to detect all kinds of spatiotemporal events, whereas they focus on specific event types. Quite a few generic spatiotemporal event detection methods have been proposed [16, 46, 1]. Chen *et al.* [16] use Wavelet transform to extract spatiotemporally bursty Flickr tags, and then cluster them based on their co-occurrences and spatiotemporal distributions. Krumm *et al.* [46] discretize the time into equal-size bins and compare the number of tweets in the same bin across different days to extract spatiotemporal events. Nevertheless, the above methods can only handle static data and detect spatiotemporal events in batch. While online methods have been gaining increasing attention in the data mining community, few methods exist for supporting online spatiotemporal event detection. Abdelhaq *et al.* [1] first extract bursty and localized keywords in the query window, then cluster such keywords based on their spatial distributions, and finally select the top-K locally bursty clusters. While these two methods support online spatiotemporal event detection, their accuracies are limited because of two reasons: 1) the clustering step does not capture short-text semantics well; and 2) the candidate filtering effectiveness is limited by heuristic ranking functions and the inflexibility of top-K selection.

5.2.3 Spatiotemporal Event Forecasting

Spatiotemporal event forecasting is another line of research that is related to our problem. Foley *et al.* [26] use distant supervision to extract future spatiotemporal events from Web pages, but the proposed method can only extract spatiotemporal events that are well advertised in advance on the Web. Zhao *et al.* [109, 110, 108] formulate spatiotemporal event forecasting as a binary prediction problem, *i.e.*, predicting whether a specific type of event (*e.g.*, civil unrest) will occur on a given day. Their methods combine social media with other data sources (*e.g.*, gold standard report, news articles) to train reliable predictors. Our problem is orthogonal to their studies in that, instead of performing binary prediction for a specific event type, we attempt to extract all types of spatiotemporal events at their onsets.

5.3 PRELIMINARIES

5.3.1 Problem Definition

Given a three-dimensional text-location-time cube, let $\mathcal{D} = (d_1, d_2, \ldots, d_n, \ldots)$ be a collection of text records with spatiotemporal information (*e.g.*, geo-tagged tweets) that arrive in chronological order. Each record d is a tuple $\langle t_d, l_d, x_d \rangle$, where t_d is its post time, l_d is its geo-location, and x_d is a bag of keywords that denote the text message. Consider a query cube chunk Q, *e.g.*, ⟨ *, NYC, June⟩, ⟨ *, LA, July 1st 9pm⟩. The spatiotemporal event detection problem aims at extracting all the spatiotemporal events that occur in Q.

5.3.2 TrioVecEvent: Event Detection with Multimodal Embeddings

A spatiotemporal event often results in relevant records around its occurring location. For example, suppose a protest occurs at the JFK Airport in New York City, many participants post tweets on the spot to express their attitude, with keywords like ‘protest’ and ‘rights’. Such records form a geo-topic cluster as they are geographically close and semantically relevant. However, not necessarily does every geo-topic cluster correspond to a spatiotemporal event. It is because a geo-topic cluster may correspond to just routine activities in the region, *e.g.*, taking flights at JFK, shopping at the 5th Ave, *etc.*. We claim that a spatiotemporal event often leads to a *bursty and unusual geo-topic cluster*. The cluster is bursty in that it consists of a considerable number of messages, and unusual in that its semantics deviates from routine activities significantly.

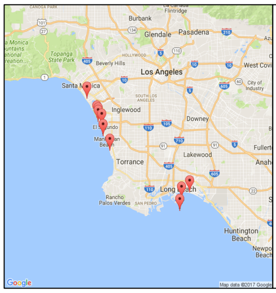
Motivated by the above, we design an embedding-based detection method TrioVecEvent. At the foundation of TrioVecEvent is a multimodal embedding learner that maps all the regions, hours, and keywords into a latent space. If two items are highly correlated (*e.g.*, 'flight' and 'airport', or the JFK Airport region and the keyword 'flight'), their embeddings in the latent space tend be close. Figure 5.2 shows two real examples in Los Angeles and New York City, where we learn multimodal embeddings using millions of tweet records in these cities and perform similarity searches. One can see that given the example queries, the multimodal embeddings well capture the correlations between different items. The usage of such embeddings is two-fold: 1) they allow us to capture the semantic similarities between text messages and further group the records into coherent geo-topic clusters; and 2) they reveal the typical keywords appearing in different regions and hours, which serve as background knowledge to help identify abnormal spatiotemporal activities.

Figure 5.3 shows the framework of TrioVecEvent. As shown, *the embedding learner* embeds the location, time, and text using massive data from the input data stream. It maintains a cache for keeping newly arrived records and updating the embeddings periodically. Based on the multimodal embeddings, TrioVecEvent employs a two-step detection scheme: 1) in *the online clustering step*, we develop a Bayesian mixture model that jointly models geographical locations and semantic embeddings to extract coherent geo-topic clusters in the query chunk; 2) in *the candidate classification* step, we extract a set of discriminative features for the candidates and determine whether each candidate is a true spatiotemporal event.

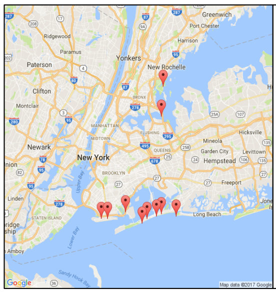
Now the key questions about TrioVecEvent are: 1) how to generate embeddings that can well capture the correlations between different items? 2) how do we perform online clustering to obtain quality geo-topic clusters in Q? and 3) what are the features that can discriminate true spatiotemporal events from non-events? In what follows, we introduce the multimodal embedding learner and then describe the two-step detection process of TrioVecEvent.

5.3.3 Multimodal Embedding

The multimodal embedding module jointly maps all the spatial, temporal, and textual items into the same low-dimensional space with their correlations preserved. The multimodal

	lax	lakers	dodgers	beachlife
	international	kobe	ladders	sand
	losangeles	bryant	dogerstadium	boardwalk
	united	bulls	itfdb	ocean
	people	cavs	letsgododgers	wave
	tsa	kevin	game	beachday
	sfo	knicks	dodgergame	pacificocean
	food	clipper	play	santamonica
	flight	lebron	losdoyers	pier
	travel	cp3	win	wave
“beach”	“33.942, -118.409”	“nba”	“baseball”	“beach”

(a) Examples on LA (the second query is the location of the LAX Airport).

	jfk	knicks	mlb	rockaway
	airport	melo	yankees	beachday
	international	lebron	mets	howard_beach
	johnfkennedy	durant	yanks	brighton
	burger	basketball	inning	longbeach
	terminal	kobe	yankee	coney
	john	cavs	ballpark	atlantic
	kennedy	theknicks	pitch	island
	sfo	game	jeter	boardwalk
	flight	lakers	game	long
“beach”	“40.641, -73.778”	“nba”	“baseball”	“beach”

(b) Examples on NY (the second query is the location of the JFK Airport).

Figure 5.2: Example similarity queries based on the multimodal embeddings learned from the geo-tagged tweets in Los Angeles and New York City. In each city, the first query retrieves regions relevant to the keyword ‘beach’; the second retrieves keywords relevant to the airport location; and the last three retrieve relevant keywords for the given query keywords. For each query, we use the learned embeddings to compute the cosine similarities between different items, and retrieve the top ten most similar items without including the query itself.

embedding learner consumes the input data stream and learns D-dimensional representations for all the regions, periods, and keywords. As aforementioned, we maintain a cache C for keeping newly arrived records, and use it to periodically update the embeddings. To effectively incorporate the information in C without overfitting, we take the embeddings learned before the arrival of C as initialization, and optimize the embeddings over C for one full epoch. Such a simple strategy efficiently incorporates the records in the cache C, while largely preserving the information in the historical stream.

The multimodal embedding learner is based on the reconstruction task we described in the previous chapter. Here let us briefly review the multimodal embedding learning process. The learning objective is to predict one item given its context. Specifically, given a record d, for any item $i \in d$ with type X (region, period, or keyword), let $\mathbf{v}_i$ be the embedding of

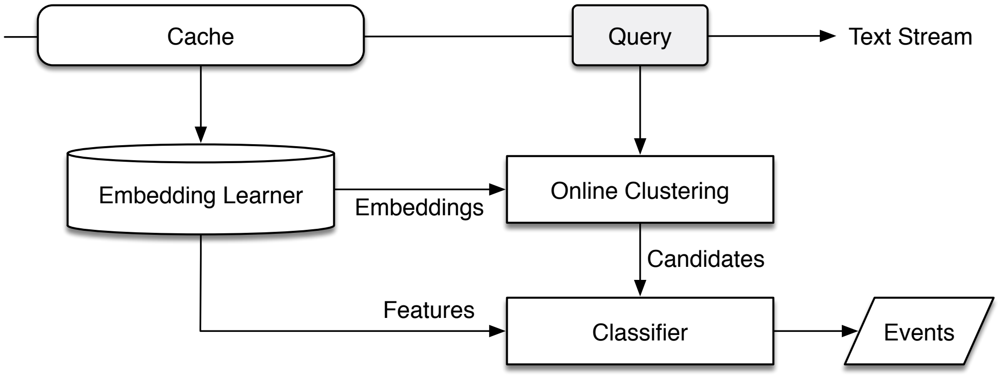


Figure 5.3: The framework of TrioVecEvent.

item i, then we model the likelihood of observing i as

$$p(i|d_{-i}) = \exp(s(i, d_{-i}))/\sum_{j \in X} \exp(s(j, d_{-i})),$$

where d_{-i} is the set of all the items in d except i; and $s(i, d_{-i})$ is the similarity score between i and d_{-i}, defined as

$$s(i, d_{-i}) = \mathbf{v}_i^{\mathrm{T}} \sum_{j \in d_{-i}} \mathbf{v}_j / |d_{-i}|.$$

For a cache C of records, the objective is to predict all the items of the records in C:

$$J_C = -\sum_{d \in C} \sum_{i \in d} \log p(i|d_{-i}).$$

To efficiently optimize the above objective function, we follow the idea of negative sampling and use stochastic gradient descent (SGD) for updating. At each time, we randomly sample a record d from C and a item $i \in d$. With negative sampling, we randomly select K negative items that have the same type with i but do not appear in d. Then we minimize the following function for the selected samples:

$$J_d = -\log \sigma(s(i, d_{-i})) - \sum_{k=1}^{K} \log \sigma(-s(k, d_{-i})),$$

where $\sigma(\cdot)$ is the sigmoid function. The updating rules for different variables can be easily derived by taking the derivatives of the above objective and then applying SGD, we omit the details here due to the space limit.

