About SIGKDD Explorations
Explorations is published twice yearly, in June/July and December/January each year. After the first two volumes, frequency may increase to quarterly. The newsletter is distributed in hardcopy form to all members of the ACM SIGKDD. It is also sent to ACM’s network of libraries. Additionally, issues are published on the web and are free to the general public (http://www.acm.org/sigkdd/explorations/).

Our goal is to make SIGKDD Explorations an informative, rapid means of publication and a dynamic forum for communication with the Knowledge Discovery and Data Mining community. SIGKDD membership is growing at a very fast pace, and with KDD being a multi-disciplinary field, we hope that Explorations will facilitate its fusion and enhance the sense of community. Submissions will be reviewed by the editor and/or associate and guest editors as appropriate. We are particularly interested in short research and survey articles on various aspects of data mining and KDD. Explorations is also a forum for publishing position papers, controversial positions, challenges to the community, product reviews, book reviews, news items and other items of interest to the field. Please see:
http://www.acm.org/sigkdd/explorations/instructions.htm

Advertiser Information:
Explorations accepts advertisements related to data mining and KDD, including company, book, vendor, and service advertisements. For rates and instructions on submitting an ad, please see:
http://www.acm.org/sigkdd/explorations/instructions.htm#advertise

Notice to Contributing Authors of SIGKDD Explorations:
By submitting your article for distribution in this Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library
- to allow users to copy and distribute the article for noncommercial, educational or research purposes.

However, as a contributing author, you retain the copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.

Notice to Past Authors of ACM-Published Articles:
ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written work that was previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform you respective editors and permissions@acm.org, stating the title of the work, the author(s), and where and when published.
Crime in Urban Areas: 
A Data Mining Perspective

Xiangyu Zhao and Jiliang Tang 
Data Science and Engineering Lab, Michigan State University
{zhaoxi35, tangjili}@msu.edu

ABSTRACT
Urban safety and security play a crucial role in improving life quality of citizen and the sustainable development of urban. Traditional urban crime research focused on leveraging demographic data, which is insufficient to capture the complexity and dynamics of urban crimes. In the era of big data, we have witnessed advanced ways to collect and integrate fine-grained urban, mobile, and public service data that contains various crime-related sources as well as rich environmental and social information. The availability of big urban data provides unprecedented opportunities, which enable us to conduct advanced urban crime research. Meanwhile, environmental and social crime theories from criminology provide better understandings about the behaviors of offenders and complex patterns of crime in urban. They can not only help bridge the gap from what we have (big urban data) to what we want to understand about urban crime (urban crime analysis); but also guide us to build computational models for crime. In this article, we give an overview to key theories from criminology, summarize crime analysis on urban data, review state-of-the-art algorithms for various types of computational crime tasks and discuss some appealing research directions that can bring the urban crime research into a new frontier.

1. INTRODUCTION
We are living in a rapidly urbanizing world. The United Nations predicted that by 2050 about 64% of the developing world and 86% of the developed world will be urbanized, which means that the urban population will be bigger than the world population today. Nowadays, the relationship between urbanization and inequalities of urbanites (such as education level and wealth gap) has been extensively studied [4; 73], and a good amount of research suggests that places with inequalities are more likely to have high crime rates [29; 8; 50]. For instance, over the period of 1980 to 2000, recorded crimes increased from 2300 to 3000 for every 100,000 people [94]. Recent studies have shown that urban safety is closely related to the quality of citizen’s life and the sustainable development of urban [28]. Safety is one of the most fundamental physical and psychological needs of residents. Meanwhile, sustainable urban development will only be achieved when well-planned city-wide, gender-sensitive, community-based, integrated and comprehensive urban crime prevention and safety strategies have been put in place.

Traditional urban crime research is mainly based on conventional demographic data, i.e., statistical socioeconomic characteristics of a population, such as education level [36], income level and wealth gap [59; 75], and ethnic and religious difference [7]. However, demographic data is insufficient to understand the dynamics and complexity of crimes. First, most demographic features are relatively stable over an extended period of time, which cannot capture the dynamic nature within a specific community. Second, the vast majority of communities in the urban share similar demographic features, thus it becomes difficult to capture the differences between different communities [98]. Recently, with the massive development of new techniques for fine-grained data collection and integration, numerous urban crime-related data has been recorded, which provides sources that contain helpful context information about urban crime. For instance, human mobility provides useful environmental factors such as the function of a region and residential stability, which can significantly impact criminal activities according to environmental criminology [9]; while meteorological data like weather information has been proven to be related to urban crimes [26; 80]. Thus, big urban data contains rich and fine-grained context information about when and where the data is collected. Such information not only enables us to understand the dynamics of crime such as how crime evolves; but also allows us to study crime from various perspectives. Hence, big urban data provides us unprecedented opportunities to conduct advanced investigations on urban crime. On the one hand, there are many criminal theories developed from criminology to explain various types of criminal phenomena. For instance, according to environmental criminology such as routine activity theory [25] and rational choice theory [27], crime distribution is highly determined by time and space, and environmental factors offered by human mobility can significantly affect criminal activities. Meanwhile, social criminology such as social disorganization theory directly links crime rates to neighborhood ecological characteristics [85], while culture conflict theory suggests that the root cause of criminality can be found in a clash of values between differently socialized groups over what is acceptable or proper behavior. These theories explore many aspects of criminal behaviors and are crucial to understand and explain how and why crime occurs. Therefore, they can help bridge the gap from what we have (big urban safety data) to what we want to understand about urban crimes (urban crime analysis).

On the other hand, urban data is typically large-scale, noisy,
dynamic and heterogeneous, thus efficient and effective computational solutions are desired. In order to facilitate crime research with big urban data, a variety of computational tasks have been proposed, which have encouraged a large body of computational models by integrating criminal theories as well as crime patterns. It is timely and necessary to provide an overview about urban crime from a data mining perspective. Then, the remaining of the article is organized as follows. In Section 2, we introduce environmental and social criminal theories from criminology. Then we summarize the major urban crime patterns in Section 3. In Section 4, we introduce key computational crime tasks with representative algorithms. Finally, we conclude the work with discussions on possible research directions.

2. CRIMINOLOGY

Crime is a complicated and multidimensional event occurring when the law, offender and target (person or object) converge within time and place [10]. Understanding the offenders’ behavior and crime patterns plays an essential role in understanding crime. Consequently, it is beneficial to be acquainted with the theories from criminology.

2.1 Environmental Criminal Theories

Environmental criminology focuses on criminal patterns within particularly built environment and analyzes the impacts of the external variables on people’s emotional behavior. These consist of space (geography), time, law, and offender, in addition to target or victim.

2.1.1 Routine Activity Theory

Routine activity theory explains crime in terms of crime opportunities that happen in daily life [37; 25]. Three elements should converge in time and space for a crime opportunity, i.e., a motivated offender, a suitable target or victim, and the absence of a capable guardian. A guardian at a place, like a street, could contain security guards or even ordinary pedestrians who would witness the criminal act and possibly intervene or report it to police. This theory is expanded by the addition of the fourth element of “place manager” who has the capacity to take nuisance abatement measures [35].

2.1.2 Crime Pattern Theory

Crime Pattern Theory is to understand why crimes are committed in particular areas. Crime is not really random – it is either planned or opportunistic. Based on the theory, crime happens when the activity space of the victim or target intersects with that of an offender. Crime Pattern Theory has three main notions – node, path and edge [22]. Node is an specific area of activity that an individual uses frequently. Path is the route that the individual takes to and from typical areas of activity in everyday life. Edges are the boundaries of an individual’s awareness space.

2.1.3 Rational Choice Theory

Rational choice theory aims to help in thinking about situational crime prevention [21]. The assumption is that crime is purposive behavior made to fulfill the criminal’s needs for things such as money, status and excitement. Meeting these requirements involves the making of decisions and choices as these are constrained by limits, ability, and the accessibility of relevant information [23]. For instance, if telephone or wallet is visible inside a vehicle and no person is around, it may tempt a criminal to grasp the opportunity.

2.1.4 Awareness Theory

Four components of crime have been suggested in [11]: victim, criminal, geo-temporal and legal. Concentrating on the particular spatial aspect of crime is important to understand the behavior of criminals. A crime’s space may be chosen either deliberately or accidentally by either the victim or the offender according to their life styles. A number of things have impact on the crime rate of an region. For example, what kind of individuals reside in a particular region and what kind of security can be obtained.

2.1.5 Broken Windows Theory

The broken windows theory is a criminal theory of the norm-setting and signaling impact of urban disorder and vandalism on extra offense and anti-social behavior [104]. The theory claims that sustaining and monitoring urban environment to avoid small offenses like public drinking and vandalism assists to produce an environment of order and lawfulness, thus preventing more severe offenses happening. The theory has been applied as an inspiration for many reforms in criminal policy, such as the controversial mass use of “stop, question, and frisk” by the New York City Police Department.

2.1.6 Crime Opportunity Theory

Crime opportunity theory suggests that analysts should search for concentrations of offense targets. For example, a dense downtown community without off-street parking could have many vehicles parked on the street. This kind of areas might become a region hotspot for thefts from vehicles; while a suburban inhabited by dual-income families will have few persons during weekdays, and their property is unprotected, thus their community can be a region hotspot of burglary. Note that in such condition, many levels of hotspots can exist simultaneously. Within region hotspots, identified by the subdivision in this case, might be streets with increased amounts of burglaries, and some of the houses on these streets might be broken into numerous times.

2.2 Social Criminal Theories

Theories in this family are used in a number of approaches as the conflict theory or structural conflict perspective in sociology are associated with crime. Social criminal theories emphasize poverty, absence of education, lack of marketable abilities, and subcultural values as fundamental causes of crime.

2.2.1 Social Disorganization Theory

Social Disorganization Theory directly links crime rates to community environmental characteristics [85]. Social disorganization theory postulates that an individual’s residential location is a substantial element shaping the chance the individual will become involved with illegal activities. The theory shows that, among determinants of an individual’s later illegal activity, residential location is more substantial than the individual’s characteristics (e.g., age, gender, or race). For instance, the theory indicates that youths from disadvantaged community participate in a subculture which approves of delinquency, and these types of youths thus acquire criminality within this social and ethnic setting.
2.2.2 Social Strain Theory

Strain theory indicates that mainstream culture is saturated with dreams of opportunity, freedom, and prosperity. The majority of people buy into this dream, and it will become a powerful cultural and psychological inspiration. If the social structure of opportunities is unequal and prevents the majority from realizing the dream, some of the people dejected will use illegal ways (crime) to realize it. Others may retreat or drop out into deviant subcultures (like gang members). Robert Agnew developed this theory to incorporate varieties of strain which were not derived from economic restrictions. This is known as “General Strain Theory” [68].

2.2.3 Culture Conflict Theory

The theory of culture conflict is linked to the disagreement over dissimilarities in values and beliefs. This is based on the idea that different cultures or classes cannot agree on what is common acceptable behavior. For instance, if the upper and middle classes work to make a living in a legal way, others may use illegitimate ways, such as stealing, to make a living.

2.2.4 Social Efficacy Theory

Recent evidence points to the role of social efficacy, which is the willingness of nearby residents to intervene with regard to the common good. This is dependent on mutual trust and solidarity among neighbors [88]. Communities that have a great deal of social efficacy have less offenses than these at low levels. Social efficacy is not a property of individual people or places, but a characteristic associated with groups of individuals.

2.2.5 Subcultural Theory

Subcultural theorists focus on small cultural groups fragmenting aside from the mainstream that form their own beliefs and meanings of life. It indicates that delinquency among lower class youths is a reaction towards the social norms of the middle class [24]. Several youth, especially from poorer regions where opportunities are few, might buy into social norms specific to those places that may consist of “toughness” and disrespect regarding authority. Criminal acts might result when youths adapt to norms of the deviant subculture [61].

2.2.6 Control Theory

Control Theory tries to describe why people do not become offender [49]. It recognizes four principal factors: (1) connection to others, (2) belief in ethical validity of principles, (3) responsibility to achievement, and (4) engagement in main-stream activities [54]. The more an individual has those factors, the less likely he/she become offender. This theory is extended with the fact that an individual with low self control is more likely to become offender.

2.2.7 Labeling Theory

Labeling theory claims that when a person is given the label of a offender, they might accept it and continue to commit crime or refuse it. Even those who initially refuse the label can eventually accept it as the label becomes more well-known especially among their peers. This stigma can be much more profound when the labels are about deviancy, and it is believed that this stigmatization can cause deviancy amplification. Klein [60] did a test which indicated that labeling theory influenced some youth offenders but not others.

3. URBAN CRIME PATTERN ANALYSIS

As suggested by theories from criminology, crime is highly related to time and location. Meanwhile, big urban data provides rich information about crime from temporal and spatial perspectives, which cultivated increasing efforts on temporal-spatial pattern analysis. Temporal-spatial pattern analysis is a procedure that obtains understanding from temporal-spatial related sources and generates understanding for crime analysts. In practice, understanding varies among different environments. In order to acquire appropriate relevant kinds of information, various kinds of temporal-spatial pattern analysis techniques should be leveraged [63].

3.1 Temporal Pattern Analysis

Criminal temporal patterns are complicated since temporal resources could be structured in various intervals like weeks, months, seasons, years and others [63]. Generally, the temporal crime analysis focuses on learning useful temporal patterns from sequential crime data. Types of temporal pattern analysis can be summarized as follows:

- Crime tendency refers to the change of a type of crime inside a given region and a long-term time period. For example, the property crime rates declined by double-digit percentages from 2008 to 2016 in the USA.
- Crime periodicity is defined as the repeating patterns of crime at time intervals, e.g., seasonal (i.e. annually recurring) crime patterns.
- Similarity search of crime aims to search crime sequences that are similar to a given crime sequence.
- Sequential behavior analysis tries to find an offender’s sequential behaviors before or after committing a crime, e.g., a burglar often buys drug after committing a burglary.

3.2 Spatial Pattern Analysis

Crimes are not evenly or randomly distributed in an urban area. Typically, crimes are dense in some regions and sparse in others. Spatial pattern analysis aims to learn the aggregation of crime, i.e., hotspots, inside a city. Additionally, crimes are proved to be correlated with environment contexts. In this subsection, we introduce crime hotspots and spatial factor analysis in detail.

- Crime hotspot is defined as a geographic location with more than normal quantity of crime activities, or a location where individuals have greater than normal risk of victimization [34]. On the contrary, there exist cold-spots with less than the normal density of crime. Some hotspots might be hotter than others because of the difference of crime density. Generally, hotspot analysis finds spatial patterns through spatial clustering.
- Spatial factor analysis aims to find the main spatial factors of crime [63]. The major hypothesis of spatial analysis is that crime should correlate with environment contexts and this hypothesis is supported by various criminal theories. For instance, according
to routine activity theory, three elements, i.e., a motivated offender, a suitable target or victim, and the absence of a capable guardian, are required to converge in time and space for a crime occurring.

3.3 Spatio-Temporal Pattern Analysis
Criminal spatio-temporal pattern analysis aims to obtain understanding from geo- and time-related crime data. The challenge is how to identify patterns from the dynamic interaction among space, time and crime [63] since crime patterns are believed to vary with time and location [89; 66; 51]. In this subsection, we will review important temporal-spatial patterns of urban crime.

- Earthquake-like pattern: The concentrating patterns suggest that an earthquake is likely to produce a series of aftershocks close to the area of the original earthquake [33]. Similar phenomena are observed in crime formation, such as burglars may repeatedly assault neighboring communities over a time period. This encourages applying seismology techniques like self-exciting point processes to model urban crime [30].

- Spatio-temporal hotspot: Crimes such as gang violence happen concentrated in time and space. Spatio-temporal hotspot is defined as a geographic location coupled with a time period where greater than normal amount of crimes occur. It aims to incorporate temporal patterns on spatial hotspot for crime analysis.

- Spatio-temporal correlations: Spatio-temporal correlations are explored in [113]. For a urban region, “intra-region temporal correlation” is observed– (i) for two consecutive time slots, they are likely to share similar crime numbers; and (ii) with the increase of differences between two time slots, the crime difference has the propensity to increase. Over all urban regions, “inter-region spatial correlation” is observed – (i) two geographically close regions have similar crime numbers; and (ii) with the increase of spatial distance between two regions, the crime difference tends to increase.

4. COMPUTATIONAL TASKS FOR URBAN CRIME
Big urban data is typically large-scale, noisy, dynamic and heterogeneous, which calls for efficient and effective computational solutions. Thus, in order to enhance crime research in big urban data era, numerous computational tasks have been proposed. In this section, we review key computational tasks for urban crime with representative algorithms.

4.1 Crime Rate Prediction
Crime rate prediction aims to predict the future crime rate of a given urban region. In this subsection, we categorize crime rate prediction models according to the data they use as prediction based on crime data, environmental context data, and social media data.

4.1.1 Prediction Based on Crime Data
Precise crime prediction 30 days ahead for small areas, like police precincts, is proposed in [48]. Prediction precision of univariate time series models are compared with techniques commonly employed by police. A fixed-effect regression model of absolute percent prediction error suggests that average offense number must be larger than 30 to obtain less than 20% prediction error. It is also found that Holt exponential smoothing is the most precise model for precinct-level crime prediction. In [17], autoregressive integrated moving average (ARIMA) is employed for near future prediction of property crime. Based on 50 weeks’ property crime data, an ARIMA model is built to predict crime number of 1 week ahead. It is found that ARIMA model has higher fitting and prediction precision than exponential smoothing.

A four-order tensor for crime forecasting is presented in [71]. The tensor encodes the longitude, latitude, time, and other related crimes. The tensor can tackle the data sparsity since each order is lower-dimensional. Additionally, the geometry structure is properly maintained in tensor. Leveraging the tensor framework, empirical discriminative tensor analysis algorithm is presented to acquire adequate discriminative information and reduce empirical risk simultaneously. In [107], a new feature selection and construction method is proposed for crime prediction by using temporal and spatial patterns. Multi-dimensional feature is denoted as spatio-temporal pattern that is built upon regional crime cluster distributions in different levels. Then a Cluster-Confidence-Rate-Boosting framework is presented to combine local spatio-temporal patterns into global crime pattern, which is then employed for crime prediction.

Temporal patterns of dynamics of violence are analyzed using a point process model in the scenario of urban crime prediction [65]. The rate of crimes is partitioned into the sum of a Poisson background rate and a self-exciting component in which crimes induce the growth in the rate of the process. Specifically, each crime produced by the process in turn produces a series of offspring crimes according to a Poisson distribution. The background rate is normally fixed for crimes. In [69], self-exciting point process models are implemented for predicting crimes. They leverage a nonparametric evaluation strategy to gain understanding of temporal-spatial triggering function and temporal tendencies in the background rate of burglary. Particularly, spatial heterogeneity in crime rates can be evaluated by background intensity estimation and the self-exciting effects detected in crime data.

4.1.2 Prediction Based on Environmental Context Data
Crime rate tendencies and periodicity are analyzed through a routine activity approach [25; 38] to predict crime. Specifically, it is assumed that the distribution of events far from homes raises the opportunities for offense and hence yields higher crime rates. The assumption can help understand crime rate tendencies in the United States 1947-1974 as a consequence of changes in such factors as labor force involvement and single-parent families. Seasonal crime patterns were analyzed for urban crime prediction [40]. An evaluation of annual, quarterly, and monthly crime data exhibited solid evidence that temperature has a positive impact on most kinds of crime. The influence was independent with seasonal variation. The key explanation is that higher temperatures trigger individuals to spend more time out of home, which is consistent with routine activity explanations for crime and has been revealed to raise the chance of crime. The results indicate that temperature is one major reason to be taken into consideration when describing quarter-to-quarter variations of urban crime.
4.1.3 Prediction Based on Social Media Data

Twitter posts with rich and event-based context is leveraged for predicting criminal incidents [100]. The framework contains two components. The first component is a spatio-temporal generalized additive model, which leverages a feature-based method to predict future crime at a given location and time. The second component extracts textual information through semantic role labeling-based latent Dirichlet allocation. In addition, a new feature selection approach is designed to discover essential features. A preliminary analysis of Twitter-based crime prediction is proposed in [101]. The method incorporates intelligent semantic analysis of Twitter posts, and dimensionality reduction through latent Dirichlet allocation. Twitter-specific linguistic analysis and mathematical topic modeling to locate discussion topics of Twitter across a urban are introduced in [45]. These topics are integrated into a crime prediction model. The authors find that the addition of Twitter information enhances crime prediction accuracy compared with kernel density estimation. It identifies several performance bottlenecks that influence the usage of Twitter in a real decision support system. In [1], Twitter content is employed for crime tendency forecast, in which a Twitter sampling approach is presented to gather historical data for handling the missing data problem over time. The experiments unveiled the relationship of Twitter content and crime tendency. Besides, some crime types like burglary are found to have closer relationship with the shared Twitter content than other types.

4.2 Crime Hotspot Detection

Crime hotspot detection (mapping) is a spatial mapping technique focusing on the identification of the concentration of crime events across the urban. In this subsection, we categorize crime hotspot detection methods based on the types of techniques they leverage, i.e., (1) KDE-based techniques: non-parametric method to calculate the probability density function of crimines, (2) Reaction-Diffusion-based techniques: a mathematical framework based on reaction-diffusion partial differential equations to learn the dynamic nature of crime hotspots, and (3) Other techniques: they include geographic boundary thematic mapping, grid thematic mapping, spatial ellipses and hotspots optimization tools.

4.2.1 KDE-based Techniques

Prospective crime hotspot detection methods are improved by analyzing interpolation technique, grid cell size, and bandwidth on the prediction precision of Kernel Density Estimation (KDE) [52]. Particularly, based on variations in essential user-defined settings that are parts of the interpolation process, this work presents scientific explanations of the quality of KDE hotspot maps. The analytical technique contains evaluating these effects across multiple crime types, such as assault, robbery and burglary. To quickly get an accurate hotspot map, an efficient approach to convert KDE hotspot map with low resolution to a new map with contour lines is designed in [31]. The outcome is a hotspot map with smooth boundaries, which is equally accurate to KDE hotspot maps leveraging smaller cell sizes, but has faster generation speed. The new maps are more natural illustration of the real world’s hotspots compared to the original KDE maps. Many helpful suggestions for setting parameters of KDE such as the grid cell size and search radius (bandwidth) are proposed for hotspot detection tasks [14; 34; 82].

4.2.2 Reaction-Diffusion-based Techniques

A computational framework based on reaction-diffusion partially differential equations is developed for studying the formation and dynamics of crime hotspots [87]. The framework is designed upon empirical evidence for how criminals move and interact with victims. Analysis suggests that crime hotspots from the recurring crimes diffuse locally, but not so far as to mix remote crime together. A nonlinear analysis is proposed to detect the crime hotspots through the reaction-diffusion system [86]. The authors discover amplitude equations that control the forming process of crime hotspot patterns through a perturbation method. Different from the super-critical hotspots discovered in existing work [87], sub-critical hotspots are discovered that arise upon sub-critical or trans-critical bifurcations with respect to the geometry. Enlightened by [86], a reaction-diffusion based approach is proposed to detect hotspots rigorously [5]. More specifically, the existence of steady states is demonstrated with multiple spikes of two types, i.e., (1) Multiple spikes having the same amplitude, and (2) Multiple spikes having different amplitudes. They leverage a strategy according to Liapunov-Schmidt reduction and improve it to the quasilinear crime hotspot model.

4.2.3 Other Techniques

Geographic boundary thematic mapping is a method to represent spatial distributions of urban crimes [81], which can rapidly generate hotspot maps and need little knowledge to interpret [103]. Boundary regions in this method are generally arbitrarily defined by government, e.g., police precinct. Crimes in the hotspot map concentrate on these regions which will then be shaded with respect to crime number inside them. Grid thematic mapping method is developed to mitigate the situation of different sizes and shapes of different regions such as police precincts [6; 62], in which grids with the same size and shape are drawn on an urban map with rich context using the study area. Thus all regions in the map are of uniform dimensions and are comparable, which is helpful to quickly and easily detect crime hotspots. Spatial ellipses is a hotspot detection software that locates hotspot within the study area [3]. It first finds the densest aggregation of crime locations on the map (hot clusters), and then fits a “standard deviation-ellipse” to each one. The ellipses rank crime clusters according to their sizes and properties. An hotspots optimization tool is presented to enhance the detection of hotspots through optimizing its boundary according to the spatial patterns of crime driving factors in [97; 96]. A pattern is introduced to indicate a mix of values of related variables which is able to distinguish hotspots and normal regions from the spatial perspective, named Geospatial Discriminative Patterns (GDPatterns) [32]. The proposed model automatically detects the crime hotspots and identifies GDPatterns between crime hotspots and normal regions simultaneously.

4.3 Next-Location Prediction

Next-Location Prediction aims to predict the location where an offender will commit a crime according to the offender’s historical trajectories or other information. A personalized
random walk based method for next-location prediction is proposed in [91]. To be specific, it leverages co-offending, crime trends and road network data to personalize the random walk process. According to crime pattern theory, criminals usually use their most familiar regions as part of their activity space. A probabilistic model is proposed to capture known criminals’ spatial trajectories in their activity space. This setting is then used to predict crime locations for criminals. Correlated walk analysis with several regression routines are designed for forecasting the behavior of a serial criminal in [64]. Based on the analysis of criminal’s repeated behavior in time, direction, and distance, the prediction of where and when the next crime will occur is produced. In [99], approaches that incorporate textual content into next-location prediction models are studied based on two hypothesis: (1) An person’s future spatial trajectory relates to his/her historical tweets, and (2) Crime rates correlate with the density of users’ spatial trajectories in the same region. Additionally, the relationship between these next-location predictions and the incidence of urban crimes is investigated, with the target to help future study into intelligent crime prediction. In [79], Rosso’s formula is introduced to predict crime’s next location with geographic profiling techniques. To be specific, a traffic network is incorporated for geographic profiling, and the next-location prediction problem is handled upon weighted traffic network, where shortest path between nodes are leveraged to replace Euclidean distance for accurate next crime location prediction. An agent-based simulation is designed to learn the influence of temporal pulse activities on the criminal location selection process [42]. The simulation offers a technique to modify spatio-temporal patterns as the consequence of some apriori special activities.

4.4 Criminal Network Analysis

Law enforcement and police departments have realized that criminal networks are important for crime analysis and prevention. Typically, a criminal network consists of (1) nodes: the individual actors within the criminal network such as offenders, and (2) ties: the relationships between actors, such as co-offenders and crime gang. In this subsection, we categorize criminal network analysis methods according to the techniques they employ as (1) Agent-based techniques: agents move in network and interact with each other, thus produce complex dynamics and patterns from simple behavioral rules, (2) Graph Theory-based techniques: Metrics (e.g. centrality) and techniques (e.g. community detection) from graph theory are employed for criminal network analysis, and (3) GIS-based techniques: Geographic Information System techniques are introduced to handle criminal network analysis.

4.4.1 Agent-based Techniques

An agent-based model to imitate the formation of street gangs is introduced in [53]. The motion dynamics of agents are combined into an evolving network of street gangs, which are influenced by prior interplays among agents in the system. Datasets of gangs, locations and criminal behaviors are incorporated into the model. The authors discover that highways, rivers, and the center locations of gangs’ activities affect the agents’ motions. The gang concentration through an interacting agent system is identified on a lattice [2]. Two-gang Hamiltonian framework is presented that agents have red or blue association but are otherwise indistinguishable. In this model, all indirect interplays happen through graffiti marks, on-site and on closest surrounding regions. The dynamic segregation [84] and dynamic subsequent variations [41; 67] models leverage agent-based techniques with structured time-invariable interactions [77] for criminal network analysis. They use structured rather than well-mixed networks since not all actors are connected to others in a gang or criminal network, and the interplays between actors generally follow an fixed manner that will not change over time.

4.4.2 Graph Theory-based Techniques

Analytical Hierarchy Process and Graph Theory are combined to solve gang crime problem [44]. The key purpose is to find the conspirators and produce a priority list according to message traffic within crime cases. To find one offender, it firstly quantifies the topics through Analytic Hierarchy Process, then builds the network model through Graph Theory and finally proves the relationship between this offender to the identified conspirators and non-conspirators. Criminal gangs are sensed and characterized in networks reconstructed from telephone network data [39]. Specifically, it presents an expert framework to reveal the main structure of criminal networks concealed in telephone data. This framework enables mathematical network analysis and community detection of telephone network data. It allows police departments to deeply understand the structure within criminal networks, discover offender leader and identify relationship among sub-groups. In [76], several police analytic tasks are handled regarding street gangs from the graph theory perspective. Particularly, it establishes the degree of membership for criminals who do not acknowledge the membership in a street gang, quickly recognizes sets of influential criminals through the tipping model, and decomposes gangs into sub-groups to identify criminal ecosystems. A unified model is proposed to bridge the conceptual gap between abstract crime dataset and co-offending network in [90]. Then it uses several centrality metrics, such as degree, closeness, betweenness, eigenvector and PageRank centrality, on the co-offending network and studies how leader criminal detection and elimination can help crime prevention. Cluster analysis is applied to detect criminal leaders, gangs and interplays between gangs within criminal network [105]. More specifically, an concept space idea is introduced to build links among criminals according to the similarities of criminals’ crime activities. The more two criminals take part in the same crime activities, the more likely to have a link between two criminals.

4.4.3 GIS-based Techniques

Influence of geography and social networks on gang violence is learned in [74]. Applying urban shootings data, it analyzes the impact of geographic proximity, organizational memory and group properties (e.g. reciprocity and transitivity) on gang violence in Chicago and Boston. Results show adjacency of criminal gang turf and previous conflict between gangs are solid predictors of future gang violence. A method for co-offense forecast is proposed in [92], where geographic, social, geo-social and similarity factors are used to classify offenders. To handle the skewed distribution problem of co-offending networks, it designs three types of criminal cooperation opportunities which are helpful to reduce the class
imbalance ratio, while maintaining half of the co-offenses. Criminal network among gang members are discovered in Los Angeles according to the rare observations of a mix of social connections and geographic regions of the individuals [95]. A similarity graph is built for the individuals and spectral clustering is leveraged to identify clusters in the graph. It analyzes various ways to encode the geo-social information on graph structure and the effect on the resulting clusterings. Spatial and social distance are employed to study the connections in youth co-offending network and essential features that influence co-offending network formation. Results demonstrate that spatial distance can better describe the overall structure, and social distance plays a role in network structure of close spatial distance.

4.5 Near Repeat Victimization

Crime does not happen randomly or evenly across time or space. The near-repeat victimization identifies the increased risk of repeat victimization at the same or surrounding regions and within a certain time period. This phenomenon has been repeatedly demonstrated for urban crime over the world. Near-repeat phenomenon is often jointly analyzed with hotspot detection tasks because their similar spatio-temporal distributions, but there are also some other techniques to handle near-repeat victimization, such as social network analysis.

4.5.1 Hotspots-based Techniques

Time span of repeat victimization and relationships between repeat victimization, deprivation and burglary hotspots are studied in [56]. It finds that the time span of repeat victimization presents an exponential manner. Results also reveal a clear connection between repeat victimization and deprivation, and declare that the geographical area of repeat victimizations may well contribute to the definition of burglary hotspots. Encouraged by the precepts of optimal foraging theory, burglary hotspots are found to shift over time [57], such that the repeating phenomenon of hotspots is not predictable over three months, but they have a tendency to move in a slippery way, i.e., moving to surrounding regions at successive time slots. Repeating burglary is studied based on police calls for service data [93]. It demonstrates: (1) hotspots can be discovered by mathematical analysis of spatial crime aggregation, (2) unstable hotspots are generally short-term concentrations of hot-dots, while stable hotspots appear to reveal more social and physical features of specific locations. In [102], the near-repeat phenomenon is used to analyze the risk levels around hotspots. To be specific, a temporal matrix is proposed to measure the fluctuation of risk around hotspots. The results show that (1) hotspots always exist, (2) space-time regions of high chance are always variable in space and time, (3) locations in the vicinity of hotspots concurrently share higher risk, and (4) crime risks around hotspots follow a wave diffusion process.

4.5.2 Other Techniques

Likelihood distribution functions for the time intervals between repeat crimes are learned in [88]. It compares these distributions to mathematically derived distributions where the repeat effects are due to solely consistent risk heterogeneity. It discovers that a form of event boosts is able to describe the observed distributions, while risk heterogeneity alone cannot, thus it models repeat victimization as a series of random activities, the likelihood of which changes once an crime occurs. Near repeat and social network techniques are incorporated on discovering crimes’ spatio-temporal patterns [55], in which burglary data of NYC is analyzed and compared upon average clustering coefficient, degree and closeness centrality. The clusters of near repeat crimes are efficiently calculated in [106]. First, R-tree is used to index crimes, where a crime is defined as a node and edges are built by range querying the vertex in R-tree, thus a graph forms. Cohesive subgraph techniques are used to find the crime chains. K-clique, k-truss, k-core plus DBSCAN algorithms are implemented in sequence regarding their diverse range of capacity to locate cohesive subgraphs.

4.6 Police Patrolling

Proper patrol route planning is one important application of crime analysis systems, which helps increase the effectiveness of police patrolling and improve public security simultaneously. In this subsection, we categorize this family of models according to the application tasks including (1) Patrol Area Allocation: partitioning a urban to precincts and arranging them to police officers, (2) Patrol Route Planning: designing the patrolling route for police car, and (3) Other Applications: they include police decision support systems, and spatial reorganization of police agents.

4.6.1 Patrol Area Allocation

A novel framework is designed for allocating patrol area against urban crime [46; 109; 108]. This framework models the connection between officers and urban crime as a dynamic Bayesian Network (DBN). Next, a series of improvements are made to the basic DBN causing in a compact model that has lower learning error. Further, by analyzing various Markov models, it is found that the number of the crime and the number of the defender in each region can impact the crime forecast, and combining hidden states within DBN can reduce prediction error. A bi-level optimization method are developed to handle the problem of spatial police patrol allocation aiming to enhance crime response speed [72]. It first designs a linear programming patrol response formulation and then use Bender’s decomposition to solve the optimization problem. The major challenge is that offenders may adjust the location and time of crime with respect to police patrols. To deal with this challenge, an iterative Bender’s decomposition approach is proposed. In [112], a police patrol discrete-event simulation model is applied to judge the patrol allocation plans. A response surface strategy is developed to discover optimal or sub-optimal allocation plans to improve response speed and reduce workload variation. An iterative searching process is designed to learn the connection between parameters in a redistricting algorithm and performance measures of allocation plans.

4.6.2 Patrol Route Planning

Planning effective and balanced routes for police patrolling is challenging with multiple police cars across different police distinct. Distributed services on road networks are introduced for patrolling route planning [16]. Specifically, they formulate this problem as a Min-Max Multiple-Depot Rural Postman Problem. To resolve the routing problem, they design an effective tabu-search-based algorithm and present three novel lower bounds to evaluate the routes. Rapid route planning is coupled with interactive spatio-temporal
hotspot exploration in [47]. The major components are: (1) a sketch-based method for dynamic route planning, which allows a police officer to fast establish a route across the city and discover the number and types of crime along the route, and (2) a spatio-temporal hotspot method considering time, location, season, and recent crime volume. A patrol routes planning method is designed to maximize the coverage of hotspots and minimize the patrolling distance simultaneously [15]. It treats the road network as graph, where node, edges and edge weights refer to intersections, streets and streets’ importance. These features as well as the topology of graph are then used to calculate the optimal patrol route. This method allows the automation of patrol planning and the dynamic adjustment in terms of constantly changing environment. A real-time patrol route planning approach for dynamic environments is proposed in [19; 18]. It first models the route planning problem in one patrol unit case. Then it designs an efficient algorithm to leverage cross entropy for real-time applications in practice. A graph-based MDP is presented to model the patrol routing process in [20], and then it proposes an $\epsilon$-optimal strategy to handle the dimensionality curse problem. This strategy is derived from the idea of $\epsilon$-optimal horizon approximation.

4.6.3 Other Applications
A decision support system, which merges predictive policing abilities with a patrolling districting model, is proposed for the planning of predictive patrolling regions [12]. The system efficiently and evenly identifies partitions of a district balanced based on the decision maker’s preferences. To analyze the crime records, it designs a method to describe the spatially and temporally indeterminate crime events. A multi-criteria police districting problem is studied with the consideration of the characteristics of region, risk, compactness, and mutual support [13]. The decision-maker can identify its preferences on the characteristics, workload balance, and efficiency. The model is resolved in the form of a heuristic algorithm. In [43], an application is proposed for assisting the spatial reorganization of police agents. To model activities of criminals, it employs self-organization strategy where police agents learn from their local regional, and make decisions based on this self-organization strategy and other environmental information. Two models to help police department design policy and prevention strategies against crimes are introduced in [70]. The first model develops a patrol dispatch strategy where police agents patrol separately without the knowledge of other agents’ locations, while in the second strategy, the patrols of police agents are jointly dispatched.

5. FUTURE RESEARCH DIRECTIONS
In this section, we discuss some insights and present some future directions.

5.1 More Crime Patterns
Urban crime is observed to have intrinsically complex spatio-temporal patterns with complicated urban configurations. However, the majority of existing algorithms can capture only certain aspects of patterns. Hence, comprehensive techniques are desired to learn complicated temporal-spatial patterns for urban crime analysis. Recently deep learning has been demonstrated in the capability of capturing complex spatio-temporal patterns for more accurate prediction in many urban computing tasks such as traffic flow forecasting [110; 111], and air quality prediction [78]. Therefore, one promising direction is to build novel models such as deep learning models to learn more complex spatio-temporal patterns for advancing urban crime analysis.

5.2 More Advanced Techniques
Urban crime is intrinsically complex because of its dynamic interplay with space, time and other factors like economics, environment and urban configuration. Thus more advanced techniques should be introduced and developed to handle crime analysis. For instance, deep reinforcement learning, which can continuously update polices during the interactions with environment, is suitable for capturing the dynamic nature of urban crime data. Besides, crime in urban is impacted by multiple sources such as meteorological data, point of interests (POIs) data and human mobility data [113]. The majority of existing algorithms handle multiple sources equally or in a linear manner, but fail to capture the nonlinear connections and subordinations among multiple sources [58]. This calls for advanced techniques to effectively incorporate features from multiple sources for crime analysis. Additionally, feature extraction from multiple sources should follow an automatic way since handcrafted features are insufficient to capture complex spatio-temporal patterns, which encourages to leverage End-to-End frameworks to automatically combine feature extraction and computational tasks of urban crime.

5.3 More Computational Tasks
The recent development of big data techniques has greatly advanced urban crime analysis, which provides unprecedented and unique opportunities to designs more sophisticated models to tackle practical policing tasks in real world. As an example, the stop-question-and-frisk program in New York City is a crime prevention policy that temporarily detains, questions and searches citizens on the street for weapons and contraband. However, this practice has been complained about its racism and failure to reduce crime such as burglary and robbery. Thus more efforts should be made on crime prevention strategies that maximize deterrent value and minimize infringement on the rights of citizens simultaneously. Besides, multiple urban tasks should be jointly considered for a safer and smarter city such as education, health, urban planning, economic development, employment, police, justice, immigration, poverty, integration, etc.

5.4 Urban Simulation
Policing strategies must be pre-evaluated before their active use in order to save the unnecessary cost of deployment and avoid negative impacts on urban safety. Thus, a urban environment simulation is necessary for the offline evaluation and visualization of new policing strategies. Moreover, urban environment simulation can allow researchers and police department to investigate, gain insights, and develop policing techniques to boost their communities, integrating the interplays between crimes, economy, and urban configurations.

6. CONCLUSION
Crime analysis plays a tremendously impactful role in the sustainable development of urban and the quality of citizen’s life. With recent advances in urban data sensing, collecting
and integrating technologies, a large amount of fine-grained urban crime-related data has been recorded with rich environmental and social information, which motivates a variety of computational tasks to advance urban crime analysis. In this article, we give an overview about urban crime from a computational perspective. We first review two families of criminal theories, i.e., environmental criminal theories and social criminal theories, and then key crime patterns in mining urban crime data. We introduce major computational urban crime tasks with representative algorithms. We also discuss some interesting research directions about computational crime with big urban data.

**Acknowledgements**

This research is supported by the National Science Foundation (NSF) under grant number IIS-1714741 and IIS-1715940.

### 7. REFERENCES

<table>
<thead>
<tr>
<th>Reference</th>
<th>Title</th>
</tr>
</thead>
</table>


A Survey on Anomaly detection in Evolving Data

[with Application to Forest Fire Risk Prediction]

Mahsa Salehi
Faculty of Information Technology
Monash University
Victoria 3800, Australia
mahsa.salehi@monash.edu

Lida Rashidi
Department of Computing and information
Systems, University of Melbourne
Victoria 3000, Australia
rashidi.l@unimelb.edu.au

ABSTRACT

Traditionally most of the anomaly detection algorithms have been designed for ‘static’ datasets, in which all the observations are available at one time. In non-stationary environments on the other hand, the same algorithms cannot be applied as the underlying data distributions change constantly and the same models are not valid. Hence, we need to devise adaptive models that take into account the dynamically changing characteristics of environments and detect anomalies in ‘evolving’ data. Over the last two decades, many algorithms have been proposed to detect anomalies in evolving data. Some of them consider scenarios where a sequence of objects (called data streams) with one or multiple features evolves over time. Whereas the others concentrate on more complex scenarios, where streaming objects with one or multiple features have causal/non-causal relationships with each other. The latter can be represented as evolving graphs. In this paper, we categorize existing strategies for detecting anomalies in both scenarios including the state-of-the-art techniques. Since label information is mostly unavailable in real-world applications when data evolves, we review the unsupervised approaches in this paper. We then present an interesting application example, i.e., forest fire risk prediction, and conclude the paper with future research directions in this field for researchers and industry.

1. INTRODUCTION

Anomalies are data points that are inconsistent with the distribution of the majority of data points [6]. Traditionally the problem of anomaly analysis has been considered by statistics community, where the outcome has been a number of invaluable publications [14; 34; 56]. According to the definition by Barnett and Lewis, an anomaly is "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" [14]. Hawkins defined an anomaly as "an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism" [34]. Anomalies are also known as rare events, abnormalities, deviants or outliers 1. Anomaly detection has received considerable attention in the field of data mining due to the valuable insights that the detection of unusual events can provide in a variety of applications. For example, in environmental monitoring of a wireless sensor network, outlier detection techniques can be applied to detect faulty sensors or interesting behavioural patterns.

The availability of data that is used for the task of anomaly detection varies based on the properties of the dataset. In a static dataset, the whole observations of objects are available and the anomalies are detected with regards to the whole dataset. There are comprehensive surveys of outlier detection in static data with respect to different scenarios that are presented in [22; 8; 16; 23; 42; 22]. On the other hand, all the observations may not be available at once and instances can arrive sequentially. The observations in the latter group are called data streams and they can be represented as a dynamic vector. In addition, objects may possess causal/non-causal relationships such as friendship, citation and communication links. This type of relational dataset can be represented as a graph. Similar to data streams, all the observations of objects (nodes in a graph) may not be available at once and they can arrive sequentially. Moreover, the relationships (edges between nodes in a graph) may change over time. This type of relational dataset is represented in an evolving graph. Although various approaches have been developed for anomaly detection, many of them as described in [22; 9] focus specifically on the task of anomaly detection in static data or static graph. However, detecting anomalies in dynamic datasets is a less explored area of research. In this paper, we focus on both evolving scenarios, i.e., data streams and evolving graphs, and provide a categorization of different techniques in each scenario. Note that there are three general application scenarios for outlier detection, i.e., supervised, semi-supervised and unsupervised. Since supervised and semi-supervised scenarios are rare in real-world applications due to the lack of label information regarding the outlierness of some observations, we only survey the unsupervised scenarios, which does not rely on label information in this paper. Figure 1 represents a categorization of different anomaly detection approaches in evolving data. We also present a real application as an example of anomaly detection in evolving data. We conclude the paper with providing interesting future research directions.

2. ANOMALY DETECTION IN DATA STREAMS

---

1 In this paper, we use the terms outlier detection and anomaly detection interchangeably.
In this section we first start with a definition of data streams and then we survey the literature in the area of anomaly detection in data streams and different techniques.

### 2.1 Data streams

A data stream is a sequence of data points with three main characteristics. A data stream has a continuous flow. Hence, the processing time of the algorithm is a challenge. In addition, the volume of data delivered by a stream continually increases. In other words, the number of incoming data points is unbounded. Therefore, memory storage is another challenge. Finally, data streams can change over time. The solution to deal with these data stream challenges is to limit the number of processed data points, called data windowing. We categorize windowing techniques into four groups [67; 18]. Figure 2 shows this categorization.

1. **Landmark windows**: In this windowing technique, a fixed point in the data stream is defined as a landmark and processing is done over data points between the landmark and the present data point.

2. **Sliding windows**: A sliding window size $w$ is considered in this technique. It processes the last $w$ data points in the stream.

3. **Damped window model**: In sliding window technique, only the last $w$ data points are processed and the previous data points are ignored completely. Whereas in damped window model, a weight is assigned to each data point in such a way that the old data points get smaller weights. Therefore, the most recent data points are used with higher weights.

4. **Adaptive windows**: In all previous models, the number of data points that are being processed are fixed. In adaptive techniques, the window size $w$ would change as the data stream evolves. In this technique, the more the data points evolves, $w$ becomes smaller. In contrast, if data points remain constant, the value of $w$ increases.

### 2.2 Categorisations and techniques

We can categorize the outlier detection techniques in data stream and in unsupervised scenarios to three main groups: statistical based, clustering based and nearest neighbor based approaches. In the following subsections we give an overview of the techniques in each group along with their properties.

#### 2.3 Statistical Based Techniques

In statistical based approaches, the aim is to learn a statistical model for a normal behavior of a dataset. Thereafter, the observations that are not (or has low probability to) fit into that model is declared as outliers. In terms of data streams, Yaminshi et. al. [62; 61] assign an outlierness score to the already built Gaussian mixture model, which shows how the incoming data point is deviated from the model. However, in these set of approaches, *a priori* knowledge regarding the underlying distribution of the dataset is required, which is almost always unavailable when data evolves over time.

#### 2.4 Clustering Based Techniques

Clustering data streams has become an interesting topic in recent years and has drawn the attention of many researchers. Clustering algorithms can be used to find anomalies in data streams. They can be categorised into two groups: 1) One group of techniques are proposed in which the anomalies are assumed to fall into the clusters with small number of data points or low density. 2) In the second group of clustering based methods the distance of data points to their nearest cluster centroids are used to detect anomalies. In the following paragraphs we discuss the techniques in each group separately.

Many algorithms have been proposed in the first group. In 1996, a clustering algorithm called BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) was proposed for large databases [66]. It considers the fact that memory can not support large databases and is much smaller than the number of data points. The method uses the landmark window model and clusters the data points incrementally and dynamically in a single scan of the data. In ad-
dition, BIRCH is the first algorithm that handles noise and detects outliers in the data stream. It works based on a concept called a Cluster Feature (CF). It is a triplet consisting of three values, that provides useful information to describe a cluster. In other words, it is a way of summarizing a cluster. A Cluster Feature is defined on the following three values:

\[ CF = \langle n, LS, SS \rangle \]

where \( n \) is the number of \( D \)-dimensional data points in a cluster, \( LS \) is the linear sum of \( n \) data points and \( SS \) is the squared sum of \( n \) data points.

The algorithm starts to create clusters and stores only the \( CF \) value for each cluster, which is more memory efficient. The \( CFs \) are stored in a height-balanced tree (B+ tree). The process of building a tree is a top-down approach and this algorithm is considered as a hierarchical clustering. For each incoming data point BIRCH finds the point’s appropriate location in the leaf, then it updates the \( CF \) values and finally the path to the root of the tree should be updated. BIRCH scans the \( CF \) tree regularly and stores low density leaves as outliers.

In [7], for the first time a clustering algorithm called ClusStream was proposed which clusters evolving data streams. This algorithm also uses the \( CF \) concept which was used in [66], but extends it by adding the temporal characteristics of clusters. Therefore, for each cluster, it considers the relevant time stamps and stores the sum of relevant time stamps in addition to the squared sum of time stamps along with the triplet \( CF \) which has been defined above. This algorithm comprises of two steps: online and offline. In the online step the micro-clusters are created, while in the offline step macro-clusters are generated by applying a k-means algorithm on micro-clusters. For each incoming data point the micro clusters are updated accordingly: 1. The new data point can be assigned to one of the micro-clusters, or 2. The new data point is far from the existing clusters, therefore a new micro-cluster is created and one of the previous micro-clusters is identified as an outlier micro-cluster and is deleted (based on micro-cluster points’ time stamps) or merged with another micro-cluster. Hence, outliers are not identified by the size of the micro-clusters, rather they are identified by not being active in the recent stream activity.

Many algorithms have been proposed to improve ClusStream in terms of quality of clusters for evolving data streams. DenStream [20] specifically focuses on building arbitrary shaped clusters. Therefore, it uses a density-based clustering approach (DBSCAN [29]) to cluster micro-clusters. This algorithm is applicable in noisy datasets and can build normal micro-clusters as well as outlier micro-clusters similar to [7]. It uses a damped window model and weights the recent data points with higher values. Later on, a grid-based density approach is proposed based on two online and offline steps, called DStream [24]. In the online step each data point is assigned to a grid cell and low density grid cells are identified as outliers and removed. Here low density grid cells refer to either the grid cells with low number of data points or the grid cells with high number of data points while their effect have been reduced by a decay factor. In the offline step grids are clustered using a density-based clustering algorithm. This approach is time and memory efficient and clusters arbitrary shaped clusters. While [20] and [24] are based on a damped window model, SDstream [54] is proposed based on a sliding window model. Similar to previous approaches it uses a two step approach, except that it only keeps the last \( w \) data points.