5.4 CANDIDATE GENERATION

We develop a Bayesian mixture clustering model to divide the records in the query chunk Q into a number of geo-topic clusters, such that the records in the same cluster are geographically close and semantically relevant. Such geo-topic clusters will serve as candidate abnormal events, which will later be filtered to pinpoint the true events.

We consider each record d as a tuple $(\mathbf{l}_d, \mathbf{x}_d)$. Here, $\mathbf{l}_d$ is a 2-dimensional vector denoting d's geo-location; and $\mathbf{x}_d$ is the D-dimensional semantic embedding of d, derived by averaging the embeddings of the keywords in d's message. Table 5.1 summarizes the notations we used in this section.

Table 5.1: The notations used in the Bayesian mixture clustering model.

$\mathcal{X}$	the set of semantic embeddings for the records in Q
$\mathcal{Z}$	the set of cluster memberships for the records in Q
$\mathcal{L}$	the set of geo-location vectors for the records in Q
$\boldsymbol{\kappa}$	the set of κ for all the clusters
$\boldsymbol{\kappa}^{\neg k}$	the subset of $\boldsymbol{\kappa}$ excluding the one for cluster k
$\mathbf{A}^{\neg d}$	the subset of any set $\mathbf{A}$ excluding element d
$\mathbf{A}^{k}$	the subset of elements that are assigned to cluster k in set $\mathbf{A}$
$\mathbf{x}^{k}$	the sum of the semantic embeddings in cluster k
$\mathbf{x}^{k,\neg d}$	the sum of the semantic embeddings in cluster k excluding d
n^{k}	the number of records in cluster k
$n^{k,\neg d}$	the number of records in cluster k excluding d

5.4.1 A Bayesian Mixture Clustering Model

The key idea behind our Bayesian mixture clustering model is that every geo-topic cluster implies a coherent activity (*e.g.*, protest) around a certain geo-location (*e.g.*, the JFK Airport). The location acts as a *geographical center* that triggers geo-location observations around it in the Euclidean space; while the activity serves as a *semantic focus* that triggers semantic embedding observations around it in the spherical space. We assume there are at most K geo-topic clusters in the query cell Q. Note that assuming the maximum number of clusters is a weak assumption that can be readily met in practice. At the end of the clustering process, some of these K cluster may become empty. As such, the appropriate number of clusters in any ad-hoc query cell can be automatically discovered.

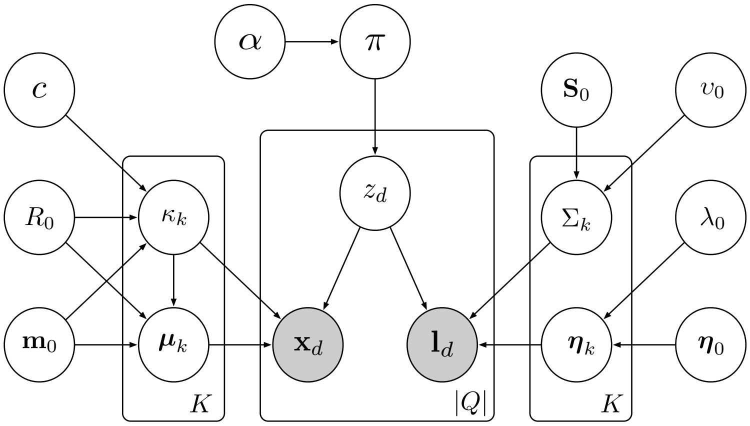


Figure 5.4: The the Bayesian mixture clustering model of generating geo-topic clusters.

Figure 5.4 shows the generative process for all the records in the query cell Q. As shown, we first draw a multinomial distribution π from a Dirichlet prior $\mathbf{Dirichlet}(.|\alpha)$. Meanwhile, for modeling the geo-locations, we draw K normal distributions from a Normal-Inverse-Wishart (NIW) prior $\mathbf{NIW}(.|\boldsymbol{\eta}_0, \lambda_0, \mathbf{S}_0, \upsilon_0)$ [65], which is a conjugate prior of the normal distribution; and for modeling the semantic embeddings, we draw K von Mises-Fisher (vMF) distributions from its conjugate prior $\Phi(\boldsymbol{\mu}, \kappa|\mathbf{m}_0, R_0, c)$ [67]. For each record $d \in Q$, we first draw its cluster membership z_d from π. Once the cluster membership is determined, we draw its geo-location $\mathbf{l}_d$ from the respective normal distribution, and its semantic embedding $\mathbf{x}_d$ from the respective vMF distribution.

While using normal distributions for modeling the geo-location $\mathbf{l}_d$ is intuitive, we justify the choice of the vMF distribution for modeling the semantic embedding $\mathbf{x}_d$ as follows. For a D-dimensional unit vector $\mathbf{x}$ that follows vMF distribution, its probability density function is given by

$$p(\mathbf{x}|\boldsymbol{\mu}, \kappa) = C_D(\kappa) \exp(\kappa \boldsymbol{\mu}^T \mathbf{x}),$$

where $C_D(\kappa) = \frac{\kappa^{D/2-1}}{I_{D/2-1}(\kappa)}$ and $I_{D/2-1}(\kappa)$ is the modified Bessel function. The vMF distribution has two parameters: the mean direction $\boldsymbol{\mu}$ ($\boldsymbol{\mu} = \mathbf{1}$) and the concentration parameter κ ($\kappa > 0$). The distribution of $\mathbf{x}$ on the unit sphere concentrates around the mean direction $\boldsymbol{\mu}$, and is more concentrated if κ is large. Our choice of the vMF distribution is motivated by the effectiveness of the cosine similarity [63] in quantifying the similarities between multimodal embeddings. The mean direction $\boldsymbol{\mu}$ acts as a semantic focus on the unit sphere, and produces relevant semantic embeddings around it, where concentration degree is controlled by the parameter κ. The superiority of the vMF distribution over other alternatives (*e.g.*, Gaussian) for modeling textual embeddings has also been demonstrated in recent studies on clustering [31] and topic modeling [9].

To summarize the above generative process, we have:

$$
\begin{aligned}
&\pi \sim \mathbf{Dirichlet}(.|\alpha) \\
&\{\boldsymbol{\eta}_k, \Sigma_k\} \sim \mathbf{NIW}(.|\boldsymbol{\eta}_0, \lambda_0, \mathbf{S}_0, \upsilon_0) \quad k = 1, 2, \ldots, K \\
&\{\boldsymbol{\mu}_k, \kappa_k\} \sim \Phi(.|\mathbf{m}_0, R_0, c) \quad k = 1, 2, \ldots, K \\
&z_d \sim \mathbf{Categorical}(.|\pi) \quad d \in \mathcal{Q} \\
&\mathbf{l}_d \sim \mathcal{N}(.|\boldsymbol{\eta}_{z_d}, \Sigma_{z_d}) \quad d \in \mathcal{Q} \\
&\mathbf{x}_d \sim \mathbf{vMF}(.|\boldsymbol{\mu}_{z_d}, \kappa_{z_d}) \quad d \in \mathcal{Q}
\end{aligned}
$$

where $\Lambda = \{\alpha, \mathbf{m}_0, R_0, c, \boldsymbol{\eta}_0, \lambda_0, \mathbf{S}_0, \upsilon_0\}$ are the hyper-parameters for the prior distributions.

5.4.2 Parameter Estimation

The key to obtain the geo-topic clusters is to estimate the posterior distributions for $\{z_d\}_{d \in \mathcal{Q}}$. We use Gibbs sampling for this purpose. Since we have chosen conjugate priors for π and $\{\boldsymbol{\mu}_k, \boldsymbol{\eta}_k, \Sigma_k\}_{k=1}^{K}$, these parameters can be integrated out during the Gibbs sampling process, resulting in a collapsed Gibbs sampling procedure. Due to the space limit, we directly give the conditional probabilities for $\{\kappa_k\}_{k=1}^{K}$ and $\{z_d\}_{d \in \mathcal{Q}}$:

$$
p(\kappa_k|\boldsymbol{\kappa}^{\neg k}, \mathcal{X}, \mathcal{Z}, \alpha, \mathbf{m}_0, \mathcal{R}_0, c) \propto \frac{(C_D(\kappa_k))^{c+n^k}}{C_D(\kappa_k \|R_0\mathbf{m}_0 + \mathbf{x}^k\|)}, \tag{5.1}
$$

$$
\begin{aligned}
p(z_d = k|\mathcal{X}, \mathcal{L}, \mathcal{Z}^{\neg d}, \boldsymbol{\kappa}, \Lambda) &\propto p(z_d = k|\mathcal{Z}^{\neg d}, \alpha) \cdot \\
p(\mathbf{x}_d|\mathcal{X}^{\neg d}, \mathcal{Z}^{\neg d}, z_d = k, \Lambda) &\cdot p(\mathbf{l}_d|\mathcal{L}^{\neg d}, \mathcal{Z}^{\neg d}, z_d = k, \Lambda).
\end{aligned} \tag{5.2}
$$

The three quantities in Equation 5.2 are given by:

$$
p(z_d = k|\cdot) \propto (n^{k,\neg d} + \alpha), \tag{5.3}
$$

$$
p(\mathbf{x}_d|\cdot) \propto \frac{C_D(\kappa_k) C_D(\|\kappa_k(R_0\mathbf{m}_0 + \mathbf{x}^{k,\neg d})\|_2)}{C_D(\|\kappa_k(R_0\mathbf{m}_0 + \mathbf{x}^{k,\neg d} + \mathbf{x}_d)\|_2)}, \tag{5.4}
$$

$$
p(\mathbf{l}_d|\cdot) \propto \frac{\lambda^{k,\neg d}(\upsilon^{k,\neg d} - 1)|\mathbf{S}^{\mathcal{L}^k \cap \mathcal{L}^{\neg d}}|^{\upsilon^{k,\neg d}/2}}{2(\lambda^{k,\neg d} + 1)|\mathbf{S}^{\mathcal{L}^k \cup \{\mathbf{l}_d\}}|^{(\upsilon^{k,\neg d}+1)/2}}, \tag{5.5}
$$

where $\lambda^{\cdot}$, $\upsilon^{\cdot}$, and $\mathbf{S}^{\cdot}$ are posterior estimations for the NIW distribution parameters [65].