While in all of the above approaches outliers are assumed to fall into the clusters with small number of data points or low density, in the remainder of this subsection we survey the papers in which the distance of data points to their nearest cluster centroids are used to detect anomalies in data streams. In [13], a new outlier detection algorithm called AnyOut is proposed to detect anytime outliers in streaming data. In anytime outlier detection, data streams have varying arrival rates and at a given time \( t \), outlierness value for the current data point \( p_t \) should be computed until the arrival of next data point. AnyOut builds a specific tree structure called ClusTree [41] suitable for anytime clustering. In this tree structure, \( CFs \) are stored in each node as well as a buffer for anytime insertion of clusters. Anyout investigates the Clustree in a top-down fashion and finds the closest \( CF \) to \( p_t \) at each level. Once the next data point \( p_{t+1} \) arrives, an outlierness score is computed as the distance of \( p_t \) to its nearest cluster feature centroid.

In [28], the data stream is divided into data chunks and \( k \)-means is used for each data chunk. Then the data points which are far enough from clusters centroids are identified as ‘candidate’ outliers. Hence, the outlierness value of data points might change depending on the incoming data chunks. Moshtaghi et. al. proposed a hyperellipsoidal clustering approach to model the normal behavior of the system, where the points outside this clustering boundary will be detected as anomalies [46]. It incrementally updates the hyperellipsoids’ parameters (mean and covariance matrix) to keep track of changes in the underlying distributions of data streams. Another algorithm that falls into the second category is eTSAAD [47], a new clustering algorithm that models data streams with a set of elliptical fuzzy rules. Similar to the previous approach the parameters of fuzzy clustering model are updated for each incoming data point to detect outliers and regime changes in data streams.

Rather than modeling data streams by only one set of cluster profiles and updating it over time, in [58] an ensemble-based algorithm has been proposed for evolving data streams in which a pool of clustering models are generated. Thereafter, for each incoming data point an outlierness values is being calculated based on a consensus decision using only the relevant set of clustering models. Similar to [46], a hyperellipsoidal clustering approach is used to model the normal behavior of the data streams in each data chunk and the distance between clusters’ centroids are being used for detecting anomalies. A more time/memory efficient anomaly detection algorithm has been proposed in [25], where the notion of active clusters are introduced. The input data streams is divided into different data chunks and for each incoming data chunk active clusters in the current data chunk are identified. Also, if there exists any emerging distributions, the model which consists of a set of hyperellipsoidal clusters are updated as well. Finally, while all of the above algorithms detect anomalies by modeling data streams based on clustering, none of them can identify anomalies in real-time. [26] proposes a new approach for discovering temporal evolution of clusters while detecting anomalies in real-time.

### 2.5 Nearest Neighbors Based Techniques
In clustering based approaches, the outlierness of an observation is evaluated considering the clustering models that are built. However, there is another group of outlier detection techniques in which outlierness scores are assigned to data points considering their nearest neighbors which lead to higher resolution analysis. This group of approaches have gained popularity as they are easy to implement and intuitive. The first trace of distance based outliers can be found in database community. Let \( P \) a dataset with \( n \) data points. According to the definitions given in [36; 38; 37], a data point \( p \in P \) is a distance-based outlier, DB-outlier, if and only if at least a fraction \( \epsilon \) of the whole number of data points are in a distance larger than radius \( r \). More specifically, a data point \( p \) is a DB-outlier if and only if:

\[
|\{q \in P|d(p, q) > r\}| \geq \epsilon n
\] (2)

where \( d \) is a distance function between two data points. These outliers are also called global outliers, as they consider all \( n \) existing data points to define outlierness for each data point.

While the output of the above definition is a binary decision on the outlierness of each data point, in [51], a score of distance-based outlierness has been proposed based on a given parameter \( k \) (a number of nearest neighbors). For a data point \( p \in P \), this score is called \( k\)-distance \((p) \) and it is computed as the distance between a data point \( p \) and its \( k \)-th nearest neighbor \((k\text{-NN}) \) in \( P \).

Later, Anguilli and Pizzuti defined another score of distance-based outlierness [11]. Given parameter \( k \), for each data point the distance between the data point and its all \( k \) nearest neighbors \((k\text{-NN}) \) are computed and considered as the outlierness score, specifically:

\[
k\text{-distance} - w(p) = \sum_{q \in N(p, k)} d(p, q)
\] (3)

where \( N(p, k) \) is the set of \( k \) nearest neighbors of \( p \).

### 2.5.1 Distance Based Techniques

In term of streaming data in [10] and [63], a sliding window is used to detect global distance-based outliers in data streams with respect to the current window. The authors in [39] improved the time complexity and memory consumption in comparison to [10] and [63]. Finally in [21], a faster and more general framework called LEAP has been proposed for high volume high dimensional data streams by optimizing the search space. In addition, different types of distance-based outliers [51], [11] have been detected by this framework. Since the outliers are computed with respect to the last \( n \) data points, the outliers that are detected by these approaches are called ‘global’ outliers. Hence these approaches fail to detect outliers in non-homogeneous densities.

### 2.5.2 Density Based Techniques

In contrast to distance-based ‘global’ outliers, distance-based ‘local’ outliers (a.k.a. density-based outliers) are data points that are outliers with respect to their \( k \) nearest neighbors. LOF is a well-known density-based outlier detection algorithm [19]. In this algorithm a score of outlierness, called \( \text{LOF} \) (Local Outlier Factor), is computed for a data point \( p \) according to the following definitions, assuming that all data points are available and the number of nearest neighbors is \( k \).

- **\( k\)-distance\( (p) \):** the distance between a data point \( p \) and its \( k\)-th nearest neighbor \((k\text{-NN}) \).

- **Reachability distance (reach-dist) of a data point \( p \) with respect to another data point \( o \):**

\[
\text{reach-dist}_k(p, o) = \max \{k\text{-distance}(o), d(p, o)\}
\] (4)

where \( d(p, o) \) is the Euclidean distance between \( p \) and \( o \).

- **Local reachability density (lrд) of a data point \( p \):**

\[
lrd_k(p) = \left(\frac{1}{k} \sum_{o \in N(p, k)} \text{reach-dist}_k(p, o)\right)^{-1}
\] (5)

where \( N(p, k) \) is the set of \( k \) nearest neighbors of \( p \).

- **Local outlier factor of a data point \( p \):**

\[
\text{LOF}_k(p) = \frac{1}{k} \sum_{o \in N(p, k)} \frac{lrd_k(o)}{lrd_k(p)}
\] (6)

Since the LOF technique achieves good detection accuracy in non-homogeneous densities (often the case in practice) without assumptions regarding the underlying distribution of the dataset, it has become a popular approach and many variants of this technique have been proposed.

Pokrajac et al. [50] were the first to propose an incremental version of the LOF technique (iLOF) that can be used for data streams. All previous versions required the entire dataset to compute LOF values for the data points, but in the iLOF technique the outlier factor is computed for each incoming data point. For each incoming data point \( p \), the iLOF finds the \( k \)-nearest neighbors \((k\text{-NNs}) \) of \( p \), computes the local outlier factor of \( p \) based on the outlier factors of its \( k\text{-NNs} \), and updates the \( k\text{-NNs} \) of past data points along with their local outlier factors if needed. Pokrajac et al. showed that only a few data points are required updating and therefore the algorithm was efficient. In order to detect outliers accurately, all past data points need to be retained to compute the outlier factor for each new incoming data point. Therefore, iLOF suffers from large memory requirements, as well as high time complexity.

Recently a memory efficient algorithm called MiLOF has been proposed to address these limitations in density-based outlier detection problems [57]. MiLOF detects outliers in data streams with the same technique as iLOF but with less memory/time consumption with a comparable accuracy to iLOF.

### 3. ANOMALY DETECTION IN EVOLVING GRAPHS

In this section, we discuss the problem of detecting anomalies in dynamic graphs. The graphs that we consider in this section are snapshots of evolving graphs. Therefore we can consider these graph snapshots to be static entities. Our objective is to detect anomalous behaviour in the graph, which might involve abnormal vertices, edges and subgraphs. This problem is quite similar to anomaly detection in clouds of data points.
Popular anomaly detection techniques can be employed in graph datasets as well. We can extract high level features from the graph and perform anomaly detection techniques on the processed data. The snapshots of the graph can be considered as potential high dimensional data, i.e., after feature extraction, thus we can consider evolving graphs as data streams and deal with the problem of anomaly detection in this type of data using the techniques described in Section 2.

Due to the popularity of graphs in representing real-world networks, numerous methodologies have been developed for spotting anomalies in graph data. We discuss anomaly detection in graph data only for plain graphs where there are only nodes and edges representing the data, and there are no attributes associated with those nodes and edges. However these techniques can be extended to attributed graphs as well. Nodes and/or edges in an attributed graph represent various features. For instance, a node in a social network may have various education levels or interests, and links may have different strengths.

We first need to define the anomaly detection problem given the snapshots of graphs over time. Anomalies are defined as nodes, edges and sub-patterns that deviate from the observed normal patterns in the graphs. Since we are considering plain graphs, the significant information to incorporate in an anomaly detection scheme is the graph structure or topology. The interesting patterns can be detected through two sets of techniques: structural sub-patterns and community-based sub-patterns.

The first category of techniques extracts useful graph centric information such as node degree, in addition to other features, and performs anomaly detection on data points. Therefore, the problem of anomaly detection in graphs is transformed to the well-known problem of spotting outliers in an n-dimensional space.

The graph centric features can be computed from nodes or the combination of two, three or more nodes, i.e., dyads, triads and communities. These features can also be extracted from the combination of all nodes in a more general manner [35]. Network intrusion detection [27] and spotting web ads and communities. These features can also be extracted.

3.1 Node-based Anomalies

One of the anomalous objects that can be found in a dynamic network is a single vertex or collection of vertices. In each time step, a series of features are extracted from the nodes. The techniques in this category constitute a feature extraction scheme, which measures vertex properties such as the degree, or the ego-net density of a single vertex. The nodes that demonstrate irregular behaviour in comparison to others are detected as anomalous. However it is worth noting that various approaches use a number of different feature extraction techniques based on the assumptions, aim and the domain of the problem they are interested in.

Since we are considering streaming graphs, the context of time is also added to the features extracted from a node. The temporal aspect of the graphs adds another level of complexity to the problem. We define node anomalies as the vertices whose scores are above the average score $\hat{f}$ as shown in Equation 7. The score is calculated using a function $f$ that yields a real-valued score for each node, $f : V \rightarrow \mathbb{R}$.

The average score $\hat{f}$ is calculated based on the scores of the normal nodes. The anomalous nodes $V' \subset V$ are thus defined as:

$$\forall v' \in V', |f(v') - \hat{f}| > c_0$$  \hspace{1cm} (7)

where $c_0$ is the acceptable deviation calculated from the normal node behaviour. It is worth mentioning that the scoring function $f$ may measure the change in the number of connections that each vertex has between consecutive time steps, or the change in the edge weights. In simple scenarios, anomalous nodes undergo substantial change, which makes their detection easier. However, this is not always the case, for instance, consider the example of a node anomaly shown in Figure 3. The two time stamps $t_1$ and $t_2$ are consecutive snapshots of the same network. As you can see, there are two full cliques consisting of nodes $V_1..5$ and $V_6..10$. These two full cliques can be considered as communities, such as the students in two different high schools. At time stamp $t_1$, there is a link between two of the nodes/students from different communities, namely nodes $V_2$ and $V_6$. However, in the next time stamp, this link is eliminated. Although the degrees of the nodes $V_2$ and $V_6$ have not changed substantially, time stamp $t_2$ is considered to be abnormal and the two previously mentioned nodes are detected as anomalous. The interesting insight from this example is that detecting abnormal nodes needs a structure-aware feature extraction technique to first spot the discontinuity.

![Graph variants](image)

Figure 3: Graph variants where circles and lines represent vertices and edges respectively.

A good technique in detecting anomalous nodes in a community structured network is to determine the level of involvement of a node in communities throughout time. An example of such an approach is [55], where the authors try to model the roles for each node and monitor their role change over time. Their definition of node roles is dissimilar to that of community memberships. The roles in [55] are defined as sets of nodes that are structurally similar to each other, and not necessarily the nodes with many connection within themselves.

They propose a Dynamic Behavioural Mixed-membership Model (DBMMM), which uses the concept of feature based roles and can generalize the learned roles to the unobserved...
vertices. DBMM focuses on temporal anomalies, where at each time step, a feature extraction procedure takes place that determines node roles. They use regularization to compute the number of roles and also employ transition matrices to calculate the probability of a node changing its role in the upcoming time-step. This approach is consistent with the definition of a node anomaly in Equation 7 in the sense that the node anomaly score is calculated based on the difference of its estimated and true roles. It is worth noting that DBMM can only detect abnormal nodes in each time step and it lacks a more general outcome, i.e., anomalous graph instances.

Another example of node-based anomaly detection in streaming graphs is [33], where the authors introduce the notion of Evolutionary Community Outliers (ECOOutlier). ECOOutlier refers to the task of detecting vertices that indicate abnormal community-membership evolution in consecutive time-stamps. The authors first define a normal evolutionary trend based on the latent communities discovered in the data. They propose an approach for integrating community matching and anomaly detection, where the problem becomes one of minimizing community matching error while anomalous nodes are ignored by assigning lower weights to them.

One of the properties of ECOOutlier [33] is that in each iteration, only the two consecutive time-stamps are compared. This method constructs a belongingness matrix, which is the basis for community matching and anomaly detection. The outcome of this approach at each time-stamp is the anomaly score of each node in the graph.

### 3.2 Edge-based Anomalies

Another anomalous entity that can be found in a dynamic network is the edge or collection of edges that indicate anomalous evolutionary trends in comparison to the majority of the edges in the network. In contrast to node-based anomalies, the abnormal edges can be directly detected from the edge weight evolution or edge addition/elimination between nodes, which do not belong to the same community or are very unlikely to form a relationship.

We can define the edge anomalies similar to the vertex anomalies that were described in Equation 7. The edges whose scores are more than the average score $\tilde{f}$ as shown in Equation 8 are considered anomalous. The edge score is calculated using a score function $f$ that yields a real-valued score for each edge, $f: E \to \mathbb{R}$. The average score $\tilde{f}$ is calculated based on the scores of the normal edges. The anomalous edges $V' \subset V$ are thus defined as:

$$\forall e' \in E', |f(e') - \tilde{f}| > c_0$$  

(8)

where $c_0$ is the acceptable deviation from the average normal edge score. A trivial example of the edge score function $f$ is to measure the weight change of an edge from one time-stamp to the next. In simple scenarios, anomalous nodes undergo substantial change, which makes their detection easier. This is shown in Figure 4, where the edge connecting nodes $V_3$ and $V_4$ is indicating abnormal evolution in the second time-stamp. Another anomalous edge weight evolution is captured in the edge connecting nodes $V_1$ and $V_4$ at the third time-stamp.

In addition to the edge weight evolution, we can model the edges based on their corresponding nodes’ interactions. The score function of each edge $f$ in this context is the probability of an edge weight at a given time-stamp. An example of such an approach is in [45], where the authors assign an anomaly score to every edge with respect to the probability of observing that edge weight at a given time-stamp. This method is called NetSpot, which uses the edge anomaly score to detect significant anomalous regions, i.e., subgraphs, within a dynamic graph. The outcome of the NetSpot approach [45] is a collection of anomalous subgraphs and their corresponding time windows.

![Figure 4](image_url)

**Figure 4:** An illustration of an anomalous edge that occurs due to an irregular weight evolution pattern, with the anomalous edge highlighted in red. At each time stamp, a vertex weight typically changes by ±0.05 at most. However, edge $(1, 3)$ has a spike in its weight at time stamp $2$, unlike any other time in the series.

### 3.3 Events or Changes

Another type of anomalies in dynamic graphs is changes or events. These are the most important categories of anomalies in networks since they can often be associated with important phenomenon in the real world. Unlike the node- and edge-based anomalies, this type of anomalous behaviour can only be found in dynamic networks. Event detection can be defined as time stamps when the graph structure is different from that of the other time stamps. This definition is much broader than the previously defined types of anomalies and contains edge or node anomalies as well. To address the problem of event detection, we need to measure the graph score for each time stamp in the dynamic network. Examples of graph scores are average clustering coefficient, average vertex degree, and so on. Each graph can be summarized through this scoring function $f$. We can define an event as a time stamp when the current graph score $f(G_t)$ is different from the previous and next graph scores, i.e., $f(G_{t-1})$ and $f(G_{t+1})$, as shown in Equation 9. The graph score is calculated using a score function $f$ that yields a real-valued score for each graph, $f: G_t \to \mathbb{R}$. The average score $\tilde{f}$ is calculated based on the scores of the normal edges. An anomalous time stamp $t$ is thus defined as:

$$|f(G_t) - f(G_{t-1})| > c_0 \quad \text{and} \quad |f(G_t) - f(G_{t+1})| > c_0$$  

(9)

where $c_0$ is the acceptable deviation from the average normal graph score. A simple example of a graph score is to calculate the number of edges or vertices at each time stamp and use it as a basis of comparison between two graphs. Note that the task of event detection in dynamic networks merely identifies an anomalous time-stamp. It does not provide any attribution to the underlying cause of anomalous behaviour. Another type of anomaly detection in dynamic networks is change detection, which is closely related to events. While events occur suddenly and in an isolated manner, changes indicate the time-stamps, where the graph structure changes.
and this change is maintained afterwards until another change point is encountered. Figure 5 shows an example of a change point in a dynamic network. At the fourth time-stamp, the structure of the network has changed to a semi-full clique and it remains the same afterwards.

Figure 5: Example of a time sequence of graphs that contains a change point at $t_4$.

One of the approaches for event detection in dynamic networks is [49]. The authors in [49] introduce a technique called ParCube, which uses tensor decomposition to efficiently represent the data using sparser latent factors. ParCube is a parallelisable approach for fast tensor decomposition on large-scale datasets. Since the data is stored in a tensor, ParCube can manage attributed graphs as well as plain graphs. The authors in ParCube employ a random sampling strategy to select slices of the original tensor and thereafter perform tensor decomposition on these more manageable slices. The random sampling and decomposition on different slices can be done simultaneously. Then they merge the smaller decompositions and combine them to retrieve the overall tensor decomposition outcome. ParCube is used for event detection by computing the reconstruction error of the tensor decomposition from one time-stamp to another. In addition to ParCube, there are other approaches that use tensor decomposition as the basis of event detection. For instance, the authors in [12] introduce a technique called Com2, which detects community-based anomalies in a dynamic network. The intuition behind this approach is to determine comets or communities that appear and disappear periodically. The authors use low-rank tensor factorization in combination with Minimum Description Length (MDL) to detect communities in a temporal environment.

Another event detection technique is introduced in [17], where the authors propose an approach called NetSimile, which extracts structural features from each graph. These features comprise the signature vector for each graph in the dynamic network setting. The problem of finding the graph similarity between two graphs is reduced to finding the distance of their corresponding signature vectors. NetSimile uses the Canberra distance between the pairs of signature vectors. The features that NetSimile captures include ego-net properties, node degree, clustering coefficient and so on.

Finally, the last techniques that we discuss in this category are [40; 52; 53], which were introduced recently. All of these approaches compare consecutive graphs based on their structural features. The authors in [40] introduce an approach called DeltaConnectivity (DeltaCon), where the underlying intuition is to calculate node affinities, i.e., the influence that nodes have on each other, by using Fast Belief propagation. DeltaCon uses the pairwise node affinities to measure graph similarity. The authors in [52] propose a scalable technique that creates a compact yet structure-aware feature set for each graph using a matrix permutation technique called Amplay. The matrix permutation step can be used as a heuristic for determining the Maximal Independent Sets (MISs) in a graph. The resulting feature set includes the rank of each node in a graph and this rank ordering is used by rank correlation for comparing a pair of graphs. Finally, [53] proposes a structure aware graph embedding scheme based on random projection and exploits the Johnson and Lindenstrauss lemma to provide a theoretical proof of its performance. They capture features pertaining to community structure or topological proximity of nodes in a graph.

Table 1 provides a comparison between dynamic graph anomaly detection schemes.

4. APPLICATION: FOREST FIRE RISK PREDICTION

In this section we first review a few methods currently used for forest fire danger assessment and then we show how anomaly detection techniques can be used to predict the risk of forest fire.

Natural disasters are a prevalent reality around the world. Roughly 102 million people worldwide were affected by natural disasters in 2014 alone [4], with a global annual economic loss estimated at over $300B [5]. Researchers link natural disasters with climate change and statistics show that in 2014, 87% of worldwide natural disasters were climate-related [4]. Some countries, including Australia, experience forest fires, locally known as bushfires, as the most damaging disasters. In the Australian state of Victoria, forest fires pose the largest annual risk to the safety of residents [3]. For example, in the recent ‘Black Saturday’ Victorian forest fire event of February 2009, over 1.1 million acres burnt, 173 people lost their lives and over 400 people were injured [2]. Another notable impactful event was ‘Ash Wednesday’ in February 1983, with 75 fatalities and over 1 million acres burnt in the states of Victoria and South Australia. The
impact of these disasters is multi-dimensional, from social and psychological to economic and environmental. The reality described above makes a strong case for forest fire research, not only in Australia but also around the world. One of the most pressing problems in this context is the ability to predict the risk of a forest fire event. By knowing the risk, government agencies, communities and individuals can be better informed so they can take the most appropriate measures to mitigate and prepare for forest fire events, if and when they eventuate.

In Australia, the McArthur Forest Fire Danger Index (FFDI) [43] is used by the national Bureau of Meteorology to draw Fire Danger Index maps [1], which in turn are used by state forest fire authorities to determine Fire Danger Ratings (FDR). FFDI is calculated based on temperature, humidity, wind speed, dryness and fuel weight as shown in Equations 10 and 11 [48]:

\[
FFDI_\gamma = 2e^{-0.45+0.987\ln(DF_\gamma)-0.0345H_\gamma+0.0338T_\gamma+0.0234V_\gamma}
\]

(10)

\[
DF_\gamma = \frac{0.191(I_\gamma + 104)(N_\gamma + 1)^{1.5}}{3.52(N_\gamma + 1)^{1.5} + Pre_{\gamma}} - 1
\]

(11)

where \(H_\gamma\) is relative humidity, \(T_\gamma\) is air temperature, \(V_\gamma\) is average wind velocity in the open at the height of 10 m, \(FFDI_\gamma\) is forest fire danger index and \(DF_\gamma\) is drought factor which uses precipitation observations \(Pre_{\gamma}\), \(N_\gamma\) is time since rain and \(I_\gamma\) is soil’s moisture content all at time \(\gamma\).

The output FFDI values are further interpreted based on a defined categorisation. Figure 6 depicts the six different categories. Using the above equations and these categories, the Australian emergency management agencies find the relevant category and broadcast it daily.

![Figure 6: Forest fire Danger Categories in Australia](image)

The main drawback of operational forest fire rating systems including FFDI is that the danger index is calculated based on the most current weather observations available, using a static model, and not customised to the particular location where the danger index is used. This determines a coarse variation between the danger indices of consecutive days and is not indicative of the temporal development of the forest fire danger conditions, which can be observed prior to forest fire events.

Along with FFDI, other approaches to predict forest fire danger also exist around the world. There is a vast body of work modelling the risk of forest fires with diverse focus. Some investigate the likelihood of forest fires, while others study the intensity or effects of forest fires (based on ecological, social or economic values) [44]. To counter the issue of spatially limited data (e.g. received from sparse weather stations), a number of studies look at forest fire danger detection using sensor networks. For example, [64] propose a wireless sensor network where sensor nodes are used to collect the data (e.g. temperature, relative humidity) and submit the information to specific cluster nodes. The cluster nodes will then construct neural networks to produce a weather index showing the likelihood for the weather to cause a forest fire. The paper does not describe the actual model hence the spatiotemporal customisation of the model cannot be assessed. In [31], it is claimed that forest fire risk needs to consider both forest fire behaviour probability and effect. The former depends on the spatial and temporal factors controlling forest fire spread, including fuel and weather. Moreover, the authors investigate the burn probability based on historic forest fire data, and highlight the need to move beyond assumptions of spatial and temporal uniformity while modelling that probability. Their approach is in line with [30] that argues that localised spatial properties (topography, fuel, weather) produce local differences in the forest fire behaviour, hence local differences in forest fire risk.