From Equation 5.2, 5.3, 5.4, and 5.5, we observe that our Bayesian mixture model enjoys several nice properties when determining the cluster membership for a record d: 1) With Equation 5.3, d tends to join a cluster that has more members, resulting in a rich-get-richer effect; 2) With Equation 5.4, d tends to join a cluster that is more semantically similar to its textual embedding $\mathbf{x}_d$, leading to semantically coherent clusters; and 3) With Equation 5.5, d tends to join a cluster that is more geographically close to its geo-location $\mathbf{l}_d$, resulting in geographically compact clusters.

5.5 CANDIDATE CLASSIFICATION

We have so far obtained a set of coherent geo-topic clusters in the query window as candidates. Now we proceed to describe the candidate classifier for pinpointing the true spatiotemporal events.

5.5.1 Features Induced from Multimodal Embeddings

The key observation for the candidate filtering component is that the multimodal embeddigns we learned allow for extracting a small feature set that are discriminative in determining whether a candidate event is true abnormaly or not. In the following, we introduce a set of features that can well discriminate true spatiotemporal events from non-events.

1. **Spatial unusualness** quantifies how unusual a candidate is in its geographical region. As the multimodal embeddings can unveil the typical keywords in different regions, we use them as background knowledge to measure the spatial unusualness of a candidate C. Specifically, we compute the spatial unusualness as $f_{su}(C) = \sum_{d \in C} \cos(\mathbf{v}_{l_d}, \mathbf{x}_d)/|C|$, where $\mathbf{v}_{l_d}$ is the embedding of the region of record d, and $\mathbf{x}_d$ is the semantic embedding of record d.

2. **Temporal unusualness** quantifies how temporally unusual a candidate is. We define the temporal unusualness of a candidate C as $f_{tu}(C) = \sum_{d \in C} \cos(\mathbf{v}_{t_d}, \mathbf{x}_d)/|C|$, where $\mathbf{v}_{t_d}$ is the embedding of the hour of record d.

3. **Spatiotemporal unusualness** jointly considers the space and time to quantify how unusual a candidate C is: $f_{stu}(C) = \sum_{d \in C} \cos((\mathbf{v}_{l_d} + \mathbf{v}_{t_d})/2, \mathbf{x}_d)/|C|$.

4. **Semantic concentration** computes how semantically coherent C is. The semantic concentration for a candidate is computed as $f_{su}(C) = \sum_{d \in C} \cos(\overline{\mathbf{x}}_d, \mathbf{x}_d)/|C|$, where $\overline{\mathbf{x}}_d$ is the average semantic embedding of the records in C.

5. **Spatial and temporal concentrations** quantify how concentrated a candidate C is over the space and time. We compute three quantities for the records in C: 1) the

standard deviation of the longitudes; 2) the standard deviation of the latitudes; and 3) the standard deviation of the creating timestamps.

6. **Burstiness** quantifies how bursty a candidate C is. We define it as the number of records in C divided by the time span of C.

5.5.2 The Classification Procedure

To summarize, for each candidate C, we extract the following features: 1) the spatial unusualness; 2) the temporal unusualness; 3) the spatiotemporal unusualness; 4) the semantic concentration; 5) the longitude concentration; 6) the latitude concentration; 7) the temporal concentration; and 8) the burstiness. With the above features, we use logistic regression to train a binary classifier and judge whether each candidate is indeed a spatiotemporal event. We choose the logistic regression classifier because of its robustness when there is limited training data. We have also tried other classifiers like Random Forest, and find that the logistic regression classifier has slightly better performance in our experiments. The training instances are collected over 100 query windows in a crowdsourcing platform. We will shortly describe the labeling process in Section 5.8.

5.6 SUPPORTING CONTINUOUS EVENT DETECTION

When the query window Q shifts, it is undesirable to re-compute the geo-topic clusters in the new query window from scratch for the purpose of fast online detection. We employ an incremental updating strategy that efficiently approximates the clustering results in the new window. As shown in Figure 5.5, assume the query window shifts from Q to Q', we denote by $D_- = \{d_1, \ldots, d_m\}$ the outdated tweets, and $D_+ = \{d_{n-k+1}, \ldots, d_n\}$ the new tweets. Instead of performing Gibbs sampling for all the tweets in Q', we simply drop D_- and sample the cluster memberships for the tweets in D_+. Such an incremental updating strategy achieves excellent efficiency and yields quality geo-topic clusters in practice as the memberships of the remaining tweets are mostly stable.

5.7 COMPLEXITY ANALYSIS

We analyze the time complexities of the candidate generation step and the candidate classification step separately. For candidate generation, to extract geo-topic clusters in the new query window, the time complexity is $O(INKD)$, where I is the number of Gibbs

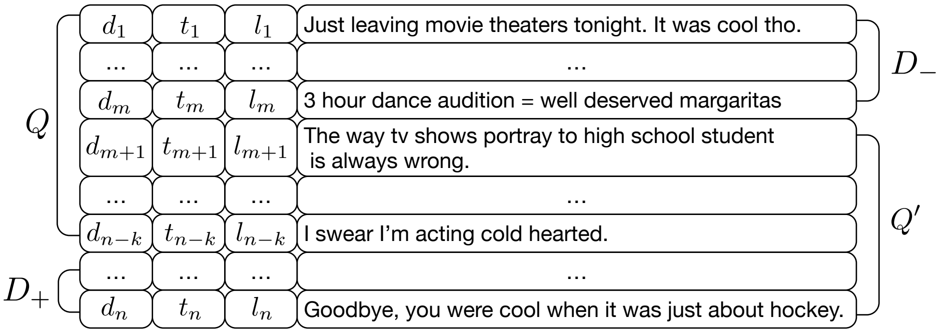

Q
D_-
Q'
D_+
d_1 t_1 l_1 Just leaving movie theaters tonight. It was cool tho.
... ...
d_m t_m l_m 3 hour dance audition = well deserved margaritas
d_{m+1} t_{m+1} l_{m+1} The way tv shows portray to high school student is always wrong.
... ...
d_{n-k} t_{n-k} l_{n-k} I swear I'm acting cold hearted.
... ...
d_n t_n l_n Goodbye, you were cool when it was just about hockey.

Figure 5.5: Incremental updating as the query window shifts.

sampling iterations, N is the number of new tweets, K is the maximum number of clusters; and D is the latent embedding dimension. Note that I, K and D are usually fixed to moderate values in practice, thus the candidate generation step scales roughly linearly with N and has good efficiency. For candidate classification, the major overhead lies in feature extraction. Let N_c be the maximum number of tweets in each candidate, then the time complexity of feature extraction is $O(KN_cD)$.

5.8 EXPERIMENTS

5.8.1 Experimental Settings

Baselines

We compare TrioVecEvent with all the existing online spatiotemporal event detection methods that we are aware of, described as follows:

- EvenTweet [1] extracts bursty and localized keywords from the query window, then clusters these keywords based on spatial distributions, and finally selects top-K locally bursty clusters.
- GeoBurst [105] is a strong random-walk-based method for online local event detection. It first uses random walk on a keyword co-occurrence graph to detect geo-topic clusters, and then ranks all the clusters by the weighted combination of spatial burstiness and temporal burstiness.
- GeoBurst+ [100] is an upgraded version of GeoBurst by replacing the ranking module with a classifier. Instead of heuristically ranking the candidates, we train a classifier to determine whether each candidate is a spatiotemporal event. The used features include spatial burstiness, temporal burstiness, as well as spatial and temporal concentrations

(Section 5.5).

Parameters

As EvenTweet and GeoBurst both perform top-K selection to identify spatiotemporal events from the candidate pool, we set $K = 5$ for them to achieve a tradeoff between precision and recall. Meanwhile, EvenTweet requires to partition the whole space into $M \times M$ small grids. After tuning, we set $M = 50$. In GeoBurst and GeoBurst+, there are three additional parameters: the kernel bandwidth h; (2) the restart probability α; and (3) the RWR similarity threshold δ. We set them as $h = 0.01, \alpha = 0.2$, and $\delta = 0.02$. All the baseline methods require a reference window that precedes the query to quantify the burstiness of the candidates, we follow [105] and set the reference duration to one week.

TrioVecEvent involves the following major parameters: (1) the latent dimension D for embedding; and (2) the maximum number of clusters K; and (3) the number of Gibbs sampling iterations I. After tuning, we set $D = 100$, $K = 500$, and $I = 10$, as we find such a setting can produce geo-topic clusters that are fine-grained enough while achieving good efficiency. In addition, the Bayesian mixture model involves several hyper-parameters, as shown in Figure 5.4. In general, we observe that our model is not very sensitive to them. We set $\alpha = 1.0, c = 0.01, R_0 = 0.01, \mathbf{m}_0 = 0.1 \cdot \mathbf{1}, \lambda_0 = 1.0, \boldsymbol{\eta}_0 = \mathbf{0}, \upsilon_0 = 2.0, \mathbf{S}_0 = 0.01 \cdot \mathbf{I}$, which are commonly adopted values for the prior distributions used in our model. We conduct the experiments on a computer with Intel Core i7 2.4GHz CPU and 8GB memory.

Data Sets and Groundtruth

Our experiments are based on real-life data from Twitter. The first data set LA consists of the geo-tagged tweets in Los Angeles collected during 2014.08.01 — 2014.11.30; and the second data set NY consists of the geo-tagged tweets in New York City during the same period. For each data set, we use an off-the-shelf tool [72] to preprocess the text messages by preserving entities and nouns, and then remove the keywords that appear less than 100 times in the entire corpus.

To evaluate the methods and collect training data for GeoBurst+ and TrioVecEvent, we randomly generate 200 non-overlapping query windows with four different lengths: 3-hour, 4-hour, 5-hour, and 6-hour. After ranking these windows in chronological order, we run each the method online by shifting a fixed-length (3h, 4h, 5h, 6h) query window on a 5-minute basis, and save the results falling in each target query window. After collecting labeled data with crowdsourcing, we use the groundtruth in the first 100 windows for training the classifiers of GeoBurst+ and TrioVecEvent; and that in the rest 100 windows for comparing all the methods.

Now we describe the labeling process based on crowdsourcing. For all the methods, we upload their results to CrowdFlower[1] for human judging. Since EvenTweet and GeoBurst are top-K methods with $K = 5$, we upload five results for each of them in each query window. GeoBurst+ and TrioVecEvent are classification-based methods, and the raw numbers of candidate events could be large. To limit the number of candidates while ensuring the coverages of the two methods, we employ a simple heuristic for eliminating negative candidates. It removes the candidates that have too few users (*i.e.*, the number of users is less than five) or too dispersed spatial distributions (*i.e.*, the longitude or latitude standard deviation is larger than 0.02). After filtering such trivial negatives, we upload the remaining candidates for evaluation.

On CrowdFlower, we represent each event with five tweets and ten keywords, and ask three CrowdFlower workers to judge whether the event is indeed a local event. To ensure the quality of the workers, we label 20 queries as groundtruth judgments on each data set, such that only the workers who can achieve no less than 80% accuracy on the groundtruth can submit their answers. Finally, we use majority voting to aggregate the workers' answers. The representative tweets and keywords are selected as follows: (1) For GeoBurst and GeoBurst+, we select five tweets having the largest authority scores, and ten keywords having the largest TF-IDF weights. (2) EvenTweet represents each event as a group of keywords. We select ten keywords with the highest scores in each event. Then we regard the group of keywords as a query to retrieve the top five most similar tweets using the BM25 retrieval model. (3) TrioVecEvent represents a candidate as a group of tweets. We first compute the average semantic embedding, and then select the closest keywords and tweets using cosine similarity.

Metrics

As aforementioned, we use the groundtruth in the last 100 query windows to evaluate all the methods. To quantify the performance of all the methods, we report the following metrics: (1) *Precision.* The detection precision is $P = N_{\text{true}}/N_{\text{report}}$, where N_{true} is the number of true spatiotemporal events and N_{report} is the total number of reported events. (2) *Pseudo Recall.* The true recall is hard to measure due to the lack of the comprehensive set of events in the physical world. We thus measure the pseudo recall for each method. Specifically, for each query window, we aggregate the true positives of different methods.

[1] http://www.crowdflower.com/

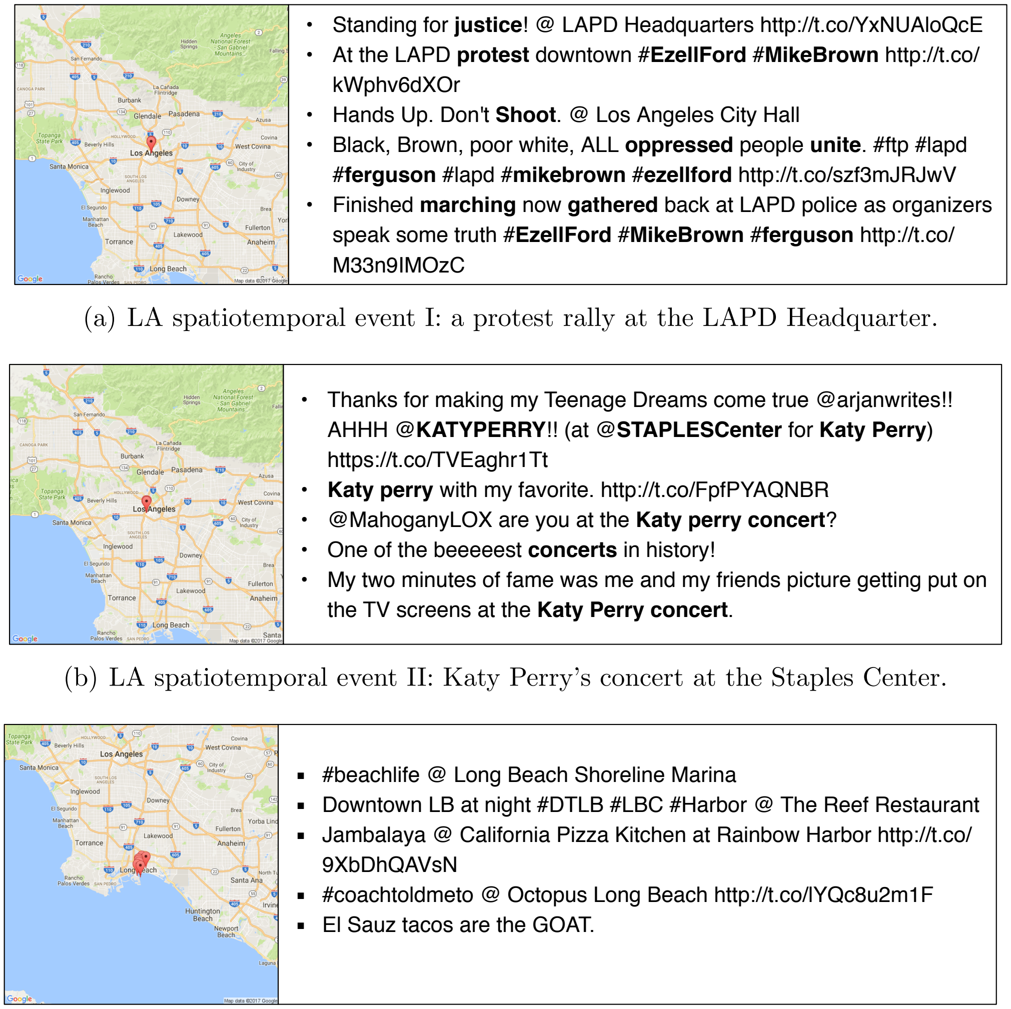
Let N_{total} be the total number of distinct spatiotemporal events detected by all the methods; we compute the pseudo recall of each method as $R = N_{\text{true}}/N_{\text{total}}$. (3) *Pseudo F1-Score.* Finally, we also report the pseudo F1 score of each method, which is computed as $F1 = 2 * P * R/(P + R)$.

5.8.2 Qualitative Results

Before reporting the quantitative results, we first present several examples for TrioVecEvent. Figure 5.6 and 5.7 show several exemplifying geo-topic clusters detected by TrioVecEvent on LA and NY, respectively. For each cluster, we plot the locations of the member tweets and show the top five representative tweets. The clusters in Figure 5.6(a) and 5.6(b) correspond to two positive spatiotemporal events in LA: 1) a protesting rally held at the LAPD Headquarter for making voice for Mike Brown and Ezell Ford; and 2) Katy Perry's concert at the Staples Center. For each event, one can see the generated geo-topic cluster is of high quality — the tweets in each cluster are highly geographically compact and semantically coherent. Even if there are tweets discussing about the event with different keywords (*e.g.*, 'shoot', 'justice', and 'protest'), TrioVecEvent can group them into the same cluster. This is because the multimodal embeddings can effectively capture the subtle semantic correlations between the keywords. While the first two clusters are classified as true spatiotemporal events by TrioVecEvent, the last one in Figure 5.6(c) is marked as negative. Although the last one is also a meaningful geo-topic cluster, it reflects routine activities around the long beach instead of any unusual events. TrioVecEvent is able to capture this fact and classify it into the negative class.

Figure 5.7(a) and 5.7(b) show two example spatiotemporal events detected by TrioVecEvent on NY. The first is the Hoboken Arts and Music Festival; and the second is the basketball game between the Knicks and the Hawks. Again, we can see the member tweets are highly relevant both geographically and semantically. As they represent interesting and unusual activities in their respective areas, TrioVecEvent successfully classifies them as true spatiotemporal events. In contrast, the third cluster just reflects the everyday activity of having food around the Time Square, and is returned as a non-event.