### 4.1 Anomaly detection based approach

In this subsection we argue how anomaly detection can be used for forest fire risk prediction. The input data for most of the forest fire danger prediction methods are meteorological observations sourced from weather stations (e.g. temperature, wind speed, wind gust, wind direction relative humidity, precipitations) together with spatial features such

<table>
<thead>
<tr>
<th>Method</th>
<th>Graph Properties</th>
<th>Type</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[45]</td>
<td>✓</td>
<td></td>
<td>Subgraph-Region</td>
</tr>
<tr>
<td>[33]</td>
<td>✓</td>
<td></td>
<td>Community Membership</td>
</tr>
<tr>
<td>[12]</td>
<td>✓</td>
<td></td>
<td>Change</td>
</tr>
<tr>
<td>[55]</td>
<td>✓</td>
<td></td>
<td>Node-Role Membership</td>
</tr>
<tr>
<td>[46]</td>
<td>✓</td>
<td></td>
<td>Change</td>
</tr>
<tr>
<td>[47]</td>
<td>✓</td>
<td></td>
<td>Change</td>
</tr>
<tr>
<td>[48]</td>
<td>✓</td>
<td></td>
<td>Change</td>
</tr>
</tbody>
</table>

Table 1: Comparison of dynamic graph anomaly detection schemes. The graph properties are summarized in the first four columns, the type column corresponds to the category of anomaly detection, i.e., whether the method detects node, edge or subgraph anomaly. The last column denotes the time complexity of each method where high complexity means nonlinear in regards to the number of edges.
as topography, vegetation type and vegetation density to predict the forest fire danger/risk for the given conditions based on empirical methods. The meteorological observations are examples of evolving data streams. Any of the anomaly detection techniques described in Section 2 can be used to detect forest fire risk. Note here extreme weather conditions can be seen as high risk episodes. In [59] a similar approach is used and it is shown that leveraging anomaly detection approaches is beneficial in predicting risk of forest fire in 3 weather stations in Australia, and as a result giving the decision makers more fine-grained information about the increasing forest fire danger and allowing them to take even better informed decision to protect communities and the environment.

If we monitor meteorological features in more than one location then we can consider both meteorological as well as spatial features along with their relationships. Hence we can represent this dataset as an evolving graph. Anomaly detection techniques described in Section 3 can be potentially used to model this problem and predict the risk of forest fire.

5. CONCLUSIONS

While there exists a considerable number of literature surveys capturing a variety of techniques in detecting anomalies in static data [22; 9], anomaly detection in the presence of evolving data has not been well explored. In this paper, we have considered the problem of detecting anomalies in evolving data and reviewed the existing ‘unsupervised’ anomaly detection techniques in this domain. We first surveyed techniques proposed for data streams and then focused on more complex scenarios, i.e., evolving graphs. We have also showcased the importance of anomaly detection in dynamic settings through a real-world application example, i.e., forest fire risk prediction.

5.1 Future Work

One of the emerging directions of research in detecting anomalies is their implications in high dimensional data streams or attributed dynamic graphs. Although there have been limited attempts by [60; 65; 32] in data streams, most of the existing algorithms in this survey lose their effectiveness in the presence of high dimensional data. Thus, we need to redesign the current models to be able to detect outlying patterns accurately and efficiently. More specifically, when there is a large number of features, there might exist a set of anomalies that emerge in only a subset of dimensions at a particular period of time. This set of anomalies may appear normal with regards to a different subset of dimensions and/or period of time. Another interesting future research direction is the appearance/disappearance of new/old data dimensions over time. This is an interesting area with many potential applications such as anomaly detection in IoT devices, where the sensors (representing the number of dimensions) can go off/on intermittently over time.

Another emerging area is ensemble analysis. Ensemble analysis is a technique that has been shown to improve the accuracy of many data mining tasks including anomaly detection in static data. Initial attempts on ensemble analysis for anomaly detection in streaming setting are described in [58; 32]. However, this area of research is still unexplored and more accurate models can be proposed based on the bias-variance trade-off.

6. REFERENCES


ABSTRACT
Looking from a global perspective, the landscape of online social networks is highly fragmented. A large number of online social networks have appeared, which can provide users with various types of services. Generally, information available in these online social networks is of diverse categories, which can be represented as heterogeneous social networks (HSNs) formally. Meanwhile, in such an age of online social media, users usually participate in multiple online social networks simultaneously, who can act as the anchors aligning different social networks together. So multiple HSNs not only represent information in each social network, but also fuse information from multiple networks.

Formally, the online social networks sharing common users are named as the aligned social networks, and these shared users are called the anchor users. The heterogeneous information generated by users’ social activities in the multiple aligned social networks provides social network practitioners and researchers with the opportunities to study individual user’s social behaviors across multiple social platforms simultaneously. This paper presents a comprehensive survey about the latest research works on multiple aligned information sources, and evaluates the corresponding benefits. Great challenges exist in broad learning for the effective fusion of relevant knowledge across different aligned information sources, which depends upon not only the relatedness of these information sources, but also the target application problems to be studied. Broad learning aims at developing general methodologies, which will be shown to work for a diverse set of applications, while the specific parameter settings can be learned for each application from the training data.

In this paper, we will focus on introducing the broad learning research works done based on online social media data. Nowadays, to enjoy more social network services, people are usually involved in multiple online social networks simultaneously, such as Facebook, Twitter and Foursquare [104; 32]. Individuals usually have multiple separate accounts in different social networks, and discovering the correspondence between accounts of the same user (i.e., network alignment or user anchoring) [98; 99; 32; 91; 96; 84] will be an interesting problem. What’s more, network alignment is also the crucial prerequisite step for many interesting inter-network synergistic knowledge discovery applications, like (1) inter-network link prediction/recommendation [94; 104; 86; 87; 96; 84; 25; 100; 88], (2) mutual community detection [95; 26; 97; 57; 85; 101], (3) cross-platform information diffusion [78; 77; 103], and (4) multiple networks synergistic embedding [83; 93]. These application tasks are fundamental problems in social network studies, which together with the network alignment problem will form the backbone of the multiple social network broad learning ecosystem.

This paper will cover five strongly correlated social network research problems based on broad learning:

- **Network Alignment**: Identifying the common users shared by different online social networks can effectively combine these networks together, which will also provide the opportunity to study users’ social behaviors from a more comprehensive perspective. Many research works have proposed to align the online social networks together by inferring the mappings of the shared users between different networks, which will be introduced in a great detail in this paper.

- **Link Prediction**: Users’ friendship connections in different networks have strong correlations. With the social activity data across multiple aligned social networks, we can acquire more knowledge about users and their personal social prefer-
ences. We will introduce the existing research works on the social link prediction problem across multiple aligned social sites simultaneously.

- **Community Detection**: Information available across multiple aligned social networks provides more complete signals revealing the social communities formed by people in the real world. We will introduce the existing works on community detection with knowledge fused from multiple aligned heterogeneous social networks as the third task.

- **Information Diffusion**: The formulation of multiple aligned heterogeneous social networks provides researchers with the opportunity to study the information diffusion process across different social sites. The latest research papers on cross-network information diffusion will be illustrated as well.

- **Network Embedding**: Information from other aligned networks can help refine the feature representations of users effectively. In recent years, some research papers introduce the synergistic network embedding across aligned social networks, where knowledge from other external networks can be effectively utilized in representation learning tasks.

The remainder parts of this paper will be organized as follows. We will first provide the basic terminology definitions in Section 2. Via the anchor links, we will introduce the inter-network meta path concept in Section 3, which will be used in the following sections. The network alignment research papers will be introduced in Section 4. Inter-network link prediction and friend recommendation will be talked about in Section 5. A detailed review about cross-network community detection will be provided in Section 6. Broad learning based information diffusion is introduced in Section 7 and network embedding works are available in Section 8. Finally, we will illustrate several potential future development directions about broad learning and conclude this paper in Section 9.

2. TERMINOLOGY DEFINITION

Online social networks (OSNs) denote the online platforms which allow people to build social connections with other people, who share similar personal or career interests, backgrounds, and real-life connections. Online social networking sites vary greatly and each category of online social networks can provide a specific type of featured services. To enjoy different kinds of social networks services simultaneously, users nowadays are usually involved in many of these online social networks aforementioned at the same time, in each of which they will form separate social connections and generate a large amount of social information.

Formally, the online social networks can be represented as graphs. Besides the users, there usually exist many other types of information entities, like posts, photos, videos and comments, generated by users' online social activities. Information entities in online social networks are extensively connected, and the connections among different types of nodes usually have different physical meanings.

To model the phenomenon that users are involved multiple networks, a new concept called “multiple aligned heterogeneous social networks” [104; 32] has been proposed in recent years.

For the networks with simple structures, like the heterogeneous networks merely involving users and friendship links, the social patterns in them are usually easy to study. However, for the networks with complex structures, like the heterogeneous networks, the nodes can be connected by different types of link, which will have totally different physical meanings. One general technique for heterogeneous network studies is “meta path” [65; 104], which specifically depicts certain link-sequence structures connecting node defined based on the network schema. The meta path concept can also be extended to the multiple aligned social network scenario as well, which can connect the node across different social networks. Given a network $G = (V, E)$, we can represent the set of node and link types involved in the network as sets $N$ and $R$ respectively. Based on such information, the social network concept can be formally defined based on the graph concept by adding the mappings indicating the node and link type information.

**DEFINITION 1.** (Social Networks): Formally, a heterogeneous social network can be represented as $G = (V, E, \phi, \psi)$, where $V$, $E$ are the sets of nodes and links in the network, and mappings $\phi : V \rightarrow N$, $\psi : E \rightarrow R$ project the nodes and links to their specific types respectively. In many cases, the mappings $\phi, \psi$ are omitted assuming that the node and link types are known by default.

In the following parts of this paper, depending on the categories of information involved in the online social networks, we propose to categorize the online social networks into three groups: homogeneous social networks, heterogeneous social networks and aligned heterogeneous social networks.

2.1 Homogeneous Social Network

**DEFINITION 2.** (Homogeneous Social Network): For an online social network $G$, if there exists one single type of nodes and links in the network (i.e., $|V| = |R| = 1$), then the network is called a homogeneous social network.

Given a homogeneous social network $G = (V, E)$ with user set $V$ and social relationship set $E$, depending on whether the links in $G$ are directed or undirected, the social link can denote either the follow links or friendship links among individuals. Given an individual user $u \in V$ in an undirected friendship social network, the set of users connected to $u$ can be represented as the friends of user $u$ in the network $G$, denoted as $\Gamma(u) \subseteq V = \{v|v \in V \wedge (u, v) \in E\}$. The number of friends that user $u$ has in the network is also called the degree of node $u$, i.e., $|\Gamma(u)|$.

Meanwhile, in a directed network $G$, the set individuals followed by $u$ (i.e., $\Gamma_{out}(u) = \{v|v \in V \wedge (v, u) \in E\}$) are identified as the set of followers of $u$ and the set of individuals that follow $u$ (i.e., $\Gamma_{in}(u) = \{v|v \in V \wedge (u, v) \in E\}$) are called the set of followers of $u$. The number of users who follow $u$ is called the in-degree of $u$, and the number of users followed by $u$ is called the out-degree of $u$ in the network. For the users with large out-degrees, they are called the hubs [31] in the network; while those with large in-degrees, they are called the authorities [31] in the network.

2.2 Heterogeneous Social Network

**DEFINITION 3.** (Heterogeneous Social Network): For an online social network $G$, if there exists multiple types of nodes or links in the network (i.e., $|N| > 1$, or $|R| > 1$), then the network is called a heterogeneous social network.

Most of the online networks in the real world may contain very complex information involving multiple types of nodes and connections. For instance, in the social networks to be studied in the following part, they may involve users, posts, check-ins, words and timestamps, as well as the friendship links, write links and contain links among these nodes. Formally, such an online social network can be defined as $G = (V, E)$, where $V$ denotes the set of nodes and $E$ represents the set of links in $G$. The node set $V$ can be divided into several subsets $V = \mathcal{U} \cup \mathcal{P} \cup \mathcal{L} \cup \mathcal{T} \cup \mathcal{W}$ involving
the user nodes, post nodes, location nodes, word nodes and timestamps respectively. The link set $E$ can be divided into several subsets as well, $E = E_{u,w} \cup E_{u,p} \cup E_{p,l} \cup E_{p,w} \cup E_{p,t}$, containing the links among users, the links between users and posts, and those connecting posts with location checkins, words, and timestamps.

In the heterogeneous social networks, each node can be connected with a set of nodes belonging to different categories via various type of connections. For example, given a user $u \in U$, the set of user node incident to $u$ via the friend links can be represented as the online friends of $u$, denoted as set $\{v | v \in U, (u, v) \in E_{u,u}\}$; the set of post node incident to $u$ via the write links can be represented as the posts written by $u$, denoted as set $\{w | w \in P, (u, w) \in E_{u,p}\}$. The location check-in nodes, word nodes and timestamp nodes are not directly connected to the user node, while via the post nodes, we can also obtain the set of locations/words/timestamps that are visited/used/active-at by user $u$ in the network. Such indirect connections can be described more clearly by the meta path concept more clearly in Section 3.

### 2.3 Aligned Heterogeneous Social Networks

**Definition 4.** (Multiple Aligned Heterogeneous Networks): Formally, the multiple aligned heterogeneous networks involving $n$ networks can be defined as $G = \{(G^{(1)}, G^{(2)}, \ldots, G^{(n)}) , (A^{(1)}, A^{(2)}, \ldots, A^{(n-1,n)})\}$, where $G^{(1)}, G^{(2)}, \ldots, G^{(n)}$ denote these $n$ heterogeneous social networks and the set $A^{(1)}, A^{(2)}, \ldots, A^{(n-1,n)}$ represent the undirected anchor links aligning these networks respectively. Anchor links actually refer to the mappings of information entities shared across different sources, which correspond to the same information entity in the real world, e.g., users in online social networks, authors in different bibliographic networks, and movies in the movie knowledge libraries.

**Definition 5.** (Anchor Link): Given two heterogeneous networks $G^{(1)}$ and $G^{(2)}$ which share some common information entities, the set of anchor links connecting $G^{(1)}$ and $G^{(2)}$ can be represented as set $A^{(1,2)} = \{(u^{(1)}_{m_1}, u^{(2)}_{n_1}) | u^{(1)}_{m_1} \in V^{(1)} \land u^{(2)}_{n_1} \in V^{(2)} \land u^{(1)}_{m_1} \neq u^{(2)}_{n_1}\}$ denote the same information entity).

The anchor links depict a transitive relationship among the information entities across different networks. Given $i$ information entities $u^{(1)}_{m_1}, u^{(2)}_{n_1}$ from networks $G^{(1)}, G^{(2)}$ respectively, if $u^{(1)}_{m_1}, u^{(2)}_{n_1}$ are connected by an anchor link and $u^{(1)}_{m_1}, u^{(2)}_{n_1}$ are connected by another anchor link, then the user pair $u^{(1)}_{m_1}, u^{(2)}_{n_1}$ will be connected by an anchor link by default. For more detailed definitions about other related terms, like anchor users, non-anchor users, full alignment, partial alignment and non-alignment, please refer to [104].

### 3. META PATH

To deal with the social networks, especially the heterogeneous social networks, a very useful technique is meta path [65, 104]. Meta path is a concept defined based on the network schema, outlining the connections among nodes belonging to different categories. For the nodes which are not directly connected, their relationships can be depicted with the meta path concept. In this part, we will define the meta path concept, and introduce a set of meta paths within and across real-world heterogeneous social networks respectively.

#### 3.1 Network Schema

Given a network $G = (V, E)$, we can define its network schema to describe the categories of nodes and links involved in $G$.

**Definition 6.** (Network Schema): Formally, the schema of network $G$ can be denoted as $S_G = (N, R)$, where $N$ and $R$ are the sets of node type and link type in the network respectively. Network schema provides a meta level description of networks. Meanwhile, if a network $G$ can be outlined by the network schema $S_G$, $G$ is also called a network instance of the network schema. For a given node $u \in V$, we can represent its corresponding node type as $\phi(u) = N \in N$, and call $u$ is an instance of node type $N$, which can also be denoted as $u \in N$ for simplicity. Similarly, for a link $(u, v)$, we can denote its link type as $\psi(u, v) = \mathbb{R} \in \mathbb{R}$, or $(u, v) \in \mathbb{R}$ for short. The inverse relation $R^{-1}$ denotes a new link type with reversed direction. Generally, $R$ is not equal to $R^{-1}$, unless $R$ is symmetric.

#### 3.2 Meta Path in Heterogeneous Social Networks

Meta path is a concept defined based on the network schema denoting the correlation of nodes based on the heterogeneous information (i.e., different types of nodes and links) in the networks.

**Definition 7.** (Meta Path): A meta path $P$ defined based on the network schema $S_G = (N, R)$ can be represented as $P = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k$, where $N_i \in N, i \in \{1, 2, \cdots, k\}$ and $R_i \in \mathbb{R}, i \in \{1, 2, \cdots, k-1\}$.

Furthermore, depending on the categories of node and link types involved in the meta path, we can specify the meta path concept into several more refined groups, like homogeneous meta path and heterogeneous meta path, or social meta path and other meta paths.

**Definition 8.** (Homogeneous/Heterogeneous Meta Path): Let $P = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k$. $N_i$ denote a meta path defined based on the network schema $S_G = (N, R)$. If all the node types and link types involved in $P$ are of the same category, $P$ is called a homogeneous meta path; otherwise, $P$ is called a heterogeneous meta path.

The meta paths can connect any kinds of node type pairs, and specifically, for the meta paths starting and ending with the user node types, those meta paths are called the social meta paths.

**Definition 9.** (Social Meta Path): Let $P = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k$. $N_i$ denote a meta path defined based on network schema $S_G = (N, R)$. If the starting and ending node types $N_1$ and $N_k$ are both the user node type, $P$ is called a social meta path.

Users are usually the focus in social network studies, and the social meta paths are frequently used in both research and real-world applications and services. The number of path segments in the meta path is called the meta path length. For instance, the length of meta path $P = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k$ is $k - 1$. Meta paths can also be concatenated together with the meta path composition operator.

**Definition 10.** (Meta Path Composition): Meta paths $P^1 = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k, P^2 = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{l-1}} N_l$ can be concatenated together to form a longer meta path $P = P^1 \circ P^2 = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k+l-2}} N_k$. If the ending node type of $P^1$ is the same as the starting node type of $P^2$, i.e., $N_k = N_1 = T_k$. The new composed meta path is of length $k + l - 2$.

Meta path $P = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{k-1}} N_k$ can also be treated as the concatenation of simple meta paths $N_1 \xrightarrow{R_1} N_2, N_2 \xrightarrow{R_2} N_3, \cdots, N_{k-1} \xrightarrow{R_{k-1}} N_k$, which can be represented as $P = R_1 \circ R_2 \circ \cdots \circ R_{k-1} \circ R_k$. SIGKDD Explorations Volume 20, Issue 1 Page 26
3.3 Meta Path across Aligned Heterogeneous Social Networks

Besides the meta paths within one single heterogeneous network, the meta paths can also be defined across multiple aligned heterogeneous networks via the anchor meta paths.

**Definition 11. (Anchor Meta Path):** Let \( G^{(1)} \) and \( G^{(2)} \) be two heterogeneous networks sharing the common anchor information entity of types \( N^{(1)} \subseteq N^{(2)} \) and \( N^{(2)} \subseteq N^{(2)} \) respectively. The anchor meta path between the schemas of networks \( G^{(1)} \) and \( G^{(2)} \) can be represented as \( \Phi = N^{(1)} \prec \text{Anchor} \prec N^{(2)} \) of length 1.

The anchor meta path is the simplest meta path across aligned networks, and a set of inter-network meta paths can be defined based on the intra-network meta paths and the anchor meta path.

**Definition 12. (Inter-Network Meta Path):** A meta path \( \Psi = N_1 \xrightarrow{R_1} N_2 \xrightarrow{R_2} \cdots N_k \xrightarrow{R_k} \) is called an inter-network meta path between networks \( G^{(1)} \) and \( G^{(2)} \) iff \( 3m \in \{1, 2, \ldots, k-1\} \), \( R_m = \text{Anchor} \).

The inter-network meta paths can be viewed as a composition of intra-network meta paths and the anchor meta path via the user node types. An inter-network meta path can be a meta path starting with an anchor meta path followed by the intra-network meta paths, or those with anchor meta paths in the middle. Here, we introduce several categories of inter-network meta paths involving the anchor meta paths at different positions as defined in [104]:

- \( \Psi(G^{(1)}, G^{(2)}) = \Phi(G^{(1)}, G^{(2)}) \), which denotes the simplest inter-network meta path composed of the anchor meta path only between networks \( G^{(1)} \) and \( G^{(2)} \).
- \( \Psi(G^{(1)}, G^{(2)}) = \Phi(G^{(1)}, G^{(2)}) \circ P(G^{(2)}) \), which denotes the inter-network meta path starting with an anchor meta path and followed by the intra-network social meta path in network \( G^{(2)} \).
- \( \Psi(G^{(1)}, G^{(2)}) = P(G^{(1)}) \circ \Phi(G^{(1)}, G^{(2)}) \), which denotes the inter-network meta path starting with the intra-network social meta path in network \( G^{(3)} \) followed by an anchor meta path between networks \( G^{(1)} \) and \( G^{(2)} \).
- \( \Psi(G^{(1)}, G^{(2)}) = P(G^{(1)}) \circ \Phi(G^{(1)}, G^{(2)}) \circ P(G^{(2)}) \), which denotes the inter-network meta path starting and ending with the intra-network social meta path in networks \( G^{(1)} \) and \( G^{(2)} \) respectively connected by an anchor meta path between networks \( G^{(1)} \) and \( G^{(2)} \).

These meta path concepts introduced in this section will be widely used in various social network learning tasks to be introduced later.

4. NETWORK ALIGNMENT

Network alignment is an important research problem and dozens of papers have been published on this topic in the past decades. Depending on specific disciplines, the studied networks can be social networks in data mining [98; 99; 32; 91; 96; 84] protein-protein interaction (PPI) networks and gene regulatory networks in bioinformatics [27; 60; 38; 61], chemical compound in chemistry [63], data schemas in data warehouse [45], ontology in web semantics [14], graph matching in combinatorial mathematics [44], as well as graphs in computer vision [11; 6].

In bioinformatics, the network alignment problem aims at predicting the best mapping between two biological networks based on the similarity of the molecules and their interaction patterns. By studying the cross-species variations of biological networks, network alignment problem can be applied to predict conserved functional modules [58] and infer the functions of proteins [50]. Graemlin [17] conducts pairwise network alignment by maximizing an objective function based on a set of learned parameters. Some works have been done on aligning multiple network in bioinformatics. IsoRank proposed in [62] can align multiple networks greedily based on the pairwise node similarity scores calculated with spectral graph theory. IsoRankN [38] further extends IsoRank model by exploiting a spectral clustering scheme in the framework.

In recent years, with rapid development of online social networks, researchers’ attention starts to shift to the alignment of social networks. Enlightened by the homogeneous network alignment method in [70], Koutra et al. [35] propose to align two bipartite graphs with a fast alignment algorithm. Zafarani et al. [76] propose to match users across social networks based on various node attributes, e.g., username, typing patterns and language patterns etc. Kong et al. formulate the heterogeneous social network alignment problem as an anchor link prediction problem. A two-step supervised network alignment method MNA is proposed in [32] to infer potential anchor links across networks with heterogeneous information in the networks. However, social networks in the real world are mostly partially aligned actually and lots of users are not anchor users. Zhang et al. have proposed a partial network alignment method specifically in [91].

In the social network alignment model building, the anchor links are very expensive to label manually, and achieving a large-sized anchor link training set can be extremely challenging. In [96], Zhang et al. propose to study the network alignment problem based on the PU (Positive and Unlabeled) learning setting instead, where the model is built based on a small amount of positive set and a large unlabeled set. Furthermore, in the case when no training data is available, via inferring the potential anchor user mappings across networks, Zhang et al. have introduced an unsupervised network alignment models for multiple (more than 2) social networks in [98] and an unsupervised network concurrent alignment model via multiple shared information entities simultaneously in [99].

Next, we will introduce the social network alignment methods based on the pairwise and global alignment settings respectively.

4.1 Pairwise Unsupervised Network Alignment

In this part, we will study the network alignment problem based on unsupervised learning setting, which needs no labeled training data. Given two heterogeneous online social networks, which can be represented as \( G^{(1)} = (V^{(1)}, E^{(1)}) \) and \( G^{(2)} = (V^{(2)}, E^{(2)}) \) respectively, the unsupervised network alignment problem aims at inferring the anchor links between networks \( G^{(1)} \) and \( G^{(2)} \). Let \( U^{(1)} \subseteq V^{(1)} \) and \( U^{(2)} \subseteq V^{(2)} \) be the user set in these two networks respectively, we can represent the set of potential anchor links between networks \( G^{(1)} \) and \( G^{(2)} \) as \( \mathcal{A} = U^{(1)} \times U^{(2)} \). In the unsupervised network alignment problem, among all the potential anchor links in set \( \mathcal{A} \), we want to infer which one exists in the real world. Given two homogeneous networks \( G^{(1)} \) and \( G^{(2)} \), mapping the nodes between them is an extremely challenging task, which is also called the graph isomorphism problem [54; 18]. The graph isomorphism has been shown to be NP, but it is still not known whether it also belongs to P or NP-complete yet. So far, no efficient algorithm exists that can address the problem in polynomial time. In this part, we will introduce several heuristics based methods to solve the pairwise homogeneous network alignment problem.

4.1.1 Heuristics based Network Alignment Model

The information generated by users’ online social activities can indicate their personal characteristics. The features introduced in the previous subsection, like ECN, EIC and EAA based on social connection information, similarity/distance measures based on location checkin information, temporal activity closeness, and text word usage similarity can all be used as the predictors indicating
whether the cross-network user pairs are the same user or not. Besides these measures, in this part, we will introduce a category new measures, Relative Centrality Difference (RCD), which can also be applied to solve the unsupervised network alignment problem. The centrality concept can denote the importance of users in the online social networks. Here, we assume that important users in one social network (like celebrities, movie stars and politicians) will be important as well in other networks. Based on such an assumption, the centrality of users in different networks can be an important signal for inferring the anchor links across networks.

**Definition 13.** (Relative Centrality Difference): Given two users $u_i^{(1)}$, $u_j^{(2)}$ from networks $G^{(1)}$ and $G^{(2)}$ respectively, let $C(u_i^{(1)})$ and $C(u_j^{(2)})$ denote the centrality scores of the users, we can define the relative centrality difference (RCD) as

$$RCD(u_i^{(1)}, u_j^{(2)}) = \left( 1 + \frac{|C(u_i^{(1)}) - C(u_j^{(2)})|}{C(u_i^{(1)}) + C(u_j^{(2)})} \right)^{-1}.$$  

Depending on the centrality measures applied, the different type of relative centrality difference measures can be defined. For instance, if we use node degree as the centrality measure, the relative degree difference can be represented as

$$RDD(u_i^{(1)}, u_j^{(2)}) = \left( 1 + \frac{|D(u_i^{(1)}) - D(u_j^{(2)})|}{D(u_i^{(1)}) + D(u_j^{(2)})} \right)^{-1}.$$  

Meanwhile, if the PageRank scores of the nodes are used to define their centrality, we can represent the relative centrality difference measure as

$$RCD(u_i^{(1)}, u_j^{(2)}) = \left( 1 + \frac{|S(u_i^{(1)}) - S(u_j^{(2)})|}{S(u_i^{(1)}) + S(u_j^{(2)})} \right)^{-1}.$$  

In the above equations, $D(u)$ and $S(u)$ denote the node degree and page rank score of node $u$ within each network respectively.

### 4.1.2 IsoRank

Model IsoRank [62] initially proposed to align the biomedical networks, like protein protein interaction (PPI) networks and gene expression networks, can be used to solve the unsupervised social network alignment problem as well. The IsoRank algorithm has two stages. It first associates a score with each possible anchor links between nodes of the two networks. For instance, we can define $r(u_i^{(1)}, u_j^{(2)})$ as the reliability score of an potential anchor link $(u_i^{(1)}, u_j^{(2)})$ between the networks $G^{(1)}$ and $G^{(2)}$, and all such scores can be organized into a vector $\mathbf{r}$ of length $|U^{(1)}| \times |U^{(2)}|$. In the second stage of IsoRank, it constructs the mapping for the networks by extracting from $\mathbf{r}$.

**Definition 14.** (Reliability Score): The reliability score $r(u_i^{(1)}, u_j^{(2)})$ of anchor link $(u_i^{(1)}, u_j^{(2)})$ is highly correlated with the support provided by the mapping scores of the neighborhoods of users $u_i^{(1)}$ and $u_j^{(2)}$. Therefore, we can define $r(u_i^{(1)}, u_j^{(2)})$ as

$$r(u_i^{(1)}, u_j^{(2)}) = \sum_{u_m^{(1)} \in \Gamma(u_i^{(1)})} \sum_{u_n^{(2)} \in \Gamma(u_j^{(2)})} \frac{1}{|\Gamma(u_i^{(1)})| |\Gamma(u_j^{(2)})|} r(u_m^{(1)}, u_n^{(2)}),$$  

where sets $\Gamma(u_i^{(1)})$ and $\Gamma(u_j^{(2)})$ represent the neighborhoods of users $u_i^{(1)}$ and $u_j^{(2)}$ respectively in networks $G^{(1)}$ and $G^{(2)}$. If the networks are weighted, and all the intra-network connections like $(u_i^{(1)}, u_m^{(1)})$ will be associated with a weight $w_i^{(1)}$, we can represented the reliability measure of $r(u_i^{(1)}, u_j^{(2)})$ in the weighted network as

$$r(u_i^{(1)}, u_j^{(2)}) = \sum_{u_m^{(1)} \in \Gamma(u_i^{(1)})} \sum_{u_n^{(2)} \in \Gamma(u_j^{(2)})} w_i^{(1)} r(u_m^{(1)}, u_n^{(2)}),$$  

where the weight term

$$w_i^{(1)} = \frac{w_i^{(1)}(u_i^{(1)}, u_j^{(2)})}{\sum_{u_p^{(2)} \in \Gamma(u_j^{(2)})} w_i^{(1)}(u_i^{(1)}, u_p^{(2)})}.$$  

As we can see, Equation 4 is a special case of Equation 6 with link weight $w_i^{(1)}(u_i^{(1)}, u_j^{(2)}) = 1$ for $u_i^{(1)} \in U^{(1)}$ and $u_j^{(2)} \in U^{(2)}$. Equation 4 can also be rewritten with linear algebra

$$\mathbf{r} = \mathbf{A} \mathbf{r},$$

where matrix $\mathbf{A} \in \mathbb{R}^{|U^{(1)}| \times |U^{(2)}|}$ with entry

$$A(i, j) = \begin{cases} 1 & \text{if } (u_i^{(1)}, u_j^{(2)}) \in E^{(1)}, (u_j^{(2)}, u_k^{(2)}) \in E^{(2)} \text{ and } (u_i^{(1)}, u_k^{(1)}) \exists, \\ 0, & \text{otherwise.} \end{cases}$$

The matrix $\mathbf{A}$ is of dimension $|U^{(1)}| \times |U^{(1)}| \times |U^{(2)}|$, where the row and column indexes correspond to different potential anchor links across the networks. The entry $A(i, j, p, q)$ corresponds the anchor links $(u_i^{(1)}, u_j^{(2)})$ and $(u_p^{(2)}, u_q^{(2)})$. As we can see, the above equation denotes a random walk across the graphs $G^{(1)}$ and $G^{(2)}$ vs. the social links and anchor links in them. The solution to the above equation denotes the principal eigenvector of the matrix $\mathbf{A}$ corresponding to the eigenvalue 1. For more information about the random walk model, please refer to [62].

### 4.1.3 Matrix Inference based Network Alignment

Formally, given a homogeneous network $G^{(1)}$, its structure can be organized as the adjacency matrix $\mathbf{A}_{G^{(1)}} \in \mathbb{R}^{|U^{(1)}| \times |U^{(2)}|}$. If network $G^{(1)}$ is unweighted, then matrix $\mathbf{A}_{G^{(1)}}$ will be a binary matrix and entry $A_{G^{(1)}}(i, p) = 1$ (or $A_{G^{(1)}}(u_i^{(1)}, u_p^{(2)}) = 1$) iff the correspond social link $(u_i^{(1)}, u_p^{(2)})$ exists. In the case that the network is weighted, the entries like $A_{G^{(1)}}(i, p) = 1$ denotes the weight of link $(u_i^{(1)}, u_p^{(2)})$ and 0 if $(u_i^{(1)}, u_p^{(2)})$ doesn’t exist. In a similar way, we can also represent the social adjacency matrix $\mathbf{A}_{G^{(2)}}$ for network $G^{(2)}$ as well.

The network alignment problem aims at inferring an one-to-one node mapping function, that can project nodes from one network to other networks. For instance, we can denote the mapping between networks $G^{(1)}$ to $G^{(2)}$ as $f : U^{(1)} \rightarrow U^{(2)}$. Via the mapping $f$, besides the nodes, the network structure can be projected across networks as well. For instance, given a social connection $(u_i^{(1)}, u_p^{(2)})$ in $G^{(1)}$, we can represent its corresponding connection in $G^{(2)}$ as $(f(u_i^{(1)}), f(u_p^{(2)}))$. Via the mapping $f$, we can denote the network structure differences between $G^{(1)}$ and $G^{(2)}$ as the summation of the link projection difference between

$$L(G^{(1)}, G^{(2)}) = \sum_{u_i^{(1)} \in U^{(1)}} \sum_{u_p^{(2)} \in U^{(2)}} \left( A_{G^{(1)}}(u_i^{(1)}, u_p^{(2)}) - A_{G^{(1)}}(f(u_i^{(1)}), f(u_p^{(2)})) \right)^2.$$
provides an algorithm that can solve the function with a nearly optimal user mapping across different networks. In [70], Umeyama binary and each row and column should contain at most one entry where the matrix exists between networks (i.e., the dashed bi-directional red lines) connecting anchor users in both networks I and II simultaneously, "Bob" is using networks can actually share some common users, e.g., "David" participates to help illustrate the multi-network alignment problem more clearly, we also give an example in Figure 1, which involves multiple anonymized social networks alignment" problem formally [98].

To solve the multi-network alignment problem, two global network alignment methods IsoRankN [38] and UMA (Unsupervised Multi-network Alignment) [98] will be introduced as follows.

4.2 IsoRankN

IsoRankN [38] algorithm is an extension to IsoRank. Based on the learning results of IsoRank, IsoRankN further adapts the spectral clustering method on the induced graph of pairwise alignment scores to achieve the final alignment results. The new approach provides significant advantages not only over the original IsoRank but also over other methods. IsoRankN has 4 main steps: (1) initial network alignment with IsoRank, (2) star spread, (3) spectral partition, and (4) star merging, where steps (3) and (4) will repeat until all the nodes are assigned to a cluster.

Initial Network Alignment: Given k isolated networks \(G^{(1)}, G^{(2)}, \ldots, G^{(k)}\), IsoRankN computes the local alignment scores of node pairs across networks with IsoRank algorithm. For instance, if the networks are unweighted, the alignment score between nodes \(u_i\) and \(u_j\) between networks \(G^{(1)}, G^{(2)}\) can be denoted as.

\[
r(u_i^{(1)}, u_j^{(2)}) = \sum_{u_m^{(1)} \in F(u_i^{(1)})} \sum_{u_n^{(2)} \in F(u_j^{(2)})} \frac{1}{\|F(u_i^{(1)})\| \|F(u_j^{(2)})\|} r(u_m^{(1)}, u_n^{(2)}).
\]

It will lead to a weighted k-partite graph, where the links denotes the anchor links across networks weighted by the scores calculated above. If the networks \(G^{(1)}, \ldots, G^{(k)}\) are all complete graphs, the alignment results will be the maximum weighted cliques. However, in the real world, such an assumption can hardly met, and IsoRankN proposes to use "Star Spread" technique to select a sub-graph with high weights.

Star Spread: For each node in a network, e.g., \(u_i^{(1)}\) in network \(G^{(1)}\), the set of nodes connected with \(u_i^{(1)}\) via potential anchor links can be denoted as set \(\Gamma(u_i^{(1)})\). The nodes in \(\Gamma(u_i^{(1)})\) can be further pruned by removing the nodes connected with weak anchor links. Here, the "weak" denotes the anchor links with a low score calculated with IsoRank. Formally, among all the nodes in \(\Gamma(u_i^{(1)})\), we can denote the node connected to \(u_i^{(1)}\) with the strongest link as.
For each node $u_i^{(t)}$, IsoRankN aims at selecting a subgraph $S_{u_i^{(t)}}^*$ from $S_{u_i^{(t)}}$, which contains the highly weighted neighbors of $u_i^{(t)}$. To achieve such an objective, IsoRankN proposes to identify a subgraph with low conductance from $S_{u_i^{(t)}}$ instead. Formally, given a network $G = (V, E)$, let $S \subset V$ denote a subset of $G$. The conductance of the subgraph involving $S$ can be represented as

$$
\phi(S) = \frac{\sum_{u \in S} \sum_{v \in \bar{S}} w_{uv}}{\min(\text{vol}(S), \text{vol}(\bar{S}))},
$$

(18)

where $\bar{S} = V \setminus S$, and $\text{vol}(S) = \sum_{u \in S} \sum_{v \in V} w_{uv}$. IsoRankN points out that a node subset $S$ containing node $u_i^{(t)}$ can be computed efficiently and effectively with the personalized PageRank algorithm starting from node $u_i^{(t)}$.

**Star Merging:** Considering that links in the star graph $S_{u_i^{(t)}}^*$ are all the anchor links across networks, there exist no intra-network links at all in $S_{u_i^{(t)}}^*$, e.g., the links in network $G^{(1)}$ only. However, in many cases, there may exist multiple nodes corresponding to the same entity inside the network as well. To solve such a problem, IsoRankN proposes a star merging step to combine several star graphs together, e.g., $S_{u_i^{(t)}}^*$ and $S_{u_j^{(t)}}^*$. Formally, given two star graphs $S_{u_i^{(t)}}^*$ and $S_{u_j^{(t)}}^*$, if the following conditions both hold, $S_{u_i^{(t)}}^*$ and $S_{u_j^{(t)}}^*$ can be merged into one star graph.

$$
\forall u \in S_{u_i^{(t)}}^* \setminus \{u_m^{(1)}\}, r(u, u_m^{(1)}) \geq \beta \cdot \max_{v' \in \Gamma(u^{(1)})} r(v', u_i^{(t)}),
$$

(19)

$$
\forall u \in S_{u_j^{(t)}}^* \setminus \{u_m^{(1)}\}, r(u, u_m^{(1)}) \geq \beta \cdot \max_{v' \in \Gamma(u^{(1)})} r(v', u_j^{(t)}).
$$

(20)

### 4.2.2 UMA

The UMA model proposed in [98] addresses the multi-network alignment problem with two steps: (1) unsupervised transitive anchor link inference across multi-networks, and (2) transitive multi-network matching to maintain the one-to-one constraint, where the first step is very similar to the matrix inference based alignment algorithm introduced in Section 4.1.3. Next, we will mainly focus on introducing the transitivity property on the alignment results.

The **transitivity** property should holds for the alignment of any $n$ networks, where the minimum of $n$ is 3. To help illustrate the transitivity property more clearly, here we will use 3 network alignment as an example to introduce the multi-network alignment problem and the UMA model, which can be easily generalized to the case of $n$ networks alignment. Let $G^{(1)}$, $G^{(2)}$ and $G^{(3)}$ be 3 social networks to be aligned concurrently. To accommodate the alignment results and preserve the transitivity property, UMA introduces the following alignment transitivity penalty:

**Definition 15. (Alignment Transitivity Penalty):** Formally, let $T^{(1,2)}$, $T^{(2,3)}$ and $T^{(3,1)}$ be the inferred binary transitional matrices from $G^{(1)}$ to $G^{(2)}$, from $G^{(2)}$ to $G^{(3)}$ and from $G^{(3)}$ to $G^{(1)}$ respectively among these 3 networks. Based on the adjacency matrices $S^{(1)}$, $S^{(2)}$ and $S^{(3)}$ of networks $G^{(1)}$, $G^{(2)}$ and $G^{(3)}$, the alignment transitivity penalty $C(\{G^{(1)}, G^{(2)}, G^{(3)}\})$ introduced by the inferred transitional matrices can be quantified as the number of inconsistent social links being mapped from $G^{(1)}$ to $G^{(2)}$ via two different alignment paths $G^{(1)} \rightarrow G^{(2)} \rightarrow G^{(3)}$ and $G^{(1)} \rightarrow G^{(3)} \rightarrow G^{(2)}$, i.e.,

$$
C(G^{(1)}, G^{(2)}, G^{(3)}) = C(G^{(1)}, G^{(2)}, G^{(3)}) = ||(T^{(1,2)}\cdot T^{(2,3)} - T^{(3,1)})\cdot T^{(3,1)}||^2_2.
$$

(21)

Alignment transitivity penalty is a general penalty concept and can be applied to $n$ networks $\{G^{(1)}; G^{(2)}; \ldots; G^{(n)}\}$, $n \geq 3$ as well, which can be defined as the summation of penalty introduced by any three networks in the set, i.e.,

$$
C(\{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}) = \sum_{i < j < k} C(G^{(i)}, G^{(j)}, G^{(k)}).
$$

(22)

Based on the loss function introduced in Section 4.1.3 and the above alignment transitivity penalty, the optimal binary transitional matrices $T^{(1,2)}$, $T^{(2,3)}$ and $T^{(3,1)}$ should minimize friendship inconsistency and the alignment transitivity penalty at the same time, which is learned by addressing an optimization problem in UMA [98]. The objective function aims at obtaining the hard mappings among users across different networks and entries in all these transitional matrices are binary, which can lead to a fatal drawback: hard assignment can be neither possible nor realistic for networks with star structures as proposed in [35] and the hard subgraph isomorphism [36] is NP-hard. To address the function, UMA proposes to relax the hard binary constraints on the variables first and solve the function with gradient descent. Furthermore, based on the learning results UMA keeps the one-to-one constraint on anchor links by selecting those which can maximize the overall existence probabilities while maintaining the matching transitivity property at the same time.

### 5. LINK PREDICTION

Given a screenshot of an online social network, the problem of inferring the missing links or the links to be formed in the future is called the link prediction problem [39; 23; 104]. Link prediction problem has concrete applications in the real world, and many social network services can be cast to the link prediction problem. For instance, the friend recommendations problem in online social networks can be modeled as the social link prediction problem among users. Users’ trajectory prediction problem can be formulated as the prediction task of potential checkin links between users and offline POIs (point of interest) in location based social networks. The user identifier resolution problem across networks (i.e., the network alignment problem introduced in the previous section) can be modeled as the anchor link prediction problem of user accounts across different online social networks.

In this section, we will introduce the general link prediction problems in online social networks. Formally, given the training set $T_{\text{train}}$ involving links belong to different classes $\mathcal{Y} = \{+1, -1\}$ denoting the links that have been/will be formed and those will never be formed and the test set $T_{\text{test}}$ (with unknown labels), the link prediction problem aims at building a mapping $f: T_{\text{test}} \rightarrow \mathcal{Y}$ to infer the potential labels of links in the test set $T_{\text{test}}$. Depending on the scenarios of the link prediction problems, the existing link prediction works can be divided into several different categories. Traditional link prediction problems are mainly focused on inferring the links in one single homogeneous network [87; 104], like inferring the friendship links among users in online social networks or co-author links in bibliographic networks. As the network structures are becoming more and more complicated, many of them are modeled as the heterogeneous networks involving different types of nodes and complex connections among them.
The heterogeneity of the networks leads to many new link prediction problems, like predicting the links between nodes belonging to different categories and the concurrent inference of multiple types of links in the heterogeneous networks [87; 99]. In recent years, many online social networks have appeared, and lots of new research opportunities exist for researchers and practitioners to study the link prediction problem from the cross-network perspective [87; 104; 84; 83].

Meanwhile, depending on the learning settings used in the problem formulation, the existing link prediction works can be categorized in another way. For some of the link prediction models, they calculate the user-pair closeness as the prediction result without needing any training data, which are referred to as the unsupervised link prediction models [39]. For some other models, they will label the known links into different classes, and use them as the training set to learn a supervised classification models as the base model instead. These models are called the supervised link prediction models [23]. Usually, manual labeling of the links is very expensive and tedious. In recent years, many of the works have proposed to apply semi-supervised learning techniques in the link prediction problem to utilize the links without labels [87; 104; 84; 83].

In this part, we will introduce the link prediction problems in online social networks, including the traditional homogeneous link prediction and cross-network link prediction.

5.1 Homogeneous Network Link Prediction

Traditional link prediction problems are mainly studied based on one homogeneous network, involving one single type of nodes and links. In this section, we will first briefly introduce how to use the social closeness measures for link prediction tasks. To integrate different social closeness measures together in the link prediction task, we will talk about the supervised link prediction model. Furthermore, several semi-supervised link prediction models will also be introduced in this section, which formulate the link prediction as a semi-supervised learning problem.

5.1.1 Unsupervised Link Prediction

Given a snapshot of a homogeneous network $G = (V, E)$, the unsupervised link prediction methods [39] aim at inferring the potential links that will be formed in the future. Usually, the unsupervised link prediction models will calculate the closeness scores of the node pairs, which will be used as the predicted confidence scores of these links. Depending on the specific scenario and the link formation assumptions applied, different measures have been proposed for the link prediction models.

**Local Neighbor based Predicators:** Local neighbor based predictors are based on regional social network information, i.e., neighbors of users in the network. Consider, for example, given a social link $(u, v)$ in network $G$, where $u$ and $v$ are both users in $G$, the neighbor sets of $u, v$ can be represented as $\Gamma(u)$ and $\Gamma(v)$ respectively. Based on $\Gamma(u)$ and $\Gamma(v)$, the following measures predicting the proximity of users $u$ and $v$ in network $G$ can be obtained.

1. **Preferential Attachment Index** (PA) [5]:
   \[
   PA(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u)| \cdot |\Gamma(v)|}.
   \]  
   $PA(u, v)$ uses the product of the degrees of users $u$ and $v$ in the network as the proximity measure, considering that new links are more likely to appear between users who have large number of social connections.

2. **Common Neighbor** (CN) [24]:
   \[
   CN(u, v) = |\Gamma(u) \cap \Gamma(v)|.
   \]  
   $CN(u, v)$ uses the number of shared neighbor as the proximity score of user $u$ and $v$. The larger $CN(u, v)$ is, the closer user $u$ and $v$ are in the network.

3. **Jaccard’s Coefficient** (JC) [24]:
   \[
   JC(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}.
   \]  
   $JC(u, v)$ takes the total number of neighbors of $u$ and $v$ into account, considering that $CN(u, v)$ can be very large because each one has a lot of neighbors rather than they are strongly related to each other.

4. **Adamic/Adar Index** (AA) [1]:
   \[
   AA(u, v) = \sum_{w \in (\Gamma(u) \cap \Gamma(v))} \frac{1}{\log |\Gamma(w)|}
   \]  
   Different from $JC(u, v)$, $AA(u, v)$ further gives each common neighbor of user $u$ and $v$ a weight $\frac{1}{\log |\Gamma(w)|}$, to denote its importance.

5. **Resource Allocation Index** (RA) [107]:
   \[
   RA(u, v) = \sum_{w \in (\Gamma(u) \cap \Gamma(v))} \frac{1}{|\Gamma(w)|}
   \]  
   $RA(u, v)$ gives each common neighbor a weight $\frac{1}{|\Gamma(w)|}$ to represent its importance, where those with larger degrees will have a less weight number.

All these predictors are called local neighbor based predictors as they are all based on users’ local social network information.

**Global Path based Predicators:** In addition to the local neighbor based predictors, many other predictors based on paths in the network have also been proposed to measure the proximity among users.

1. **Shortest Path** (SP) [23]:
   \[
   SP(u, v) = \min \{|p_u \rightarrow v|\},
   \]  
   where $p_u \rightarrow v$ denotes a path from $u$ to $v$ in the network and $|p|$ represents the length of path $p$.

2. **Katz** [28]:
   \[
   Katz(u, v) = \sum_{l=1}^{\infty} \beta^l |p^l_{u \rightarrow v}|
   \]  
   where $p^l_{u \rightarrow v}$ is the set of paths of length $l$ from $u$ to $v$ and parameter $\beta \in [0, 1]$ is a regularizer of the predictor. Normally, a small $\beta$ favors shorter paths as $\beta^l$ can decay very quickly when $\beta$ is small, in which case $Katz(u, v)$ will be behave like the predictors based on local neighbors.

**Random Walk based Link Prediction:** In addition to the unsupervised link predictors which can be obtained from the networks directly, there exists another category link prediction methods which can calculate the proximity scores among users based on random walk [21; 19; 33; 4; 68; 42; 24]. In this part, we will introduce the concept of random walk at first. Next, we will introduce the proximity measures based on random walk, which include the commute time [19; 42; 24], hitting time [19; 42; 24] and cosine similarity [19; 42; 24].

Let matrix $A$ be the adjacency matrix of network $G$, where $A(i, j) = 1$ iff social link $(u_i, u_j) \in \mathcal{E}$, where $u_i, u_j \in V$. The normalized matrix of $A$ by rows will be $P = D^{-1}A$, where diagonal matrix $D$ of $A$ has value $D(i, i) = \sum_j A(i, j)$ on its diagonal and $P(i, j)$ stores the probability of stepping on node $u_j \in V$ from node $u_i \in V$. Let entries in vector $x^{(\tau)}(i)$ denote the probabilities that a random walker is at user node $u_i \in V$ at time $\tau$. Then we
have the updating equation of entry $x^{(r)}(i)$ via the random walk as follows:

$$x^{(r+1)}(i) = \sum_j x^{(r)}(j)P(j, i).$$  

(32)

In other words, the updating equation of vector $x$ will be represented as:

$$x^{(r+1)} = Px^{(r)}.$$  

(33)

By keeping updating $x$ according to the following equation until convergence, we can have the stationary vector $x^{(r+1)}$ as

$$x^{(r+1)} = P^{T}x^{(r)}, \quad x^{(r+1)} = x^{(r)}.$$  

(34)

The above equation is equivalent to

$$v = D^T v,$$  

(35)

where $v$ denotes the stationary random walk probability vector.