To further understand why TrioVecEvent is capable of generating high-quality geo-topic clusters and eliminating non-event candidates, we can re-examine the cases in Figure 5.2. As shown, the retrieved results based on the learned embeddings are highly meaningful. For instance, given the query 'beach', the top locations are all beach-life areas in LA and NYC; given the location of the airport, the top keywords reflect typical flight-related activities around the airport; and given different keywords as queries, the retrieved keywords



- Standing for **justice**! @ LAPD Headquarters http://t.co/YxNUAloQcE
- At the LAPD **protest** downtown **#EzellFord #MikeBrown** http://t.co/kWphv6dXOr
- Hands Up. Don't **Shoot**. @ Los Angeles City Hall
- Black, Brown, poor white, ALL **oppressed** people **unite**. #ftp #lapd **#ferguson** #lapd **#mikebrown #ezellford** http://t.co/szf3mJRJwV
- Finished **marching** now **gathered** back at LAPD police as organizers speak some truth **#EzellFord #MikeBrown #ferguson** http://t.co/M33n9IMOzC

(a) LA spatiotemporal event I: a protest rally at the LAPD Headquarter.

- Thanks for making my Teenage Dreams come true @arjanwrites!! AHHH @**KATYPERRY**!! (at @**STAPLESCenter** for **Katy Perry**) https://t.co/TVEaghr1Tt
- **Katy perry** with my favorite. http://t.co/FpfPYAQNBR
- @MahoganyLOX are you at the **Katy perry concert**?
- One of the beeeeest **concerts** in history!
- My two minutes of fame was me and my friends picture getting put on the TV screens at the **Katy Perry concert**.

(b) LA spatiotemporal event II: Katy Perry's concert at the Staples Center.

- #beachlife @ Long Beach Shoreline Marina
- Downtown LB at night #DTLB #LBC #Harbor @ The Reef Restaurant
- Jambalaya @ California Pizza Kitchen at Rainbow Harbor http://t.co/9XbDhQAVsN
- #coachtoldmeto @ Octopus Long Beach http://t.co/lYQc8u2m1F
- El Sauz tacos are the GOAT.

(c) LA non-event: enjoying beach life at the Long Beach.

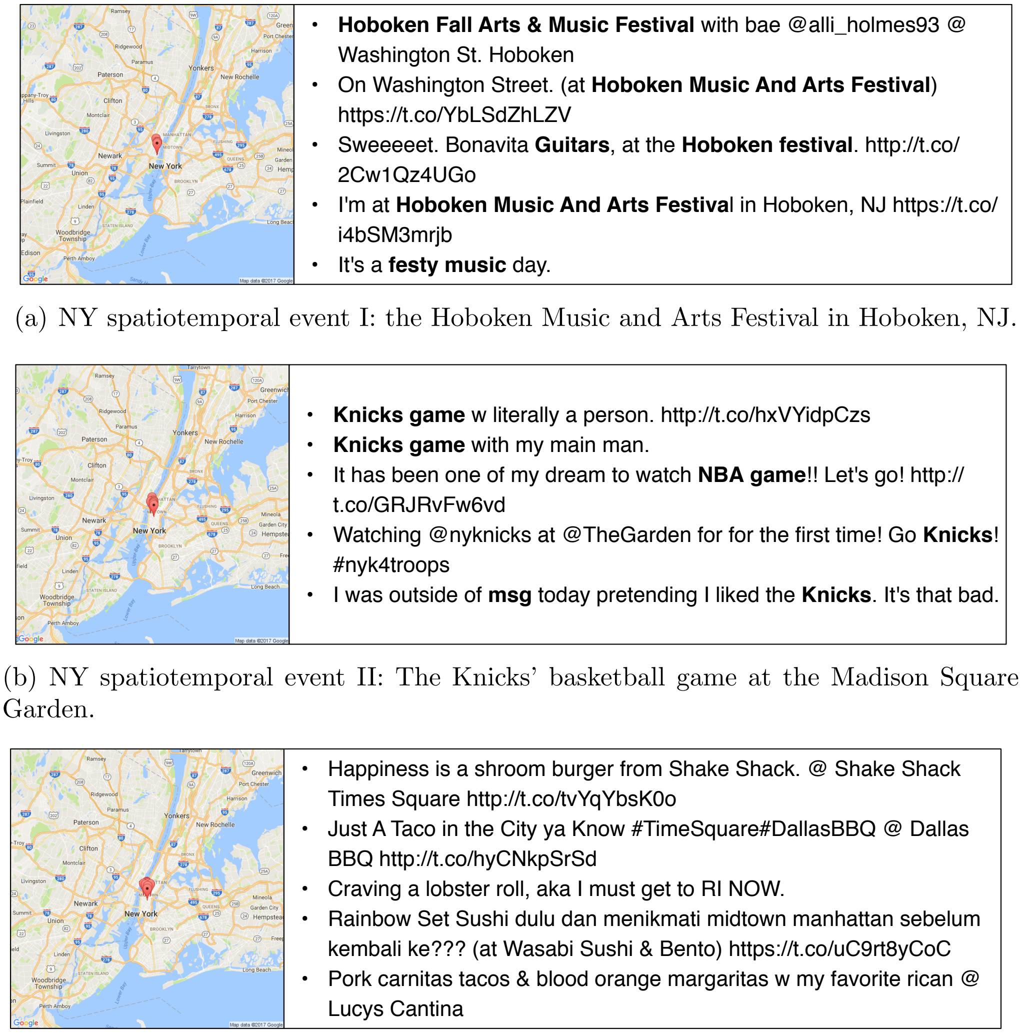
Figure 5.6: Example geo-topic clusters on LA. The first two are classified as positive spatiotemporal events and the third as negative.

are semantically relevant. Such results explain why TrioVecEvent is capable of grouping relevant tweets into the same geo-topic cluster and why the embeddings can serve as useful knowledge for extracting discriminative features (*e.g.*, spatial and temporal unusualness).

5.8.3 Quantitative Results

Effectiveness Comparison

Table 5.2 reports the precision, pseudo recall, and pseudo F1 of all the methods on LA and NY. We find that TrioVecEvent significantly outperforms the baseline methods on



- **Hoboken Fall Arts & Music Festival** with bae @alli_holmes93 @ Washington St. Hoboken
- On Washington Street. (at **Hoboken Music And Arts Festival**) https://t.co/YbLSdZhLZV
- Sweeeeet. Bonavita **Guitars**, at the **Hoboken festival**. http://t.co/2Cw1Qz4UGo
- I'm at **Hoboken Music And Arts Festiva**l in Hoboken, NJ https://t.co/i4bSM3mrjb
- It's a **festy music** day.

(a) NY spatiotemporal event I: the Hoboken Music and Arts Festival in Hoboken, NJ.

- **Knicks game** w literally a person. http://t.co/hxVYidpCzs
- **Knicks game** with my main man.
- It has been one of my dream to watch **NBA game**!! Let's go! http://t.co/GRJRvFw6vd
- Watching @nyknicks at @TheGarden for for the first time! Go **Knicks**! #nyk4troops
- I was outside of **msg** today pretending I liked the **Knicks**. It's that bad.

(b) NY spatiotemporal event II: The Knicks' basketball game at the Madison Square Garden.

- Happiness is a shroom burger from Shake Shack. @ Shake Shack Times Square http://t.co/tvYqYbsK0o
- Just A Taco in the City ya Know #TimeSquare#DallasBBQ @ Dallas BBQ http://t.co/hyCNkpSrSd
- Craving a lobster roll, aka I must get to RI NOW.
- Rainbow Set Sushi dulu dan menikmati midtown manhattan sebelum kembali ke??? (at Wasabi Sushi & Bento) https://t.co/uC9rt8yCoC
- Pork carnitas tacos & blood orange margaritas w my favorite rican @ Lucys Cantina

(c) NY non-event: having food aroun the Time Square.

Figure 5.7: Example geo-topic clusters detected on NY. The first two are classified as positive spatiotemporal events; while the third as negative.

both data sets. Compared with the strongest baseline GeoBurst+, TrioVecEvent yields around 118% improvement in precision, 26% improvement in pseudo recall, and 66% improvement in pseudo F1-score. The huge improvements are attributed to the two advantages of TrioVecEvent: (1) the embedding-based clustering model capture short-text semantics more effectively, and generate high-quality geo-topic clusters to achieve a good coverage of all the potential events; and (2) the multimodal embeddings enable the classifier to extract discriminative features for the candidates, and thus accurately pinpoint true spatiotemporal events.

Table 5.2: The performance of different methods. 'P' is precision, 'R' is pseudo recall; and 'F1' is pseudo F1 score.