The above equation denotes that the final stationary distribution vector $v$ is actually an eigenvector of matrix $D^T$ corresponding to eigenvalue 1. Some existing works have pointed out that if a markov chain is irreducible [19] and aperiodic [19] then the largest eigenvalue of the transition matrix will be equal to 1 and all the other eigenvalues will be strictly less than 1. In addition, in such a condition, there will exist one single unique stationary distribution which is vector $v$ obtained at convergence of the updating equations.

**Proximity Measures based on Random Walk**

1. **Hit Time (HT):**

$$HT(u, v) = E \left( \min\{\tau | \tau \in \mathbb{N}^+, X^{(\tau)} = v \wedge X^{0} = u \} \right),$$  

(36)

where variable $X^{(\tau)} = v$ denotes that a random walker is at node $v$ at time $\tau$.

$HT(u, v)$ counts the average steps that a random walker takes to reach node $v$ from node $u$. According to the definition, the hitting time measure is usually asymmetric, $HT(u, v) \neq HT(v, u)$. Based on matrix $P$ defined before, the definition of $HT(u, v)$ can be redefined as [19]:

$$HT(u, v) = 1 + \sum_{w \in \mathcal{L}(u)} P_{u,w}HT(w, v).$$  

(37)

2. **Commute Time (CT):**

$$CT(u, v) = HT(u, v) + HT(v, u).$$  

(38)

$CT(u, v)$ counts the expectation of steps used to reach node $u$ from $v$ and those needed to reach node $v$ from $u$. According to existing works, the commute time, $CT(u, v)$, can be obtained as follows

$$CT(u, v) = 2m(L^1_{u,u} + L^1_{v,v} - 2L^1_{u,v}),$$  

(39)

where $L^1$ is the pseudo-inverse of matrix $L = D_A - A$.

3. **Cosine Similarity based on $L^1$ (CS):**

$$CS(u, v) = \frac{x_u^T x_v}{\sqrt{(x_u^T x_u)(x_v^T x_v)}},$$  

(40)

where, $x_u = (L^1)^\dagger e_u$ and vector $e_u$ is a vector of 0s except the entries corresponding to node $u$ that is filled with 1. According to existing works [19; 42], the cosine similarity based on $L^1$, $CS(u, v)$, can be obtained as follows,

$$CS(u, v) = \frac{L^1_{u,v}}{\sqrt{L^1_{u,u}L^1_{v,v}}}.$$  

(41)

4. **Random Walk with Restart (RWR):** Based on the definition of random walk, if the walker is allowed to return to the starting point with a probability of $1 - c$, where $c \in [0, 1]$, then the new random walk method is formally defined as random walk with restart, whose updating equation is shown as follows:

$$\begin{align*}
\left\{ \begin{array}{l}
x_u^{(r+1)} = cP^{T}x_u^{(r)} + (1-c)e_u, \\
x_v^{(r+1)} = x_v^{(r)}.
\end{array} \right.
\end{align*}$$  

(42)

Keep updating $x$ until convergence, the stationary distribution vector $x$ can meet

$$x_u = (1-c)(I-cP^{T})^{-1}e_u.$$  

(43)

The proximity measure based on random walk with restart between user $u$ and $v$ will be

$$RWR(u, v) = x_u(v),$$  

(44)

where $x_u(v)$ denotes the entry corresponding to $v$ in $x_u$.

**5.1.2 Supervised Link Prediction**

In some cases, links in the networks are explicitly categorized into different groups, like links denoting friends vs those representing enemies, friends (formed connections) vs strangers (no connections). Given a set of labeled links, e.g., set $\mathcal{E}$, containing links belonging to different classes, the supervised link prediction [23] problem aims at building a supervised learning model with the labeled set. The learnt model will be applied to determine the labels of links in the test set. In this part, we still take the link formation problem as an example to illustrate the supervised link prediction model.

To represent each of the social links, like link $l = (u, v) \in \mathcal{E}$ between nodes $u$ and $v$, a set of features representing the characteristics of the link $l$ or nodes $u, v$ will be extracted in the model building. Normally, the features can be extracted for links in the prediction task can be divided into two categories:

**Link Feature Extraction**

- **Features of Nodes**: The characteristics of the nodes can be denoted by various measures, like these various node centrality measures. For instance, for the link $(u, v)$, based on the known links in the training set, the centrality measures can be computed based on degree, normalized degree, eigenvector, Katz, PageRank, Betweenness of nodes $u$ and $v$ as part of the features for link $(u, v)$.

- **Features of Links**: The characteristics of the links in the networks can be calculated by computing the closeness between the nodes composing the nodes. For instance, for link $(u, v)$, based on the known links in the training set, the closeness measures can be computed based on reciprocity, common neighbor, Jaccard’s coefficient, Adamic/Adar, shortest path, Katz, hitting time, commute time, etc. between nodes $u$ and $v$ as the features for link $(u, v)$.

We can append the features for nodes $u, v$ and those for link $(u, v)$ together and represent the extracted feature vector for link $l = (u, v)$ as vector $x_l \in \mathbb{R}^{k \times 1}$, whose length is $k$ in total.

**Link Prediction Model**

With the training set $\mathcal{L}_{train}$, the feature vectors and labels for the links in $\mathcal{L}_{train}$ can be represented as the training data $(\{x_l, y_l\}) \subseteq \mathcal{L}_{train}$. Meanwhile, with the testing set $\mathcal{L}_{test}$, the features extracted for the links in it can be represented as $\{x_l\} \subseteq \mathcal{L}_{test}$. Different classification models can be used as the base model for the link prediction task, like the Decision Tree, Artificial Neural Network and Support Vector Machine (SVM) [2]. The model can be trained with the
training data, and the labels of links in the test can be determined by applying models to the test set.

Depending on the specific model being applied, the output of the link prediction result can include (1) the predicted labels of the links, and (2) the prediction confidence scores/probability scores of links in the test set.

5.1.3 PU Link Prediction

In the real world, for the links which are unlabeled, some of them can actually be formed in the future. In this subsection, we will introduce a method MLI to solve the PU link prediction problem in one single network [104]. From a given network, e.g., $G$, two disjoint sets of links: connected (i.e., formed) links $P$ and unconnected links $U$, can be obtained. To differentiate these links, MLI uses a new concept “connection state”, $z$, to show whether a link is connected (i.e., formed) or unconnected in network $G$. For a given link $l$, if $l$ is connected in the network, then $z(l) = +1$; otherwise, $z(l) = -1$. As a result, MLI can have the “connection states” of links in $P$ and $U$ to be: $z(P) = +1$ and $z(U) = -1$.

Besides the “connection state”, links in the network can also have their own “labels”, $y$, which can represent whether a link is to be formed or will never be formed in the network. For a given link $l$, if $l$ has been formed or to be formed, then $y(l) = +1$; otherwise, $y(l) = -1$. Similarly, MLI can have the “labels” of links in $P$ and $U$ to be: $y(P) = +1$ but $y(U)$ can be either $+1$ or $-1$, as $U$ can contain both links to be formed and links that will never be formed. By using $P$ and $U$ as the positive and negative training sets, MLI can build a connection prediction model $M_c$, which can be applied to predict whether a link exists in the original network, i.e., the connection state of a link. Let $l$ be a link to be predicted, by applying $M_c$ to classify $l$, the connection probability of $l$ can be represented to be:

**Definition 16. (Connection Probability):** The probability that link $l$’s connection states is predicted to be connected (i.e., $z(l) = +1$) is formally defined as the connection probability of link $l$: $p(z(l) = +1|x(l))$, where $x(l)$ denotes the feature vector extracted for link $l$ based on meta path.

Meanwhile, if we can obtain a set of links that “will never be formed”, i.e., “-1” links, from the network, which together with $P$ (“+1” links) can be used to build a link formation prediction model, $M_f$, which can be used to get the formation probability of $l$ to be:

**Definition 17. (Formation Probability):** The probability that link $l$’s label is predicted to be formed or will be formed (i.e., $y(l) = +1$) is formally defined as the formation probability of link $l$: $p(y(l) = +1|x(l))$.

However, from the network, we have no information about “links that will never be formed” (i.e., “-1” links). As a result, the formation probabilities of potential links that we aim to obtain can be very challenging to calculate. Meanwhile, the correlation between link $l$’s connection probability and formation probability has been proved in existing works [15] to be:

$$p(y(l) = +1|x(l)) \propto p(z(l) = +1|x(l)).$$

In other words, for links whose connection probabilities are low, their formation probabilities will be relatively low as well. This rule can be utilized to extract links which can be more likely to be the reliable “-1” links from the network. MLI proposes to apply the the link connection prediction model $M_c$, built with $P$ and $U$ to classify links in $U$ to extract the reliable negative link set.

**Definition 18. (Reliable Negative Link Set):** The reliable negative links in the unconnected link set $U$ are those whose connection probabilities predicted by the link connection prediction model, $M_c$, are lower than threshold $\epsilon \in [0, 1]$:

$$RN = \{l| l \in U, p(z(l) = +1|x(l)) < \epsilon\}.$$  

Some Heuristic methods have been proposed to set the optimal threshold $\epsilon$, e.g., the spy technique proposed in [41]. As shown in Figure 2, MLI proposes randomly select a subset of links in $P$ as the spy, $SP$, whose proportion is controlled by $s\%$. $s\% = 15\%$ is used as the default sample rate in [104]. Sets ($P - SP$) and ($U \cup SP$) are used as positive and negative training sets to the spy prediction model, $M_c$. By applying $M_c$ to classify links in ($U \cup SP$), their connection probabilities can be represented to be:

$$p(z(l) = +1|x(l)), l \in (U \cup SP),$$

and parameter $\epsilon$ is set as the minimal connection probability of spy links in $SP$:

$$\epsilon = \min_{l \in SP} p(z(l) = +1|x(l)).$$

With the extracted reliable negative link set $RN$, MLI can solve the PU link prediction problem with classification based link prediction methods, where $P$ and $RN$ are used as the positive and negative training sets respectively.

5.2 Inter-Network Link Prediction

Besides the link prediction problems in one single target network, some research works have been done on simultaneous link prediction in multiple aligned online social networks concurrently [87; 104; 83].

5.2.1 MLI

Method MLI proposed in [104] is a general link prediction framework and can be applied to predict social links in $n$ partially aligned networks simultaneously. When it comes to $n$ partially aligned networks $G^{(1)}, \ldots, G^{(n)}$, the optimal labels of potential links $\{L^{(1)}, L^{(2)}, \ldots, L^{(n)}\}$ of networks $G^{(1)}, \ldots, G^{(n)}$ will be:

$$\{\hat{y}^{(1)}, \hat{y}^{(2)}, \ldots, \hat{y}^{(n)}\} = \max_{y^{(1)}, \ldots, y^{(n)}} p(y^{(1)}, \ldots, y^{(n)}|G^{(1)}, \ldots, G^{(n)}).$$

The above target function is very complex to solve and in [104], MLI proposes to obtain the solution by updating one variable, e.g., $\hat{y}^{(1)}$, and fix other variables, e.g., $\hat{y}^{(2)}, \ldots, \hat{y}^{(n)}$, alternatively with the following equation [87]:

$$\begin{align*}
\{\hat{y}^{(1)}\} &= \arg \max_{y^{(1)}} p(y^{(1)}|G^{(1)}, G^{(2)}, \ldots, G^{(n)}), \\
\{\hat{y}^{(2)}\} &= \arg \max_{y^{(2)}} p(y^{(2)}|G^{(2)}, G^{(1)}, G^{(3)}, \ldots, G^{(n)}), \\
\{\hat{y}^{(3)}\} &= \arg \max_{y^{(3)}} p(y^{(3)}|G^{(3)}, G^{(1)}, G^{(2)}, G^{(4)}, \ldots, G^{(n)}), \\
&\vdots \\
\{\hat{y}^{(n)}\} &= \arg \max_{y^{(n)}} p(y^{(n)}|G^{(n)}, G^{(1)}, G^{(2)}, \ldots, G^{(n-1)}).
\end{align*}$$
When predicting social links in network $G^{(i)}$, MLI can extract features based on the intra-network social meta path extracted from $G^{(i)}$ and those extracted based on the inter-network social meta path across $G^{(1)}, G^{(2)}, \ldots, G^{(i-1)}, G^{(i+1)}, \ldots, G^{(n)}$ for links in $P^{(i)}, U^{(i)}$ and $C^{(i)}$. Feature vectors $x(P), x(U)$ as well as the labels, $y(P), y(U)$, of links in $P$ and $U$ are passed to the PU link prediction model $M^{(i)}$ and the meta path selection model $MS^{(i)}$. The formation probabilities of links in $L^{(i)}$ predicted by model $M^{(i)}$ will be used to update the network by replace the weights of $L^{(i)}$ with the newly predicted formation probabilities. The initial weights of these potential links in $L^{(i)}$ are set as 0. After finishing these steps on $G^{(i)}$, we will move to conduct similar operations on $G^{(i+1)}$. MLI iteratively predicts links in $G^{(i)}$ to $G^{(n)}$ alternatively in a sequence until the results in all of these networks converge.

5.2.2 SLAMPRED

The cross-network link prediction model SLAMPRED introduced in [83] aims at inferring the links for emerging networks based on semi-supervised learning setting. SLAMPRED proposes to embed the feature vectors of links from aligned networks into a shared feature space. Via the shared feature space, knowledge from the source networks will be effectively transferred to the target network.

Formally, let $G'$ denote the target emerging network, where the user set can be represented as $U'$. The existing social connections in $G'$ can be represented as the binary social adjacency matrix $A' \in \{0, 1\}^{\vert U'\vert \times \vert U'\vert}$, where entry $A'(i,j) = 1$ iff the corresponding social link $(u'_i, u'_j)$ exists between users $u'_i$ and $u'_j$ in $G'$. In the studied problem here, our objective is to infer the potential unobserved social links for the target network, which can be achieved by finding a sparse and low-rank predictor matrix $S \in S$ from some convex admissible set $S \subset \mathbb{R}^{\vert U'\vert \times \vert U'\vert}$. Meanwhile, the inconsistency between the inferred matrix $S$ and the observed social adjacency matrix $A'$ can be represented as the loss function $l(S, A')$. The optimal social link predictor for the target network can be achieved by minimizing the loss term, i.e.,

$$\arg \min_{S \in S} \ l(S, A'). \quad (52)$$

Meanwhile, to utilize the other heterogeneous information available in the emerging network $G'$ and other external source networks $G^1, G^2, \ldots, G^K$. SLAMPRED proposes to extract a group of intimacy features and project the link instances to a shared feature space as introduced in [83]. The adapted features from the target network and external sources can be represented as tensors $\hat{X}^1, \hat{X}^2, \ldots, \hat{X}^K$. Formally, the intimacy scores of the potential social links based on these adapted features can be set as

$$\text{int}(S, \hat{X}) = \sum_{k=1}^{d} \left\| S \odot \hat{X}(k,:,:) \right\|_1, \quad i \in \{1, 2, \ldots, K\} \quad (53)$$

$$\text{int}(S, \hat{X}^1, \ldots, \hat{X}^K) = \sum_{k=1}^{K} \alpha^k \cdot \text{int}(S, \hat{X}^k), \quad (54)$$

where users in $\hat{X}$ are organized in the same order as $X$. Parameters $\alpha^k$ denotes the importance of the information transferred from the source network $G^k$.

By adding the intimacy terms about the source networks into the objective function, the equation can be rewritten as follows:

$$\arg \min_{S \in S} \ [l(S, A') - \alpha^k \cdot \text{int}(S, \hat{X}^k) - \sum_{k=1}^{K} \alpha^k \cdot \text{int}(S, \hat{X}^k)] + \gamma \cdot \|S\|_1 + \tau \cdot \|S\|_*$, \quad (55)$$

where $\|S\|_1$ and $\|S\|_*$ denote the $L_1$-norm and trace-norm of matrix $S$ respectively.

By studying the objective function, we observe that the intimacy terms are convex while the empirical loss term $l(S, A')$ is non-convex. In [83], the introduced model proposes to approximate it with other classical loss functions (e.g., the hinge loss and the Frobenius norm) instead, and the convex squared Frobenius norm loss function is used in [83] (i.e., $l(S, A') = \|S - A'|^2_F$). Therefore, the above objective function can be represented as a convex loss term minus another convex term together with two convex non-differentiable regularizers, which actually renders the objective function non-trivial. According to the existing works [75; 64], this kind of objective function can be addressed with the concave-convex procedure (CCCP). CCCP is a majorization-minimization algorithm that solves the difference of convex functions problems as a sequence of convex problems. Meanwhile, the regularization terms can be effectively handled with the proximal operators in each iteration of the CCCP process.

6. COMMUNITY DETECTION

In the real-world online social networks, users tend to form different social groups [3]. Users belonging to the same groups usually have more frequent interactions with each other, while those in different groups will have less interactions on the other hand [106]. Formally, such social groups form by users in online social networks are called the online social communities [97]. Online social communities will partition the network into a number of connected communities, where the intra-community social connections are usually far more dense compared with the inter-community social connections [97]. Meanwhile, from the mathematical representation perspective, due to these online social communities, the social network adjacency matrix tend to be not only sparse but also low-rank [101].

Identifying the social communities formed by users in online social networks is formally defined as the community detection problem [97; 95; 26]. Community detection is a very important problem for online social network studies, as it can be crucial prerequisite for numerous concrete social network services: (1) better organization of users’ friends in online social networks (e.g., Facebook and Twitter), which can be achieved by applying community detection techniques to partition users’ friends into different categories, e.g., schoolmates, family, celebrities, etc. [16]; (2) better recommender systems for users with common shopping preference in e-commerce social sites (e.g., Amazon and Epinions), which can be addressed by grouping users with similar purchase records into the same clusters prior to recommender system building [56]; and (3) better identification of influential users [69] for advertising campaigns in online social networks, which can be attained by selecting the most influential users in each community as the seed users in the viral marketing [55].

In this section, we will focus on introducing the social community detection problem in online social networks. Given a heterogeneous network $G$ with node set $V$, the involved user nodes in network $G$ can be represented as set $U \subset V$. Based on both the social structures among users as well as the diverse attribute information from the network $G$, the social community detection problem aims at partitioning the user set $U$ into several subsets $C = \{U_1, U_2, \ldots, U_k\}$, where each subset $U_i, i \in \{1, 2, \ldots, k\}$ is called a social community. Term $k$ formally denotes the total number of partitioned communities, which is usually provided as a hyper-parameter in the problem.

Depending on whether the users are allowed to be partitioned into multiple communities simultaneously or not, the social community
The community detection problem can actually be categorized into two different types:

- **Hard Social Community Detection**: In the hard social community detection problem, each user will be partitioned into one single community, and all the social communities are disjoint without any overlap. In other words, given the communities \( C = \{ U_1, U_2, \ldots, U_k \} \) detected from network \( G \), we have \( U = \bigcup_{i=1}^{k} U_i \) and \( U_i \cap U_j = \emptyset, \forall i, j \in \{1, \ldots, k\} \).

- **Soft Social Community Detection**: In the soft social community detection problem, users can belong to multiple social communities simultaneously. For instance, if we apply the Mixture-of-Gaussian Soft Clustering algorithm as the base community detection model [105; 74], each user can belong to multiple communities with certain probabilities. In the soft social community detection result, the communities are no longer disjoint and will share some common users with other communities.

Meanwhile, depending on the network connection structures, the community detection problem can be categorized as directed network community detection [43] and undirected network community detection [106]. Based on the heterogeneity of the network information, the community detection problem can be divided into the homogeneous network community detection [72] and heterogeneous network community detection [57; 66; 85; 101]. Furthermore, according to the number of networks involved, the community detection problem involves single network community detection [37] and multiple network community detection [97; 95; 26; 85; 101]. In this section, we will take the hard community detection problem as an example to introduce the existing models proposed for conventional (one single) homogeneous social network, and especially the recent broad learning based (multiple aligned) heterogeneous social networks [32; 86; 87; 104] respectively.

This section is organized as follows. At the beginning, in Section 6.1, we will introduce the community detection problem and the existing methods proposed for traditional one single homogeneous networks. After that, we will talk about the latest research works on social community detection across multiple aligned heterogeneous networks. In Section 6.2, we will be focused on the concurrent mutual community detection [97] across multiple aligned heterogeneous networks simultaneously, where information from other aligned networks will be applied to refine their community detection results mutually.

### 6.1 Homogeneous Network Community Detection

Social community detection problem has been studied for a long time, and many community detection models have been proposed based on different types of techniques. In this section, we will talk about the social community detection problem for one single homogeneous network \( G \), whose objective is to partition the user set \( U \) in network \( G \) into \( k \) disjoint subsets \( C = \{ U_1, U_2, \ldots, U_k \} \), where \( U = \bigcup_{i=1}^{k} U_i \) and \( U_i \cap U_j = \emptyset, \forall i, j \in \{1, 2, \ldots, k\} \). Several different community detection methods will be introduced, which include node proximity based community detection, modularity maximization based community detection, and spectral clustering based community detection.

#### 6.1.1 Node Proximity based Community Detection

The node proximity based community detection method assumes that “close nodes tend to be in the same communities, while the nodes far away from each other will belong to different communities”. Therefore, the node proximity based community detection model partition the nodes into different clusters based on the node proximity measures [39]. Various node proximity measures can be used here, including the node structural equivalence to be introduced as follows, as well as various node closeness measures as introduced in Section 5.1.1.

In a homogeneous network \( G \), the proximity of nodes, like \( u \) and \( v \), can be calculated based on their positions and connections in the network structure.

**Definition 19.** (Structural Equivalence): Given a network \( G = (\mathcal{V}, \mathcal{E}) \), two nodes \( u, v \in \mathcal{V} \) are said to be structural equivalent iff

1. Nodes \( u \) and \( v \) are not connected and \( u \) and \( v \) share the same set of neighbors (i.e., \( (u, v) \notin \mathcal{E} \land \Gamma(u) = \Gamma(v) \)).
2. Or \( u \) and \( v \) are connected and excluding themselves, \( u \) and \( v \) share the same set of neighbors (i.e., \( (u, v) \in \mathcal{E} \land \Gamma(u) \setminus \{v\} = \Gamma(v) \setminus \{u\} \)).

For the nodes which are structural equivalent, they are substitutable and switching their positions will not change the overall network structure. The structural equivalence concept can be applied to partition the nodes into different communities. For the nodes which are structural equivalent, they can be grouped into the same communities, while for the nodes which are not equivalent in their positions, they will be partitioned into different groups. However, the structural equivalence can be too restricted for practical application in detecting the communities in real-world social networks. Computing the structural equivalence relationships among all the node pairs in the network can lead to very high time cost. What’s more, the structural equivalence relationship will partition the social network structure into lots of small-sized fragments, since the users will have different social patterns in making friends online and few user will have identical neighbors actually.

To avoid the weakness mentioned above, some other measures are proposed to measure the proximity among nodes in the networks. For instance, as introduced in Section 5.1.1, the node closeness measures based on the social connections can all be applied here to compute the node proximity, e.g., “common neighbor”, “Jaccard’s coefficient”. Here, if we use “common neighbor” as the proximity measure, by applying the “common neighbor” measure to the network \( G \), the network \( G \) can be transformed into a set of instances \( V \) with mutual closeness scores \( \{ c(u, v) \}_{u,v \in V} \). Some existing similarity/distance based clustering algorithms, like k-Medoids, can be applied to partition the users into different communities.

#### 6.1.2 Modularity based Community Detection

Besides the pairwise proximity of nodes in the network, the connection strength of a community is also very important in the community detection process. Different measures have been proposed to compute the strength of a community, like the modularity measure [48] to be introduced in this part.

The modularity measure takes account of the node degree distribution. For instance, given the network \( G \), the expected number of links existing between nodes \( u \) and \( v \) with degrees \( D(u) \) and \( D(v) \) can be represented as \( \frac{D(u) \cdot D(v)}{2|\mathcal{E}|} \). Meanwhile, in the network, the real number of links existing between \( u \) and \( v \) can be denoted as entry \( A[u, v] \) in the social adjacency matrix \( \mathbf{A} \). For the user pair \( (u, v) \) with a low expected connection confidence score, if they are connected in the real world, it indicates that \( u \) and \( v \) have a relatively strong relationship with each other. Meanwhile, if the community detection algorithm can partition such user pairs into the same group, it will be able to identify very strong social communities from the network.

Based on such an intuition, the strength of a community, e.g., \( U_i \in C \) can be defined as

\[
\sum_{u,v \in U_i} \left( A[u,v] - \frac{D(u) \cdot D(v)}{2|\mathcal{E}|} \right). \tag{57}
\]
Furthermore, the strength of the overall community detection result $C = \{U_1, U_2, \ldots, U_k\}$ can be defined as the *modularity* of the communities as follows.

**Definition 20.** (Modularity): Given the community detection result $C = \{U_1, U_2, \ldots, U_k\}$, the modularity of the community structure is defined as

$$Q(C) = \frac{1}{|E|} \sum_{u \in U_i, v \in U_j} \left( A[u, v] - \frac{D(u) \cdot D(v)}{2|E|} \right).$$

(58)

The modularity concept effectively measures the strength of the detected community structure. Generally, for a community structure with a larger *modularity* score, it indicates a good community detection result.