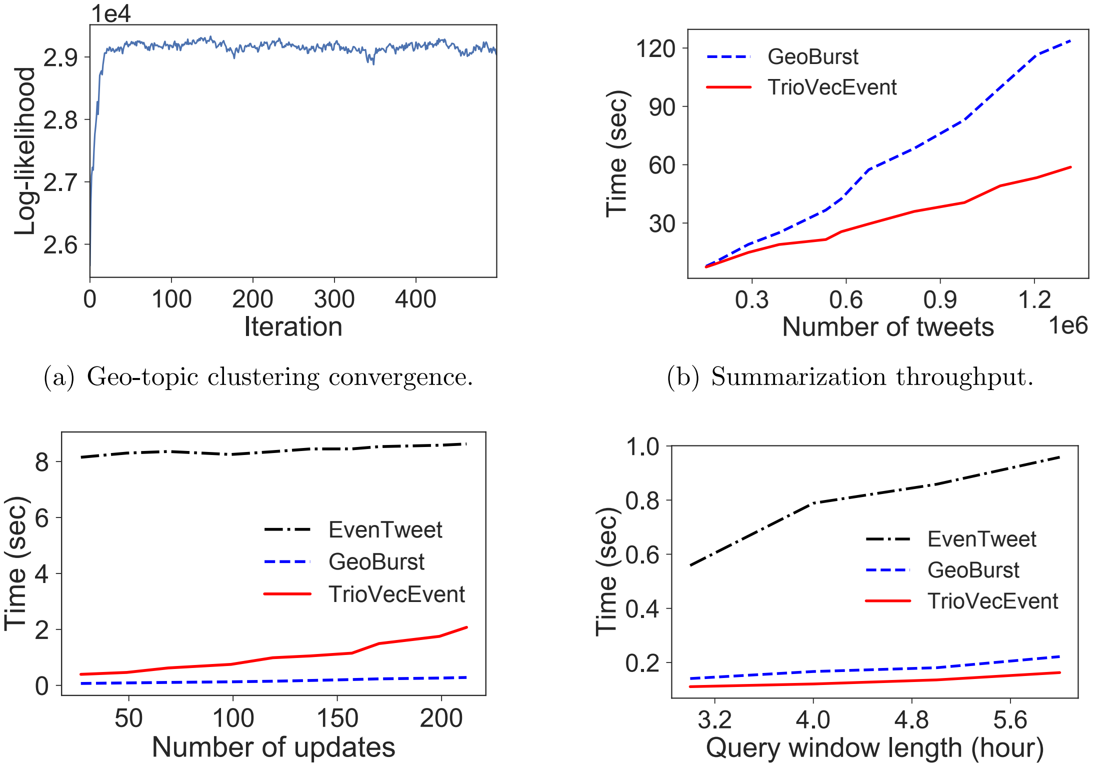
Method	LA			NY		
	P	R	F1	P	R	F1
EvenTweet	0.132	0.212	0.163	0.108	0.196	0.139
GeoBurst	0.282	0.451	0.347	0.212	0.384	0.273
GeoBurst+	0.368	0.483	0.418	0.351	0.465	0.401
TrioVecEvent	**0.804**	**0.612**	**0.695**	**0.765**	**0.602**	**0.674**

Comparing GeoBurst and its upgraded version GeoBurst+, we find that GeoBurst+ outperforms GeoBurst by a considerable margin. Such a phenomenon further verifies that classification-based candidate filtering is superior to the ranking-based strategy, even with moderately-sized training data. EvenTweet performs much poorer than the other methods on our data. After investigating the results, we find that although EvenTweet can extract spatiotemporally bursty keywords in the query window, clustering these keywords merely based on the spatial distributions often leads to semantically irrelevant keywords in the same cluster, which yields suboptimal detection accuracies.

5.8.4 Scalability Study

We proceed to report the efficiency of different methods. Since the time cost of GeoBurst+ is almost the same as GeoBurst, we only show the cost of GeoBurst for brevity. First, we study the convergence rate of the Gibbs sampler for the Bayesian mixture model. For this purpose, we randomly select a three-hour query window, and apply the Bayesian mixture model for extracting geo-topic clusters in the query window. Figure 5.8(a) shows the log-likelihood as the number of Gibbs sampling iterations increases. We observe that the log-likelihood quickly converges after a few iterations. Hence, it is usually sufficient to set the number of iterations to a relatively small value (*e.g.*, 10) in practice for better efficiency.

Both GeoBurst and TrioVecEvent require summarizing the continuous tweet stream for obtaining background knowledge: the summarization of GeoBurst is done by extending the Clustream algorithm [2]; while that of TrioVecEvent is achieved with multimodal embedding. In this set of experiments, we compare the throughputs of the summarization modules in these two methods. Specifically, we apply the two methods to process LA and record the accumulated CPU time for summarization in the process. As depicted in Figure 5.8(b), the summarization of both methods scales well with the number of tweets, and



(a) Geo-topic clustering convergence.

(b) Summarization throughput.

(c) Online clustering time.

(d) Candidate filtering time.

Figure 5.8: Efficiency study on LA. Figure 5.8(a) shows the convergence rate of the Bayesian mixture model; Figure 5.8(b) shows the summarization throughputs for GeoBurst and TrioVecEvent; Figure 5.8(c) shows the cost of online clustering; and Figure 5.8(d) shows the cost of candidate filtering.

TrioVecEvent is about 50% faster than GeoBurst. Meanwhile, we observe that the embedding learner scales roughly linearly with the number of processed tweets, making it suitable for large-scale tweet streams.

Now we investigate the efficiency of online clustering and candidate filtering for different methods. To this end, we randomly generate 1000 3-hour query window, and continuously shift each query window on a basis of 1, 2, . . ., 10 minutes. In Figure 5.8(c), we report the averaged running time of different methods in terms of the number of new tweets. As shown, both GeoBurst and TrioVecEvent are much more efficient than EvenTweet, while GeoBurst is the fastest. In terms of candidate filtering, Figure 5.8(d) reports the running time of the three methods as the query window length changes. Among the three methods, TrioVecEvent achieves the best efficiency for candidate filtering. This is because TrioVecEvent needs to extract only a small set of features for candidate classification. With the learned multimodal embeddings, all of the features are quite cheap to compute.

5.8.5 Feature Importance

Finally, we measure the importance of different features for candidate classification. Our measurement is based on the Random Forest Classifier, by computing how many times a feature is used for dividing the training samples in the learned tree ensemble. Figure 5.9 plots the normalized fractions of all the features, where larger values indicate higher importance. As shown, the spatial concentrations turn out to be the most important features on both data sets. This is expected, as a spatiotemporal event usually occurs at a specific point-of-interest, resulting in a geo-topic cluster that is spatially compact. The unusualness measures also serve as important indicators for the classifier, which clearly shows that the embeddings serve as useful knowledge for distinguishing unusual events from routine activities. The other four features (burstiness, semantic concentration, spatiotemporal unusualness, and temporal concentration) act as useful indicators as well, receiving considerable weights.

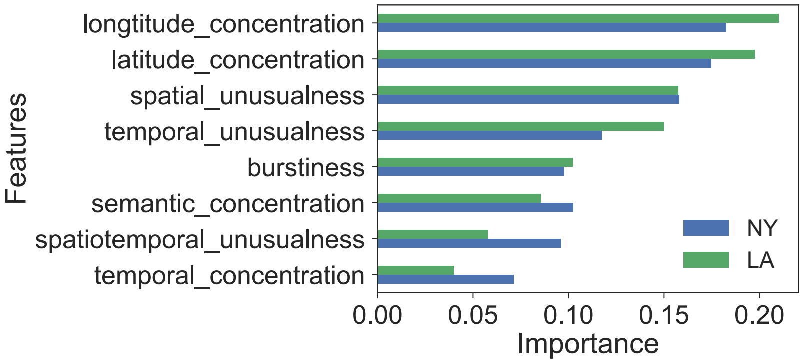


Figure 5.9: The importance of different features for candidate classification on LA and NY.

5.9 SUMMARY

In this chapter, we have proposed the TrioVecEvent method to detect abnormal spatiotemporal events in a three-dimensional space. With the multimodal embeddings of the location, time, and text, TrioVecEvent first obtains quality geo-topic clusters in the query chunk to ensure a high coverage of the underlying events. It then extracts a set of features to characterize the candidates, such that the true spatiotemporal events can be accurately identified. Our extensive experiments have demonstrated that TrioVecEvent improves the accuracy of the state-of-the-art method significantly while achieving good efficiency. Notably, it achieves up to 80% precision and 60% pseudo recall with a small amount of training data—such performance makes it feasible to be deployed for real-world abnormal event detection.

CHAPTER 6: CONCLUSION

6.1 SUMMARY

In this thesis, we have proposed a minimally supervised framework for turning unstructured text data into multi-dimensional knowledge. In the proposed framework, we have addressed two core questions on multi-dimensional mining of unstructured text data:

1. **Bringing multi-dimensional, multi-granular structures to the unstructured data.** In the first part of the thesis, we proposed to organize massive unstructured data into a cube structure, which allows end users to retrieve desired data with declarative queries along multiple dimensions at varied granularities. We show that it is feasible to learn task-aware embeddings to address the central subtasks for cube construction without labeled data. Specifically, we proposed unsupervised algorithms for taxonomy generation and document allocation: (1) Our method TaxoGen is capable of organizing a collection of terms into a topic taxonomy in an unsupervised way. TaxoGen learns locally adapted embeddings for taxonomy construction and uses hierarchical adaptive clustering for assigning terms onto proper levels. Such a design significantly outperforms state-of-the-art hierarchical topic models. (2) Our method Doc2Cube allocates documents into the cube structure by learning dimension-specific representations of documents. Doc2Cube does not require excessive labeled documents as training data but only surface label names as seed information, yet it still achieves inspiring classification performance.

2. **Discovering multi-dimensional knowledge in the cube space.** In the second part of the thesis, we proposed methods for discovering multi-dimensional patterns in the cube space. The general principle of multi-dimensional pattern discovery is simultaneously modeling multiple factors to uncover their collective behaviors. Under this principle, we developed algorithms that leverage multimodal embeddings for multi-dimensional knowledge discovery. Specifically, we first investigated the cross-dimension prediction problem—how to make accurate predictions across different dimensions. We designed the CrossMap method. It learns quality multimodal embeddings with a semi-supervised paradigm, which leverages external knowledge to guide the embedding learning process and meanwhile can operate in an online fashion to emphasize most recent information. Then we study the problem of abnormal spatiotemporal event detection. By combining multimodal embeddings and latent variable models, our pro-

posed method TrioVecEvent first detects geo-topic clusters in a multi-dimensional space, and extracts a small set of features to pinpoint truly abnormal events

With the above two modules, this thesis contributes a general and integrated framework. It allows end users to turn unstructured data into useful multi-dimensional knowledge effortlessly because of two properties. First, **it offers flexibility because of the multi-dimensional and multi-granular nature**. By organizing unstructured data into a cube and extracting patterns in the cube space, our work eases the process of on-demand multi-dimensional mining. The users can effortlessly identify relevant data with multi-dimensional, muli-granular queries; and subsequently apply existing mining primitives (*e.g.*, summarization, visualization) or our proposed methods for acquiring useful knowledge. Second, **it addresses the label scarcity bottleneck for mining multi-dimensional knowledge from text**. The algorithms in both the cube construction and exploitation modules require no or little labeled training data. As such, the end users can use the proposed framework to structure and mine massive text data where large-scale labeled data are expensive to obtain.