Another way to explain the *modularity* is from the number of links within and across communities. By rewriting the above *modularity* equation, we can have

$$Q(C) = \frac{1}{2|E|} \sum_{u \in U_i, v \in U_j} \left( A[u, v] - \frac{D(u) \cdot D(v)}{2|E|} \right).$$

(59)

$$= \frac{1}{2|E|} \left( \sum_{u \in U_i, v \in U_j} A[u, v] - \sum_{u \in U_i} D(u) \sum_{v \in U_j} D(v) \right).$$

(60)

$$= \frac{1}{2|E|} \left( \sum_{u \in U_i, v \in U_j} A[u, v] - \left( \sum_{u \in U_i} D(u) \sum_{v \in U_j} D(v) \right)^2 \right).$$

(61)

In the above equation, term $\sum_{u \in U_i, v \in U_j} A[u, v]$ denotes the number of links connecting users within the community $U_i$ (which will be 2 times the intra-community links for undirected networks, as each link will be counted twice). Term $\sum_{u \in U_i} D(u)$ denotes the sum of node degrees in community $U_i$, which equals to the number of intra-community and inter-community links connected to nodes in community $U_i$. If there exist lots of inter-community links, then the *modularity* measure will have a smaller value. On the other hand, if the inter-community links are very rare, the *modularity* measure will have a larger value. Therefore, maximizing the community *modularity* measure is equivalent to minimizing the inter-community link numbers.

The *modularity* measure can also be represented with linear algebra equations. Let matrix $A$ denote the adjacency matrix of the network, and vector $d \in \mathbb{R}^{|V| \times 1}$ denote the degrees of nodes in the network. The *modularity* matrix can be defined as

$$B = A - \frac{dd^T}{2|E|}.$$  

(64)

Let matrix $H \in \{0, 1\}^{|V| \times k}$ denotes the communities that users in $V$ belong to. In real application, such a binary constraint can be relaxed to allow real value solutions for matrix $H$. The optimal community detection result can be obtained by solving the following objective function

$$\max \frac{1}{2|E|} \text{Tr}(H^T BH) \quad \text{s.t.} \quad H^T H = I,$$

(65)

(66)

where constraint $H^T H = I$ ensures there are not overlap in the community detection result.

The above objective function looks very similar to the objective function of *spectral clustering* to be introduced in the next section. After obtaining the optimal $H$, the communities can be obtained by applying the K-Means algorithm to $H$ to determine the cluster labels of each node in the network.

### 6.1.3 Spectral Clustering and Community Detection

In the community detection process, besides maximizing the proximity of nodes belonging to the same communities (as introduced in Section 6.1.1), minimizing the connections among nodes in different clusters is also an important factor. Different from the previous proximity based community detection algorithms, another way to address the community detection problem is from the cost perspective. Partition the nodes into different clusters will cut the links among the clusters. To ensure the nodes partitioned into different clusters have less connections with each other, the number of links to be cut in the community detection process should be as small as possible [59; 71].

**Cut:** Formally, given the community structure $C = \{U_1, U_2, \ldots, U_k\}$ detected from network $G$. The number of links cut [59] between communities $U_i, U_j \in C$ can be represented as

$$\text{cut}(U_i, U_j) = \sum_{u \in U_i, v \in U_j} I(u, v),$$

(67)

where function $I(u, v) = 1$ if $(u, v) \in E$; otherwise, it will be 0. The total number of links cut in the partition process can be represented as

$$\text{cut}(C) = \sum_{U_i \in C} \text{cut}(U_i).$$

(68)

where set $\bar{U} = C \setminus U_i$ denotes the communities except $U_i$.

By minimizing the cut cost introduced in the partition process, the optimal community detection result can be obtained with the minimum number of cross-community links. However, as introduced in [59; 71], by minimizing the cut of edges across clusters, the results may involve high imbalanced communities, some community may involve one single node. Such a problem will be much more severe when it comes to the real-world social network data. In the following part of this section, we will introduce two other cost measures that can help achieve more balanced community detection results.

**Ratio-Cut and Normalized-Cut:** As shown in the above example, the minimum cut cost treat all the links in the network equally, and can usually achieve very imbalanced partition results (e.g., a singleton node as a cluster) when applied in the real-world community detection problem. To overcome such a disadvantage, some models have been proposed to take the community size into consideration. The community size can be calculated by counting the number of nodes or links in each community, which will lead to two new cost measures: *ratio-cut* and *normalized-cut* [59; 71].

Formally, given the community detection result $C = \{U_1, U_2, \ldots, U_k\}$ in network $G$, the *ratio-cut* and *normalized-cut* costs introduced in the community detection result can be defined as follows respectively.

$$\text{ratio-cut}(C) = \frac{1}{k} \sum_{U_i \in C} \frac{\text{cut}(U_i, \bar{U}_i)}{|U_i|},$$

(69)

$$\text{ncut}(C) = \frac{1}{k} \sum_{U_i \in C} \frac{\text{cut}(U_i)}{\text{vol}(U_i)},$$

(70)

where $|U_i|$ denotes the number of nodes in community $U_i$. As shown in the above example, from the computed costs, we find that the community detected in plot C achieves much lower ratio-cut and ncut costs compared with those in plots B and D. Compared
against the regular cut cost, both ratio-cut and normalized-cut prefer a balanced partition of the social network.

**Spectral Clustering:** Actually the objective function of both ratio-cut and normalized-cut can be unified as the following linear algebra equation

$$
\min_{\mathbf{H} \in \{0, 1\}^{m \times k}} \text{Tr}(\mathbf{H}^T \mathbf{LH}),
$$

where matrix $\mathbf{H} \in \{0, 1\}^{m \times k}$ denotes the communities that users in $\mathcal{V}$ belong to.

Let $\mathbf{A} \in \{0, 1\}^{m \times m}$ denote the social adjacency matrix of the network, and the corresponding diagonal matrix of $\mathbf{A}$ can be represented as matrix $\mathbf{D}$, where $\mathbf{D}$ has value $D(i, i) = \sum_j A(i, j)$ on its diagonal. The Laplacian matrix of the network adjacency matrix $\mathbf{A}$ can be represented as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Depending on the specific measures applied, matrix $\mathbf{L}$ can be represented as

$$
\mathbf{L} = \begin{cases} 
L, & \text{for ratio-cut measure}, \\
\mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}, & \text{for normalized-cut measure}.
\end{cases}
$$

The binary constraint on the variable $\mathbf{H}$ renders the problem non-linear in integer programming problem, which is very hard to solve. One common practice to learn the variable $\mathbf{H}$ is to apply spectral relaxation to replace the binary constraint with the orthogonality constraint.

$$
\min \text{Tr}(\mathbf{H}^T \mathbf{LH}),
$$

$$
\text{s.t. } \mathbf{H}^T \mathbf{H} = \mathbf{I}.
$$

As proposed in [59], the optimal solution $\mathbf{H}^*$ to the above objective function equals to the eigen-vectors corresponding to the $k$ smallest eigen-values of matrix $\mathbf{L}$.

### 6.2 Mutual Community Detection

Besides the knowledge transfer from developed networks to the emerging networks to overcome the cold start problem [95], in the networks in developed networks can also be transferred mutually to help refine the detected community structure detected from each of them. In this section, we will introduce the mutual community detection problem across multiple aligned heterogeneous networks and introduce a new cross-network mutual community detection model $\text{MCD}$. To refine the community structures, a new concept named *discrepancy* is introduced to help preserve the consensus of the community detection result of the shared anchor users according to [97].

For the given two aligned heterogeneous networks $\mathcal{G}(1)$ and $\mathcal{G}(2)$, the **Mutual Community Detection** problem aims to obtain the optimal communities $\mathcal{C}(1) = \{U_1^{(1)}, U_2^{(1)}, \ldots, U_k^{(1)}\}$ and $\mathcal{C}(2) = \{U_1^{(2)}, U_2^{(2)}, \ldots, U_{k'}^{(2)}\}$ of these two networks respectively. Users in each detected social community are more densely connected with each other than with users in other communities. Instead of the propagation based social intimacy score computation among users, $\text{MCD}$ proposes to use the meta paths introduced in Section 3 to utilize both direct and indirect connections among users in closeness scores calculation. With full considerations of the network characteristics, $\text{MCD}$ exploits the information in aligned networks to refine and disambiguate the community structures of the multiple networks concurrently based on a novel concept community discrepancy. More detailed information about the $\text{MCD}$ model will be introduced as follows.

#### 6.2.1 Discrepancy

By maximizing the consensus (i.e., minimizing the “discrepancy”) of the clustering results about the anchor users in multiple partially aligned networks, model $\text{MCD}$ will be able to refine the clustering results of the anchor users with information in other aligned networks mutually. The confidence scores for each user belonging to these communities can be represented as matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$. Matrix $\mathbf{H}^{(1)} = [h_{11}, h_{12}, \ldots, h_{1n}]^T$, $n = |\mathcal{U}|$, $h_{ij} = (h_{i1}, h_{i2}, \ldots, h_{ik})$ and $h_{ij}$ denotes the confidence that $u_i \in \mathcal{U}^{(1)}$ is in cluster $U_j^{(1)} \in \mathcal{C}^{(1)}$. And it is similar for matrix $\mathbf{H}^{(2)}$.

Let $u_i$ and $u_j$ be two anchor users in the network, whose accounts in $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$ are $u_i^{(1)}$, $u_i^{(2)}$, $u_j^{(1)}$ and $u_j^{(2)}$ respectively. If users $u_i^{(1)}$ and $u_j^{(1)}$ are partitioned into the same cluster in $\mathcal{G}^{(1)}$ but their corresponding accounts $u_i^{(2)}$ and $u_j^{(2)}$ are partitioned into different clusters in $\mathcal{G}^{(2)}$, then it will lead to a discrepancy [97; 57] between the clustering results of $u_i^{(1)}$, $u_i^{(2)}$, $u_j^{(1)}$, and $u_j^{(2)}$ in aligned networks $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$.

The discrepancy between the clustering results of $u_i$ and $u_j$ across aligned networks $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$ is defined as the difference of confidence scores of $u_i$ and $u_j$ being partitioned in the same cluster across aligned networks. Considering that in the clustering results, the confidence scores of $u_i^{(1)}$ and $u_j^{(1)}$ ($u_i^{(2)}$ and $u_j^{(2)}$) being partitioned into $k^{(1)}$ ($k^{(2)}$) clusters can be represented as vectors $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ respectively, while the confidences that $u_i$ and $u_j$ are in the same cluster in $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$ can be denoted as $h_i^{(1)}(h_j^{(1)})^T$ and $h_i^{(2)}(h_j^{(2)})^T$. Formally, the discrepancy of the clustering results about $u_i$ and $u_j$ is defined to be $d_{ij}(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) = (h_i^{(1)}(h_j^{(1)})^T - h_i^{(2)}(h_j^{(2)})^T)^2$ if $u_i$, $u_j$ are both anchor users; and $d_{ij}(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) = 0$ otherwise. Furthermore, the discrepancy of $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ will be:

$$
d(c^{(1)}, c^{(2)}) = \sum_{i,j} d_{ij}(c^{(1)}, c^{(2)}),
$$

where $n^{(1)} = |\mathcal{U}^{(1)}|$ and $n^{(2)} = |\mathcal{U}^{(2)}|$. In the definition, non-anchor users are not involved in the discrepancy calculation.

#### 6.2.2 Normalized Discrepancy

However, considering that $d(c^{(1)}, c^{(2)})$ is highly dependent on the number of anchor users and anchor links between $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$, minimizing $d(c^{(1)}, c^{(2)})$ can favor highly consensual clustering results when the anchor users are abundant but have no significant effects when the anchor users are very rare. To solve this problem, model $\text{MCD}$ proposes to minimize the normalized discrepancy instead.

The normalized discrepancy measure computes the differences of clustering results in two aligned networks as a fraction of the discrepancy with regard to the number of anchor users across partially aligned networks:

$$
n_{\text{nd}}(c^{(1)}, c^{(2)}) = \frac{d(c^{(1)}, c^{(2)})}{(|A^{(1,2)}|)(|A^{(1,2)}| - 1)}.
$$

Optimal consensus clustering results of $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$ will be:

$$
c^{(1)}, c^{(2)} = \arg \min_{c^{(1)}, c^{(2)}} n_{\text{nd}}(c^{(1)}, c^{(2)}).
$$

The normalized-discrepancy objective function can also be represented with the clustering results confidence matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ as well. Meanwhile, considering that the networks studied in this section are partially aligned, matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ contain the results of both anchor users and non-anchor users, while
non-anchor users should not be involved in the discrepancy calculation according to the definition of discrepancy. The introduced model proposes to prune the results of the non-anchor users with the anchor transition matrix between the networks. By minimizing the normalized discrepancy together with the community detection cost terms of networks $G^{(1)}$ and $G^{(2)}$, MCD can learn the consensus community structures of multiple social networks mutually.

7. INFORMATION DIFFUSION

Social influence can be widely spread among people, and information exchange has become one of the most important social activities in the real world. The creation of the Internet and online social networks has rapidly facilitated the communication among people. Via the interactions among users in online social networks, information can be propagated from one user to other users. For instance, in recent years, online social networks have become the most important social occasion for news acquisition, and many outstanding social events can get widely spread in the online social networks at a very fast speed. People as the multi-functional “sensors” can detect different kinds of signals happening in the real world, and write posts to report their discoveries to the rest of the world via the online social networks.

In this section, we will study the information diffusion process in the online social networks. Diffusion denotes the spreading process of certain entities (like information, idea, innovation, even heat in physics and disease in bio-medical science) through certain channels among the target object group in a system. The entities to be spread, the channels available, the target object group and the system can all affect the diffusion process and lead to different diffusion observations.

Depending on the system where the diffusion process is originally studied, the diffusion models can be divided into (1) information diffusion models in social networks [29; 103], (2) viral spreading in the bio-medical system [53; 12], and (3) heat diffusion in physical system [47; 8]. We will take the information diffusion in online social networks as one example. The channels for information diffusion belong to certain sources, like online world diffusion channels and offline world diffusion channels, or diffusion channels in different social networks. Meanwhile, depending on the diffusion channels and sources available, the diffusion models include (1) single-channel diffusion model [92; 29], (2) single source multi-channel diffusion model [82], (3) multi-source single-channel diffusion model [81; 77], and (4) multi-source multi-channel diffusion model [78; 103; 79]. Based on the categories of topics to be spread in the online social networks, the diffusion models can be categorized into (1) single topic diffusion [29; 78], (2) multiple intertwined topics concurrent diffusion [103; 92; 34; 13; 7].

In the following part of this section, we will introduce different kinds of diffusion models proposed to depict how information propagates among users in online social networks. We will first talk about the classic diffusion models proposed for the single-network single channel scenario, including the threshold based models, cascades based models, heat diffusion based models and viral diffusion based models. After that, the random walk based cross-network diffusion model will be introduced.

7.1 Traditional Information Diffusion Models

The “diffusion” phenomenon has been observed in different disciplines, like social science, physics, and bio-medical science. Various diffusion models have been proposed in these areas already. In this part, we will provide a brief introduction to these models, and introduce how to apply or adapt them for describe information diffusion process in online social networks.

Let $G = (V, E)$ represent the network structure, based on which we want to study the information diffusion problem. Formally, given a user node $u \in V$, we can represent the set of neighbors of $u$ as $\Gamma(u)$. Each user node in the network $G$ will have an indicator denoting whether the user has been activated or not. We will use notation $s(u) = 1$ to denote that user $u$ has been activated, and $s(u) = 0$ to represent that $u$ is still inactive. Initially, all the users are inactive to a certain information. Information can be propagated from an initial influence seed user set $S \subset V$ who are exposed to and activated by the information at the very beginning. At a timestamp in the diffusion process, given user $u$’s neighbor, we can represent the subset of the active neighbors as $\Gamma^a(u) = \{v | v \in \Gamma(u), s(v) = 1\}$. The set of inactive neighbors can be represented as $\Gamma^i(u) = \Gamma(u) \setminus \Gamma^a(u)$. Generally, the information diffusion process will stop if no new activation is available.

7.1.1 Linear Threshold (LT) Models

In this subsection, we will introduce the threshold models, and will use linear threshold model as an example to illustrate such a kind of models. Several different variants of the linear threshold models will be briefly introduced here as well.

Generally, the threshold models assume that individuals have a unique threshold indicating the minimum amount of required information for them to be activated by certain information. Information can propagate among the users, and the information amount is determined by the closeness of the users. Close friends can influence each other much more than regular friends and strangers. If the information propagated from other users in the network surpass the threshold of a certain user, the user will turn to an activated status and also start to influence other users. Therefore, the threshold values can determine the performance of users in the online social networks. Depending on the setting of the thresholds as well as the amount of information propagated among the users, the threshold models have different variants.

**LT Model:** In the linear threshold (LT) model [29], each user has a unique threshold denoting the minimum required information to activate the user. Formally, the threshold of user $u$ can be represented as $\theta_u \in [0, 1]$. In the simulation experiments, the threshold values are normally selected from the uniform distribution $U(0, 1)$. Meanwhile, for each user pair, like $u, v \in V$, information can be propagated between them. As mentioned before, close friends will have larger influence on each other compared with regular friends and strangers. Formally, the amount of information users $u$ can send to $v$ is denoted as weight $w_{u,v} \in [0, 1]$. Generally, the total amount of informations can send out is bounded. For instance, in the LT model, the total amount of information user $u$ can send out is bounded by $1$, i.e., $\sum_{v \in \Gamma(u)} w_{u,v} \leq 1$. Different ways have been proposed to define the specific value of the weight $w_{u,v}$ value, and in many of the cases $w_{u,v}$ can be different from $w_{v,u}$ since the information each user can send out can be different. However, in many other cases, to simplify the setting, for the same user pair, $w_{u,v}$ and $w_{v,u}$ are usually assigned with the same value. For instance, in some LT models, Jaccard’s Coefficient is applied to calculate the closeness between the user pairs which will be used as the weight value.

In the LT model, the information sent from the neighbors to user $u$ can be aggregated with linear summation. For instance, the total amount of information user $u$ can receive from his/her neighbors can be denoted as $\sum_{v \in \Gamma(u)} w(v, u)s(v)$ or $\sum_{v \in \Gamma^a(u)} w(v, u)$. To check whether a user can be activated or not, LT model will only need to check whether the following equation holds or not,

$$\sum_{v \in \Gamma^a(u)} w(v, u) \geq \theta_u. \tag{78}$$
It denotes whether the received information surpasses the activation threshold of user \( u \) or not. Here, we also need to notice that inactive neighbors will not send out information, and only the active neighbors can send out information. The information provided so far shows the critical details of the LT model. Next, we will show the general framework of the LT model to illustrate how it works. In the LT model, the initial activated seed user set can be represented as \( S \), users in which can start the propagation of information to their neighbors. Generally, information propagates within the network step by step.

- **Diffusion Starts**: At step 0, only the seed users in \( S \) are active, and all the remaining users have inactive status.

- **Diffusion Spreads**: At step \( t (t > 0) \), for each user \( u \), if the information propagated from \( u \)'s active neighbors is greater than the threshold of \( u \), i.e., \( \sum_{v \in N(u)} w(v, u) \geq \theta_u \), \( u \) will be activated with status \( s(u) = 1 \). All activated users will remain active in the coming rounds, and can send information to their neighbors. Active user cannot be activated again.

- **Diffusion Ends**: If no new activation happens in step \( t \), the diffusion process will stop.

**Other Threshold Models**: The LT model assumes the cumulative effects of information propagated from the neighbors, and it can illustrate the basic information diffusion process among users in the online social networks. The LT model has been well analyzed, and many other variant models have been proposed as well. Depending on the assignment of the threshold and weight values, many other different diffusion models can all be reduced to a special case of the LT model.

- **Majority Threshold Model**: Different from the LT model, in **majority threshold model** [10], an inactive user \( u \) can be activated if majority of his/her neighbors are activated. The **majority threshold model** can be reduced to the LT model in the case that: (1) the influence weight between any friends \((u, v)\) in the network is assigned with value 1; (2) the threshold of any user \( u \) is set as \( \frac{1}{2} \theta_u \), where \( \theta_u \) denotes the degree of node \( u \) in the network. For the nodes with large degrees, like the central node in the star-structured diagram, their activation will lead to the activation of lots of surrounding nodes in the network.

- **k-Threshold Model**: Another diffusion model similar to the LT model is called the **k-threshold diffusion model** [10], in which users can be activated of at least \( k \) of his/her neighbors are active. The **k-threshold model** is equivalent to the LT model with settings (1) the influence weight between any friend pairs \( (u, v) \) in the network is assigned with value 1; and (2) the activation thresholds of all the users are assigned with a shared value \( k \). For each user \( u \), if \( k \) of his/her neighbors have been activated, \( u \) will be activated.

Depending on the values of \( k \), the **k-threshold model** will have different performance. When \( k = 1 \), a user will be activated of at least one of his/her neighbor is active. In such a case, all the users in the same connected components with the initial seed users will be activated finally. When \( k \) is a very large value and even greater than the large node degree, e.g., \( k > \max_{u \in V} D(u) \), no nodes can be activated. When \( k \) is a medium value, some of the users will be activated as the information propagates, but the other users with less than \( k \) neighbors will never be activated.

### 7.1.2 Independent Cascade (IC) Model

An information cascade occurs when a people observe the actions of others and then engage in the same acts. Cascade clearly illustrates the information propagation routes, and the activating actions performed for users to their neighbors. In this part, we will talk about the cascade based models and use the **independent cascade** (IC) model as an example to illustrate the model architecture.

**IC Model**: In the diffusion process, about one certain target user, multiple activation trials can be performed by his/her neighbors. In the **independent cascade** model [29], each activation is performed independently regardless of the historical unsuccessful trials. The activation trials are performed step by step. When user \( u \) who has been activated in the previous step and tries to activate user \( v \) in the current step, the success probability is denoted as \( p_{u,v} \in [0, 1] \).

Generally, if users \( u \) and \( v \) are close friends, the activation probability will be larger compared with regular friends and strangers. The specific activation probability values is usually correlated with the social closeness between users \( u \) and \( v \), which can also be defined based the Jaccard’s Coefficient in the simulation. The activation trials will only happen among the users who are friends. If \( u \) succeeds in activating \( v \), then user \( v \) will change his/her status to “active” and will remain in the status in the following steps. However, if \( u \) fails to activate \( v \), \( v \) will lose the chance and cannot perform the activation trials any more.

In IC model, the activation trials are performed by flipping a coin with certain probabilities, whose result is uncertain. Even with the same provided initial seed user set \( S \), the number of users who will be activated by the seed users can be different if we running the IC model twice. Formally, we can represent the set of activated users by the seed users as \( V^a \subset V \). Therefore, in the experimental simulations, we usually run the diffusion model multiple times and calculate the average number of activated users, i.e., \( |V^a| \), to denote the expected influence achieved by the seed user set \( S \).

**Other Cascade Models**: Generally, the independent activation assumption renders the IC model the simplest cascade based diffusion models. In the real world, the diffusion process will be more complicated. For the users, who have been failed to be activated by many other users, it probably indicates that the user is not interested in the information. Viewed in such a perspective, the probability for the user to be activated will decrease as more activation trials have been performed. In this part, we will introduce another cascade based diffusion model, **decreasing cascade model** (DC) [30]. To illustrate the DC model more clearly and show it difference compared with the IC model, we use notation \( P(u \rightarrow v|T) \) to represent the probability for user \( u \) to activate \( v \) given a set of users \( T \) have performed and failed the activation trials to \( v \) already. Let \( T, T' \) denote two historical activation trial user set, where \( T \subseteq T' \). In the IC model, we have

\[
P(u \rightarrow v|T) = P(u \rightarrow v|T') .
\]  

(79)

In other words, every activation trial is independent with each other, and the activation probability will not be changed as more activation trials have been performed.

As introduced at the beginning of this subsection, the fact that users in set \( T \) fail to activate \( v \) indicates that \( v \) probably is not interested in the information, and the change for \( v \) to be activated afterwards will be lower. Furthermore, as more activation trials, e.g., users in \( T' \) are performed, the probability for \( u \) to active \( v \) will be decreased, i.e.,

\[
P(u \rightarrow v|T) \geq P(u \rightarrow v|T').
\]  

(80)

Intuitively, this restriction states that a contagious node’s probabil-
ity of activating some $v$ decreases if more nodes have already attempted to activate $v$, and $v$ is hence more “marketing-saturated”. The DC model incorporates the IC model as a special case, and is more general in information diffusion modeling than the IC model.

### 7.1.3 Epidemic Diffusion Model

The threshold and cascade based diffusion models introduced in the previous part mostly assume that “once a user is activated, he/she will remain the active status forever”. However, in the real world, these activated users can change their minds and the activated users can still have the chance to recover to the original status. In the bio-medical science, diffusion models have been studied for many years to model the spread of disease, and several epidemic diffusion models have been introduced already. In the disease propagation, people who are susceptible to the disease can be get infected by other people. After some time, many of these infected people can get recovered and become immune to the disease, while many other users can get recovered and get susceptible to the disease again. Depending on the people’s reactions to the disease after recovery, several different epidemic diffusion models [51] have been proposed already.