6.2 FUTURE WORK

While working on this thesis, we see several promising future directions of extending the proposed algorithms, which we discuss here.

Alleviating label scarcity with data locality. The lack of sufficient labeled data has become the major bottleneck that prevents many supervised learning techniques from being applied. Such a bottleneck goes beyond text data, and an important strategy for dealing with label scarcity is to transfer information from one domain to another. Our proposed framework is capable of organizing unstructured data into a multi-dimensional cube structure, where as the data instances in sibling cells are closely related. In the future, it is interesting to leverage such data locality to fight against label scarcity. Take sentiment analysis as an example, assume one cube cell consists of few labeled instances, can we transfer information from its sibling and parent cells? Which cells should we give more priorities to during the transferring process? Those issues are new and challenging research questions, but hold vast potential to improve existing transfer learning paradigm by virtue of the data locality with the cube structure.

Accelerating machine learning by online model aggregation. Practically, users' demands for statistical models can be ad-hoc and context-specific. From the same dataset, different users may select totally different subsets and learn models on their own selected data. Model training, however, can be costly. Can we avoid training models from scratch for an ad-hoc data subset? The cube structure serves as a promising direction for addressing this bottleneck. Inspired by existing OLAP techniques, it is interesting to leverage pre-computation to enable fast online model serving. The key philosophy is to train local models in different chunks of the data cube and aggregate pre-trained local models for online model serving. This will largely accelerate the knowledge discovery process from data, but new model materialization and aggregation techniques need to be designed to realize such a functionality.

Structure-aided interactive data mining. In many applications, acquiring knowledge from data is an interactive process where people and machines need to collaborate with each other. There is great potential to leverage our work to facilitate such a human-in-the-loop process: (1) machines accept user-selected data, perform data analysis along different dimensions and granularities, and provide *interpretable patterns and visualizations*; and (2) users make sense of the resultant patterns and visual cues, adjust their data selection schemas, and *provide feedbacks to guide the machines* to extract more useful knowledge. To realize this goal, several research problems need to be addressed: How to design cube materialization strategies that return user-desired results in real or near-real time? How to develop cube-tailored visualization techniques and interfaces to help users more easily gain useful knowledge? How to leverage user feedbacks to learn effective policies that intelligently explore different cells in the cube to better satisfy users' information needs?

As a final note, with the ever-increasing digitalization process, we anticipate both the complexity and volume of data will increase continuously in the next few years. Our work in this thesis has the potential to serve as a general-to-use knowledge acquisition framework in taming complex datasets, by fighting against data heterogeneity and label scarcity, and allowing users to structure and mine with their data effortlessly. We envision the framework extensible to more data types, and will take it as a start point to continue exploring how to further deal with other challenges in large-scale data mining scenarios.

REFERENCES

[1] H. Abdelhaq, C. Sengstock, and M. Gertz. Eventweet: Online localized event detection from twitter. *PVLDB*, 6(12):1326–1329, 2013.

[2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *VLDB*, pages 81–92, 2003.

[3] C. C. Aggarwal and K. Subbian. Event detection in social streams. In *SDM*, pages 624–635, 2012.

[4] C. C. Aggarwal and C. Zhai. A survey of text classification algorithms. In *Mining Text Data*, pages 163–222. Springer, 2012.

[5] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ACM DL*, pages 85–94, 2000.

[6] J. Allan, R. Papka, and V. Lavrenko. On-line new event detection and tracking. In *SIGIR*, pages 37–45, 1998.

[7] L. E. Anke, J. Camacho-Collados, C. D. Bovi, and H. Saggion. Supervised distributional hypernym discovery via domain adaptation. In *EMNLP*, pages 424–435, 2016.

[8] M. Bansal, D. Burkett, G. de Melo, and D. Klein. Structured learning for taxonomy induction with belief propagation. In *ACL*, pages 1041–1051, 2014.

[9] K. Batmanghelich, A. Saeedi, K. Narasimhan, and S. Gershman. Nonparametric spherical topic modeling with word embeddings. In *ACL*, 2016.

[10] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*, pages 17–24, 2003.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(1):993–1022, 2003.

[12] R. J. Byrd, S. R. Steinhubl, J. Sun, S. Ebadollahi, and W. F. Stewart. Automatic identification of heart failure diagnostic criteria, using text analysis of clinical notes from electronic health records. *International Journal of Medical Informatics*, 83(12):983–992, 2014.

[13] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.

[14] M. Chang, L. Ratinov, D. Roth, and V. Srikumar. Importance of semantic representation: Dataless classification. In *AAAI*, pages 830–835, 2008.

[15] S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. *ACM Sigmod Record*, 26(1):65–74, 1997.

[16] L. Chen and A. Roy. Event detection from flickr data through wavelet-based spatial analysis. In *CIKM*, pages 523–532, 2009.

[17] X. Chen, Y. Xia, P. Jin, and J. A. Carroll. Dataless text classification with descriptive LDA. In *AAAI*, pages 2224–2231, 2015.

[18] P. Cimiano, A. Hotho, and S. Staab. Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text. In *ECAI*, pages 435–439, 2004.

[19] B. Cui, J. Yao, G. Cong, and Y. Huang. Evolutionary taxonomy construction from dynamic tag space. In *WISE*, pages 105–119, 2010.

[20] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):224–227, 1979.

[21] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1/2):143–175, 2001.

[22] F. Diaz, B. Mitra, and N. Craswell. Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*, 2016.

[23] B. Ding, B. Zhao, C. X. Lin, J. Han, and C. Zhai. Topcells: Keyword-based search of top-k aggregated documents in text cube. In *ICDE*, pages 381–384, 2010.

[24] D. Downey, C. Bhagavatula, and Y. Yang. Efficient methods for inferring large sparse topic hierarchies. In *ACL*, pages 774–784, 2015.

[25] W. Feng, C. Zhang, W. Zhang, J. Han, J. Wang, C. Aggarwal, and J. Huang. Streamcube: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream. In *ICDE*, pages 1561–1572, 2015.

[26] J. Foley, M. Bendersky, and V. Josifovski. Learning to extract local events from the web. In *SIGIR*, pages 423–432, 2015.

[27] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu. Learning semantic hierarchies via word embeddings. In *ACL*, pages 1199–1209, 2014.

[28] G. P. C. Fung, J. X. Yu, P. S. Yu, and H. Lu. Parameter free bursty events detection in text streams. In *VLDB*, pages 181–192, 2005.

[29] A. Gandomi and M. Haider. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35:137–144, 2015.

[30] P. Giridhar, S. Wang, T. F. Abdelzaher, J. George, L. Kaplan, and R. Ganti. Joint localization of events and sources in social networks. In *DCOSS*, pages 179–188, 2015.

[31] S. Gopal and Y. Yang. Von mises-fisher clustering models. In *ICML*, pages 154–162, 2014.

[32] G. Grefenstette. Inriasac: Simple hypernym extraction methods. In *SemEval@NAACL-HLT*, 2015.

[33] V. Ha-Thuc and J. Renders. Large-scale hierarchical text classification without labelled data. In *WSDM*, pages 685–694, 2011.

[34] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers Inc., 3rd edition, 2011.

[35] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):84, 1970.

[36] Q. He, K. Chang, and E.-P. Lim. Analyzing feature trajectories for event detection. In *SIGIR*, pages 207–214, 2007.

[37] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545, 1992.

[38] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[39] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis. Discovering geographical topics in the twitter stream. In *WWW*, pages 769–778, 2012.

[40] M. Jiang, J. Shang, T. Cassidy, X. Ren, L. M. Kaplan, T. P. Hanratty, and J. Han. Metapad: Meta pattern discovery from massive text corpora. In *KDD*, pages 877–886, 2017.

[41] T. Joachims. Transductive inference for text classification using support vector machines. In *ICML*, pages 200–209, 1999.

[42] W. Kang, A. K. H. Tung, W. Chen, X. Li, Q. Song, C. Zhang, F. Zhao, and X. Zhou. Trendspedia: An internet observatory for analyzing and visualizing the evolving web. In *ICDE*, pages 1206–1209, 2014.

[43] C. C. Kling, J. Kunegis, S. Sizov, and S. Staab. Detecting non-gaussian geographical topics in tagged photo collections. In *WSDM*, pages 603–612, 2014.

[44] Y. Ko and J. Seo. Automatic text categorization by unsupervised learning. In *COLING*, pages 453–459, 2000.

[45] Z. Kozareva and E. H. Hovy. A semi-supervised method to learn and construct taxonomies using the web. In *ACL*, pages 1110–1118, 2010.

[46] J. Krumm and E. Horvitz. Eyewitness: Identifying local events via space-time signals in twitter feeds. In *SIGSPATIAL*, pages 20:1–20:10, 2015.

[47] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. On semi-automated web taxonomy construction. In *WebDB*, pages 91–96, 2001.

[48] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196, 2014.

[49] C. Li, A. Sun, and A. Datta. Twevent: segment-based event detection from tweets. In *CIKM*, pages 155–164, 2012.

[50] H. Li, G. Fei, S. Wang, B. Liu, W. Shao, A. Mukherjee, and J. Shao. Bimodal distribution and co-bursting in review spam detection. In *WWW*, pages 1063–1072, 2017.

[51] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, pages 661–670, 2010.

[52] R. Li, K. H. Lei, R. Khadiwala, and K.-C. Chang. Tedas: A twitter-based event detection and analysis system. In *ICDE*, pages 1273–1276, 2012.

[53] Y. Li, J. Nie, Y. Zhang, B. Wang, B. Yan, and F. Weng. Contextual recommendation based on text mining. In *COLING*, pages 692–700, 2010.

[54] C. X. Lin, B. Ding, J. Han, F. Zhu, and B. Zhao. Text cube: Computing IR measures for multidimensional text database analysis. In *ICDM*, pages 905–910, 2008.

[55] X. Liu, Y. Song, S. Liu, and H. Wang. Automatic taxonomy construction from keywords. In *KDD*, pages 1433–1441, 2012.

[56] Y. Lu and C. Zhai. Opinion integration through semi-supervised topic modeling. In *WWW*, pages 121–130, 2008.

[57] A. T. Luu, J. Kim, and S. Ng. Taxonomy construction using syntactic contextual evidence. In *EMNLP*, pages 810–819, 2014.

[58] A. T. Luu, Y. Tay, S. C. Hui, and S. Ng. Learning term embeddings for taxonomic relation identification using dynamic weighting neural network. In *EMNLP*, pages 403–413, 2016.

[59] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(85):2579–2605, 2008.

[60] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD*, pages 1155–1158, 2010.

[61] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *WWW*, pages 533–542, 2006.

[62] M. Mendoza, E. Alegría, M. Maca, C. A. C. Lozada, and E. León. Multidimensional analysis model for a document warehouse that includes textual measures. *Decision Support Systems*, 72:44–59, 2015.

[63] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[64] D. M. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *ICML*, pages 633–640, 2007.

[65] K. P. Murphy. *Machine learning: a probabilistic perspective.* MIT press, 2012.

[66] N. Nakashole, G. Weikum, and F. Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *EMNLP*, pages 1135–1145, 2012.

[67] G. Nunez-Antonio and E. Gutiérrez-Pena. A bayesian analysis of directional data using the von mises–fisher distribution. *Communications in Statistics—Simulation and Computation®*, 34(4):989–999, 2005.

[68] A. Panchenko, S. Faralli, E. Ruppert, S. Remus, H. Naets, C. Fairon, S. P. Ponzetto, and C. Biemann. Taxi at semeval-2016 task 13: a taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *SemEval@NAACL-HLT*, 2016.

[69] J. M. Pérez-Martínez, R. Berlanga-Llavori, M. J. Aramburu-Cabo, and T. B. Pedersen. Contextualizing data warehouses with documents. *Decision Support Systems*, 45(1):77–94, 2008.

[70] S. P. Ponzetto and M. Strube. Deriving a large-scale taxonomy from wikipedia. In *AAAI*, pages 1440–1445, 2007.

[71] M. Quezada, V. Peña-Araya, and B. Poblete. Location-aware model for news events in social media. In *SIGIR*, pages 935–938, 2015.

[72] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, pages 1524–1534, 2011.

[73] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *TREC*, pages 109–126, 1994.

[74] D. Roy, D. Paul, M. Mitra, and U. Garain. Using word embeddings for automatic query expansion. *CoRR*, abs/1606.07608, 2016.

[75] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860, 2010.

[76] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. Twitterstand: news in tweets. In *GIS*, pages 42–51, 2009.

[77] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[78] J. Seitner, C. Bizer, K. Eckert, S. Faralli, R. Meusel, H. Paulheim, and S. P. Ponzetto. A large database of hypernymy relations extracted from the web. In *LREC*, 2016.

[79] R. Shearer and I. Horrocks. Exploiting partial information in taxonomy construction. *The Semantic Web-ISWC 2009*, pages 569–584, 2009.

[80] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu. Fake news detection on social media: A data mining perspective. *SIGKDD Explorations*, 19:22–36, 2017.

[81] S. Sizov. Geofolk: latent spatial semantics in web 2.0 social media. In *WSDM*, pages 281–290, 2010.

[82] Y. Song and D. Roth. On dataless hierarchical text classification. In *AAAI*, pages 1579–1585, 2014.

[83] J. Tang, M. Qu, and Q. Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pages 1165–1174, 2015.

[84] F. Tao, K. H. Lei, J. Han, C. Zhai, X. Cheng, M. Danilevsky, N. Desai, B. Ding, J. Ge, H. Ji, R. Kanade, A. Kao, Q. Li, Y. Li, C. X. Lin, J. Liu, N. C. Oza, A. N. Srivastava, R. Tjoelker, C. Wang, D. Zhang, and B. Zhao. Eventcube: multi-dimensional search and mining of structured and text data. In *KDD*, pages 1494–1497, 2013.

[85] F. Tao, C. Zhang, X. Chen, M. Jiang, T. Hanratty, L. Kaplan, and J. Han. Doc2cube: Automated document allocation to text cube via dimension-aware joint embedding. In *ICDM*, 2018.

[86] F. Tao, H. Zhuang, C. W. Yu, Q. Wang, T. Cassidy, L. R. Kaplan, C. R. Voss, and J. Han. Multi-dimensional, phrase-based summarization in text cubes. *IEEE Data Engineering Bulletin*, 39(3):74–84, 2016.

[87] C. Wang, M. Danilevsky, N. Desai, Y. Zhang, P. Nguyen, T. Taula, and J. Han. A phrase mining framework for recursive construction of a topical hierarchy. In *KDD*, 2013.

[88] C. Wang, J. Wang, X. Xie, and W.-Y. Ma. Mining geographic knowledge using location aware topic model. In *GIR*, pages 65–70, 2007.

[89] C. Wang, X. Yu, Y. Li, C. Zhai, and J. Han. Content coverage maximization on word networks for hierarchical topic summarization. In *CIKM*, pages 249–258, 2013.

[90] P. Warrer, E. H. Hansen, L. Juhl-Jensen, and L. Aagaard. Using text-mining techniques in electronic patient records to identify adrs from medicine use. *British Journal of Clinical Pharmacology*, 73-5:674–84, 2012.

[91] K. Watanabe, M. Ochi, M. Okabe, and R. Onai. Jasmine: a real-time local-event detection system based on geolocation information propagated to microblogs. In *CIKM*, pages 2541–2544, 2011.

[92] J. Weeds, D. Clarke, J. Reffin, D. J. Weir, and B. Keller. Learning to distinguish hypernyms and co-hyponyms. In *COLING*, 2014.

[93] J. Weng and B.-S. Lee. Event detection in twitter. In *ICWSM*, pages 401–408, 2011.

[94] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, pages 481–492, 2012.

[95] H. Yang and J. Callan. A metric-based framework for automatic taxonomy induction. In *ACL*, pages 271–279, 2009.

[96] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, pages 1480–1489, 2016.

[97] Z. Yin, L. Cao, J. Han, C. Zhai, and T. S. Huang. Geographical topic discovery and comparison. In *WWW*, pages 247–256, 2011.

[98] Z. Yu, H. Wang, X. Lin, and M. Wang. Learning term embeddings for hypernymy identification. In *IJCAI*, pages 1390–1397, 2015.

[99] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Who, where, when and what: discover spatio-temporal topics for twitter users. In *KDD*, pages 605–613, 2013.

[100] C. Zhang, D. Lei, Q. Yuan, H. Zhuang, L. M. Kaplan, S. Wang, and J. Han. Geoburst+: Effective and real-time local event detection in geo-tagged tweet streams. *ACM Transactions on Intelligent Systems and Technology*, 9(3):34:1–34:24, 2018.

[101] C. Zhang, L. Liu, D. Lei, Q. Yuan, H. Zhuang, T. Hanratty, and J. Han. Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams. In *KDD*, pages 595–604, 2017.

[102] C. Zhang, F. Tao, X. Chen, J. Shen, M. Jiang, B. M. Sadler, M. Vanni, and J. Han. Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering. In *SIGKDD*, pages 2701–2709, 2018.

[103] C. Zhang, K. Zhang, Q. Yuan, H. Peng, Y. Zheng, T. Hanratty, S. Wang, and J. Han. Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning. In *WWW*, 2017.

[104] C. Zhang, K. Zhang, Q. Yuan, F. Tao, L. Zhang, T. Hanratty, and J. Han. React: Online multimodal embedding for recency-aware spatiotemporal activity modeling. In *SIGIR*, pages 245–254, 2017.

[105] C. Zhang, G. Zhou, Q. Yuan, H. Zhuang, Y. Zheng, L. Kaplan, S. Wang, and J. Han. Geoburst: Real-time local event detection in geo-tagged tweet streams. In *SIGIR*, pages 513–522, 2016.

[106] D. Zhang, C. Zhai, and J. Han. Topic cube: Topic modeling for OLAP on multidimensional text databases. In *SDM*, pages 1124–1135, 2009.

[107] B. Zhao, C. X. Lin, B. Ding, and J. Han. Texplorer: keyword-based object search and exploration in multidimensional text databases. In *CIKM*, pages 1709–1718, 2011.

[108] L. Zhao, F. Chen, C. Lu, and N. Ramakrishnan. Multi-resolution spatial event forecasting in social media. In *ICDM*, pages 689–698, 2016.

[109] L. Zhao, Q. Sun, J. Ye, F. Chen, C. Lu, and N. Ramakrishnan. Multi-task learning for spatio-temporal event forecasting. In *KDD*, pages 1503–1512, 2015.

[110] L. Zhao, J. Ye, F. Chen, C. Lu, and N. Ramakrishnan. Hierarchical incomplete multi-source feature learning for spatiotemporal event forecasting. In *KDD*, pages 2085–2094, 2016.

[111] J. Zhu, Z. Nie, X. Liu, B. Zhang, and J.-R. Wen. Statsnowball: a statistical approach to extracting entity relationships. In *WWW*, pages 101–110, 2009.