**Susceptible-Infected-Recovered (SIR) Diffusion Model:** The SIR model was proposed by W. O. Kermack and A. G. McKendrick in 1927 to model the infectious diseases, which consider a fixed population with three main categories: susceptible (S), infected (I), and recovered (R). As the disease propagates, the individual status can change among {S, I, R} following flow:

$$S \rightarrow I \rightarrow R.$$

In other words, the individuals who are susceptible to the disease can get infected, while those infected individuals also have the chance to recover from the disease as well. In this part, we will use the SIR model to describe the information cascading process in online social networks. Let $\mathcal{V}$ denote the set of users in the network. We introduce the following notations to represent the number of users in different categories:

- $S(t)$: the number of users who are susceptible to the information at time $t$, but have not yet gotten infected.
- $I(t)$: the number of users who are currently infected by the information, and can spread the information to others in the susceptible category.
- $R(t)$: the number of users who have been infected and already recovered from the information infection. The users are immune to the information and will not be infected again.

Based on the above notations, we have the following equations hold in the SIR model.

$$S(t) + I(t) + R(t) = |\mathcal{V}|,$$

$$\frac{dS(t)}{dt} + \frac{dI(t)}{dt} + \frac{dR(t)}{dt} = 0,$$

where,

$$\frac{dS(t)}{dt} = -\beta S(t) I(t),$$

$$\frac{dI(t)}{dt} = \beta S(t) I(t) - \gamma I(t),$$

$$\frac{dR(t)}{dt} = \gamma I(t).$$

#### Other Epidemic Diffusion Model

In some cases, the users cannot get immune to the information and don’t exist the recovery status actually. For the users, who get infected, they can go to the susceptible status and can get infected again in the future. To model such a phenomenon, another diffusion model very similar to the SIR model has been proposed, which is called the Susceptible-Infected-Susceptible (SIS) model. Furthermore, in another variant of SIR, the individuals in the recovery category can lose the immunity and transit to the susceptible category and have the potential to be infected again. Therefore, the individual status flow will be

$$S \rightarrow I \rightarrow R \rightarrow S.$$  \hspace{1cm} (85)

And the model is also named as the SIRS diffusion model.

Besides these epidemic diffusion models introduced in this subsection, there also exist many different version of the epidemic diffusion models, which considers many other factors in the diffusion process, like the birth/death of individuals. It is also very common in the real-world online social networks, since new users will join in the social network, and existing users will also delete their account and get removed from the social network. Involving such factors will make the diffusion model more complex, and we will not introduce them here due to the limited space. More information about these epidemic diffusion models is available in [49; 51].

### 7.1.4 Heat Diffusion Models

Heat diffusion is a well observed physical phenomenon. Generally, in a medium, heat will always diffuses from regions with a high temperature to the region with a lower temperature. Recently, many works have applied the heat diffusion to model the information propagation in online social networks. In this subsection, we will talk about the heat diffusion model and introduce how to adapt it to model the information diffusion in online social networks.

**General Heat Diffusion:** Throughout a geometric manifold, let function $f(x, t)$ denote the temperature at location $x$ at time $t$, and we can represent the initial temperature at different locations as $f_0(x)$. The heat flows with initial conditions can be described by the following second order differential equation

$$\frac{\partial f(x, t)}{\partial t} - \Delta f(x, t) = 0$$

$$f(x, 0) = f_0(x),$$

where $\Delta f(x, t)$ is a Laplace-Beltrami operator on function $f(x, t)$.

Many existing works on the heat diffusion studies are mainly focused on the heat kernel matrix. Formally, let $K_t$ denote the heat kernel matrix at timestamp $t$, which describes the heat diffusion among different regions in the medium. In the matrix, entry $K_t(x, y)$ denotes the heat diffused from the original position $y$ to position $x$ at time $t$. However, it is very difficult to represent the medium as a regular geometry with a known dimension. In the next part, we will introduce how to apply the heat diffusion observations to model the information diffusion in the network-structured graph data.

**Heat Diffusion Model:** Given a homogeneous network $G = (\mathcal{V}, \mathcal{E})$, for each node $u \in \mathcal{V}$ in the network, we can represent the information at $u$ in timestamp $t$ as $f(u, t)$. The initial information available at each of the node can be denoted as $f(u, 0)$. The information can be propagated among the nodes in the network if there exists a pipe (i.e., a link) between them. For instance, with a link $(u, v) \in \mathcal{E}$ in the network, information can be propagated between $u$ and $v$.

Generally, in the diffusion process, the amount of information propagated between different nodes in the network depends on (1) the difference of information available at them, and (2) the thermal conductivity—the heat diffusion coefficient $\alpha$. For instance, at timestamp $t$, we can represent the amount of information reaching nodes $u, v \in \mathcal{V}$ as $f(u, t)$ and $f(v, t)$. If $f(u, t) > f(v, t)$, information tends to propagate from $u$ to $v$ in the network, and the amount of information propagated is $\alpha \cdot (f(u, t) - f(v, t))$, and the propagation direction will be reversed if $f(u, t) < f(v, t)$. The information amount changes at node $u$ at timestamps $t$ and $t + \Delta t$ can be
represented as
\[
\frac{f(u, t + \Delta t) - f(u, t)}{\Delta t} = - \sum_{v \in \mathcal{G}(u)} \alpha \cdot (f(u, t) - f(v, t)).
\]

(87)

Let’s use vector \(f(t)\) to represent the amount of information available at all the nodes in the network at timestamp \(t\). The above information amount changes can be rewritten as
\[
\frac{f(t + \Delta t) - f(t)}{\Delta t} = \alpha \mathbf{H} f(t),
\]

(88)

where in the matrix \(\mathbf{H} \in \mathbb{R}^{|V| \times |V|}\), entry \(H(u, v)\) has value
\[
H(u, v) = \begin{cases} 
1, & \text{if } (u, v) \in \mathcal{E} \lor (v, u) \in \mathcal{E}, \\
-D(u), & \text{if } u = v, \\
0, & \text{otherwise,}
\end{cases}
\]

(89)

where \(D(u)\) denotes the degree of node \(u\) in the network. In the limit case \(\Delta t \to 0\), we can rewrite the equation as
\[
\frac{df(t)}{dt} = \alpha \mathbf{H} f(t).
\]

(90)

Solving the function, we can represent the amount of information at each node in the network as
\[
f(t) = \exp^{\alpha t \mathbf{H}} f(0)
\]

(91)

\[
= \left(1 + \alpha t \mathbf{H} + \frac{\alpha^2 t^2}{2!} \mathbf{H}^2 + \frac{\alpha^3 t^3}{3!} \mathbf{H}^3 + \cdots\right) f(0),
\]

(92)

where term \(\exp^{\alpha t \mathbf{H}}\) is called the diffusion kernel matrix, which can be expanded according to Taylor’s theorem.

### 7.2 Random Walk based Diffusion Model

Different online social networks usually have their own characteristics, and users tend to have different statuses regarding the same information. For instance, information about personal entertainments (like movies, pop stars) can be widely spread among users in Facebook, and users interested in them will be activated very easily and also share the information to their friends. However, such a kind of information is relatively rare in the professional social network LinkedIn, where people seldom share personal entertainment with their colleagues, even though they may have been activated already in Facebook. What’s more, the structures of these online social networks are usually heterogeneous, containing many different kinds of connections. Besides the direct follow relationships among the users, these diverse connections available among the users may create different types of communication channels for information diffusion. To model such an observation in information diffusion across multiple heterogeneous online social sites, in this part, we will introduce a new information diffusion model, IPATH [80], based on random walk.

#### 7.2.1 Intra-Network Propagation

In a heterogeneous network, multi-typed and interconnected entities, such as images, videos and locations, can create various information propagation relations among users. We can represent the information diffusion routes among users via other information entities, which can be formally represented as the diffusion route set \(\mathcal{R} = \{r_1, r_2, \ldots, r_m\}\), where \(m\) is the route number. Let’s take the source network \(\mathcal{G}(s) = (\mathcal{V}(s), \mathcal{E}(s))\) as an example. For any diffusion route \(r_i \in \mathcal{R}\), the adjacency matrix of \(r_i\) will be \(A^{(s)}(u, v)\), where \(A^{(s)}(u, v)\) is a binary-value variable and \(A^{(s)}(u, v) = 1\) iff \(u\) and \(v\) are connected with each other via relation \(r_i\). The weighted diffusion matrix can be represented as

\[
W = |\mathcal{R}| \sum_{r_i \in \mathcal{R}} w_i A^{(s)}(u, v)
\]

\[
\text{where } w_i \text{ is the weight of relation } r_i.
\]

7.2.2 Inter-Network Propagation

Across the aligned networks, information can propagate not only within networks but also across networks. Based on the known anchor links between networks \(\mathcal{G}^{(s)}\) and \(\mathcal{G}^{(t)}\), i.e., set \(A^{(s,t)}\), we can define the binary adjacency matrix \(A^{(s,t)}(u, v) = 1\) iff \((u, v) \in \mathcal{E}^{(s,t)}\). IPATH assumes that each anchor user in \(\mathcal{G}^{(s)}\) only has one corresponding account in \(\mathcal{G}^{(t)}\). Therefore \(A^{(s,t)}\) has been normalized and the weight matrix \(W^{(s,t)}\) has been defined to denote the chance of information propagating from \(\mathcal{G}^{(s)}\) to \(\mathcal{G}^{(t)}\). Furthermore, we can represent the weighted diffusion matrix from networks \(\mathcal{G}^{(s)}\) to \(\mathcal{G}^{(t)}\) as \(W^{(t,s)} = (W^{(s,t)})^T\), considering that the anchor links are undirected.

Both the intra-network propagation relations, represented by weight matrices \(W^{(s)}\) and \(W^{(t)}\) in networks \(\mathcal{G}^{(s)}\) and \(\mathcal{G}^{(t)}\) respectively, and the inter-network propagation relations, represented by weight matrix \(W^{(s,t)}\) and \(W^{(t,s)}\) have been constructed already in the previous subsection. As shown in Figure 3, to model the cross-network information diffusion process involving both the intra- and inter-network relations simultaneously, IPATH proposes to combine these weighted diffusion matrices to build an integrated matrix \(\mathbf{W} \in \mathbb{R}^{(|\mathcal{V}^{(s)}| + |\mathcal{V}^{(t)}|)^2}\). In the integrated matrix \(\mathbf{W}\), the parameter \(\alpha \in [0, 1]\) denotes the probability that the message stays in the original network, thus \(1 - \alpha\) represents the chance of being transmitted across networks (i.e., the probability of activated anchor user passing the influence to the target network).

#### 7.2.3 The IPATH Information Propagation Model

Let vector \(\pi_k \in \mathbb{R}^{(|\mathcal{V}^{(s)}| + |\mathcal{V}^{(t)}|)}\) represent the information that users in \(\mathcal{G}^{(s)}\) and \(\mathcal{G}^{(t)}\) can receive after \(k\) steps. As shown in Figure 3, vector \(\pi_k\) consists of two parts \(\pi_k = [\pi_k^{(s)}, \pi_k^{(t)}]\), where \(\pi_k^{(s)} \in \mathbb{R}^{(|\mathcal{V}^{(s)}|)}\) and \(\pi_k^{(t)} \in \mathbb{R}^{(|\mathcal{V}^{(t)}|)}\). The initial state of the vector can be denoted as \(\pi_0\), which is defined based on the seed user set \(\mathcal{Z}\) with function \(g(\cdot)\) as follows:

\[
\pi_0 = g(\mathcal{Z}), \text{ where } \pi_0[u] = \begin{cases} 1 & \text{if } u \in \mathcal{Z}, \\ 0 & \text{otherwise}. \end{cases}
\]

(94)

Seed set \(\mathcal{Z}\) can also be represented as \(\mathcal{Z} = g^{-1}(\pi_0)\). Users from \(\mathcal{G}^{(s)}\) and \(\mathcal{G}^{(t)}\) both have the chance of being selected as seeds, but when the structure information of \(\mathcal{G}^{(t)}\) is hard to obtain, the seed
users will be only chosen from \( G^{(k)} \). In IPATH, the information diffusion process is modeled by random walk, because it is widely used in which the total probability of the diffusing through different relations remains constant \( I \) \cite{68,20}. Therefore, in the information propagation process, vector \( \pi \) will be updated stepwise with the following equation:

\[
\pi^{(k+1)} = (1 - \alpha) \times W \pi_k + \alpha \times \pi_0,
\]

where constant \( \alpha \) denotes the probability of returning to the initial state. By keeping updating \( \pi \) according to (95) until convergence, we can present the stationary state of vector \( \pi \) to be \( \pi^* \),

\[
\pi^* = \alpha(I - (1 - \alpha)W)^{-1} \pi_0,
\]

where matrix \( I - (1 - \alpha)W \) is an identity matrix. The value of entry \( \pi^*[u] \) denotes the activation probability of \( u \), and user \( u \) will be activated if \( \pi^*[u] \geq \theta \), where \( \theta \) denotes the threshold of accepting the message. In IPATH, parameter \( \theta \) is randomly sampled from range \([0, \theta_{\text{bound}}] \). The threshold bound \( \theta_{\text{bound}} \) is a small constant value, as the amount of information each user can get at the stationary state in IPATH can be very small (which is set as 0.01 in the experiments). In addition, we can further represent the activation status of user \( u \) as vector \( \pi' \), where

\[
\pi'[u] = \begin{cases} 1 & \text{if } \pi^*[u] \geq \theta, \\ 0 & \text{otherwise}. \end{cases}
\]

**8. NETWORK EMBEDDING**

In the concrete applications, great challenges exist in handling the network structured data with traditional machine learning algorithms, which usually take feature vector representation data as the input. A general representation of heterogeneous networks as feature vectors is desired for knowledge discovery from such complex network structured data. In recent years, many research works propose to embed the online social network data into a low-dimensional feature space, in which the user node is represented as a unique feature vector, and the network structure can be reconstructed from these feature vectors. With the embedded feature vectors, classic machine learning models can be applied to deal with the social network data directly, and the storage space can be saved greatly.

In this section, we will talk about the network embedding problem, aiming at projecting the nodes and links in the network data in low-dimensional feature spaces. Depending on the application setting, existing graph embedding works can be categorized into the embedding of homogeneous networks, heterogeneous networks, and multiple aligned heterogeneous networks. Meanwhile, depending on the models being applied, current embedding works can be divided into the matrix factorization based embedding, translation based embedding, and deep learning based embedding.

In the following parts in this section, we will first introduce the translation based graph embedding models in Section 8.1, which are mainly proposed for the multi-relational knowledge graphs, including TransE \cite{9}, TransH \cite{73} and TransR \cite{40}. After that, in Section 8.2, we will introduce three homogeneous network embedding models, including DeepWalk \cite{52}, LINE \cite{67} and node2vec \cite{22}. Finally, we will talk about the model proposed for the multiple aligned heterogeneous network \cite{93} in Section 8.3, where the anchor links are utilized to transfer information across different sites for mutual refinement of the embedding results mutually.

**8.1 Translation based Network Embedding**

Multi-relational data refers to the directed graphs whose nodes correspond to entities and links denote the relationships. The multi-relational data can be represented as a graph \( G = (V, E) \), where \( V \) denotes the node set and \( E \) represents the link set. For the link in the graph, e.g., \( r = (h, t) \in E \), the corresponding entity-relation can be represented as a triple \( (h, r, t) \), where \( h \) denotes the link initiator entity, \( t \) denotes the link recipient entity and \( r \) represents the link. The embedding problem studied in this section is to learn a feature representation of both entities and relations in the triples, i.e., \( h, r \) and \( t \).

### 8.1.1 TransE

The TransE \cite{9} model is an energy-based model for learning low-dimensional embeddings of entities and relations, where the relations are represented as the translations of entities in the embedding space. Given a entity-relation triple \( (h, r, t) \), the embedding feature representation of the entities and relations can be represented as vectors \( h \in \mathbb{R}^k, r \in \mathbb{R}^k \) and \( t \in \mathbb{R}^k \) (\( k \) denotes the objective vector dimension). If the triple \((h, r, t)\) holds, i.e., there exists a link \( r \) starting from \( h \) to \( t \) in the network, the corresponding embedding vectors \( h + r \) should be as close to vector \( t \) as possible.

Let \( S^+ = \{(h, r, t) \mid r = (h, t) \in E \} \) represents the set of positive training data, which contains the triples existing in the networks. The TransE model aims at learning the embedding features vectors of the entities \( h \) and the relation \( r \), i.e., \( h, r \) and \( t \). For the triples in the positive training set, we want to ensure the learnt embedding vectors \( h + r \) is very close to \( t \). Let \( d(h + r, t) \) denotes the distance between vectors \( h + r \) and \( t \). The loss introduced for the triples in the positive training set can be represented as

\[
\mathcal{L}(S^+) = \sum_{(h, r, t) \in S^+} d(h + r, t).
\]

Here the function distance can be defined in different ways, like the \( L_2 \) norm of the difference between vectors \( h + r \) and \( t \), i.e.,

\[
d(h + r, t) = \|h + r - t\|_2.
\]

By minimizing the above loss function, the optimal feature representations of the entities and relations can be learnt. To avoid trivial solutions, like \( 0s \) for \( h \) and \( t \), additional constraints that the \( L_2 \)-norm of the embedding vectors of the entities should be 1 will be added in the function. Furthermore, a negative training set is also sampled to differentiate the learnt embedding vectors. For a triple \((h, r, t) \in S^+\), the corresponding sampled negative training set can be denoted as \( S^-(h, r, t) \) which contains the triples formed by replacing the initiator entity \( h \) or the recipient entity \( t \) with random entities. In other words, the negative training set \( S^-(h, r, t) \) can be represented as

\[
S^-(h, r, t) = \{(h', r, t) \mid h' \in V \} \cup \{(h, r', t) \mid r' \in V \}.
\]

The loss function involving both the positive and negative training set can be represented as

\[
\mathcal{L}(S^+, S^-) = \sum_{(h, r, t) \in S^+} \sum_{(h', r', t') \in S^-(h, r, t)} \max(0, \gamma + d(h + r, t) - d(h' + r', t'), 0),
\]

where \( \gamma \) is a margin hyperparameter and \( \max(\cdot, 0) \) will count the positive loss only.

The optimization is carried out by stochastic gradient descent (in minibatch mode). The embedding vectors of entities and relationships are initialized with a random procedure. At each iteration of the algorithm, the embedding vectors of the entities are normalized and a small set of triplets is sampled from the training set, which will serve as the training triplets of the minibatch. The parameters are then updated by taking a gradient step.
8.1.2 TransH

TransE is a promising method proposed recently, which is very efficient while achieving state-of-the-art predictive performance. However, in the embedding process, TransE fail to consider the cardinality constraint on the relations, like one-to-one, one-to-many and many-to-many. The TransH model [73] to be introduced in this part considers such properties on relations in the embedding process. Different from the other complex models, which can handle these properties but sacrifice efficiency, TransH achieves comparable time complexity as TransE. TransH models the relation as a hyperplane together with a translation operation on it, where the correlation among the entities can be effectively preserved.

In TransH, different from the embedding space of entities, the relations, e.g., r, is denoted as a transition vector d_r in the hyperplane w_r (a normal vector). For each of the triple (h, r, t), the embedding vector h, t are first projected to the hyperplane w_r, whose corresponding projected vectors can be represented as h_{\perp} and t_{\perp} respectively. The vectors h_{\perp} and t_{\perp} can be connected by the translation vector d_r on the hyperplane. Depending on whether the triple appears in the positive or negative training set, the distance d(h_{\perp} + d_r, t_{\perp}) should be either minimized or maximized.

Formally, given the hyperplane w_r, the projection vectors h_{\perp} and t_{\perp} can be represented as

\[ h_{\perp} = h - w_r^T h w_r, \]
\[ t_{\perp} = t - w_r^T t w_r. \]

Furthermore, the L_2 norm based distance function can be represented as

\[ d(h_{\perp} + d_r, t_{\perp}) = \| (h - w_r h w_r) + d_r - (t - w_r t w_r) \|_2^2. \]

The variables to be learnt in the TransH model include the embedding vectors of all the entities, the hyperplane and translation vectors for each of the relations. To learn these variables simultaneously, the objective function of TransH can be represented as

\[ \mathcal{L}(S^+, S^-) = \sum_{(h, r, t) \in S^+} \sum_{(h', r', t') \in S^-} \max \left( \gamma + d(h_{\perp} + d_r, t_{\perp}) - d(h'_{\perp} + d_r', t'_{\perp}), 0 \right), \]

where \( S_{(h, r, t)} \) denotes the negative set constructed for triple (h, r, t). Different from TransE, TransH applies a different to sample the negative training triples with considerations of the relation cardinality constraint. For the relations with one-to-many, TransH will give more chance to replace the initiator node; and for the many-to-one relations, TransH will give more chance to replace the recipient node instead.

Besides the loss function, the variables to be learnt are subject to some constraints, like the embedding vector for entities is a normal vector; w_r and d_r should be orthogonal, and w_r is also a normal vector. We summarize the constraints of the TransH model as follows

\[ \| h \|_2 \leq 1, \| t \|_2 \leq 1, \forall h, t \in \mathcal{V}, \]
\[ \| w_r^T d_r \|_2 \leq 1, \forall r \in \mathcal{E}, \]
\[ \| w_r \|_2 \leq 1, \forall r \in \mathcal{E}. \]

The constraints can be relaxed as some penalty terms, which can be added to the objective function with a relatively large weight. The final objective function can be learnt with the stochastic gradient descent, and by minimizing the loss function, the model variables can be learned and we will get the final embedding results.

8.1.3 TransR

Both TransE and TransH introduced in the previous subsections assume embeddings of entities and relations within the same space \( \mathbb{R}^k \). However, entities and relations are actually totally different objects, and they may be not capable to be represented in a common semantic space. To address such a problem, TransR [40] is proposed, which models the entities and relations in distinct spaces, i.e., the entity space and relation space, and performs the translation in relation space.

In TransR, given a triple (h, r, t), the entities h and t are embedded as vectors h, t ∈ \( \mathbb{R}^k_h \), and the relation r is embedded as vector r ∈ \( \mathbb{R}^k_r \), where the dimension of the entity space and relation space are not the same, i.e., \( k_h \neq k_r \). To project the entities from the entity space to the relation space, a projection matrix \( M_r \in \mathbb{R}^{k_h \times k_r} \) is defined in TransR. With the projection matrix, the projected entity embedding vectors can be defined as

\[ h_r = h M_r, \]
\[ t_r = t M_r. \]

The loss function is defined as

\[ d(h_r + r, t_r) = \| h_r + r - t_r \|_2^2. \]

The constraints involved in TransR include

\[ \| h \|_2 = 1, \| t \|_2 = 1, \forall h, t \in \mathcal{V}, \]
\[ \| h M_r \|_2 = 1, \| t M_r \|_2 = 1, \forall h, t \in \mathcal{V}, \]
\[ \| w_r \|_2 \leq 1, \forall r \in \mathcal{E}. \]

The nonnegative training set \( S^+ \) in TransR can be obtained in a similar way as TransH, where the variables can be learnt with the stochastic gradient descent. We will not introduce the information here to avoid content duplication.

8.2 Homogeneous Network Embedding

Besides the translation based network embedding models, in this section, we will introduce three embedding models for network data, including DeepWalk, LINE and node2vec. Formally, the networks studied in this part are all homogeneous networks, which is represented as \( G = (\mathcal{V}, \mathcal{E}) \). Set \( \mathcal{V} \) denotes the set of nodes in the homogeneous network, and \( \mathcal{E} \) represents the set of links among the nodes inside the network.

8.2.1 DeepWalk

The DeepWalk [52] algorithm consists of two main components: (1) a random walk generator, and (2) an update procedure. In the first step, the DeepWalk model randomly selects a node, e.g., \( u \in \mathcal{V} \), as the root of a random walk \( W_k \) from the nodes in the network. Random walk \( W_k \) will sample the neighbors of the node last visited uniformly until the maximum length \( l \) is met. In the second step, the sampled neighbors are used to update the representations of the nodes inside the graph, where SkipGram [46] is applied here.

Random Walk Generator: The random walk model has been introduced in Section 5.1.1. Formally, the random walk starting at node \( u \in \mathcal{V} \) can be represented as \( W_u \), which actually denotes a stochastic process with random status \( W_u^0, W_u^1, \ldots, W_u^k \). Formally, at the very beginning, i.e., step 0, the random walk is at the initial node, i.e., \( W_u^0 = u \). The status variable \( W_u^k \) denotes the node where the node is at step \( k \).

Random walk can capture the local network structures effectively, where the neighborhood and social connection closeness can affect the next nodes that the random walk will move to in the next step. Therefore, in the DeepWalk, random walk is applied to sample a
stream of short random walks as the tool for extracting information from a network. Random walk can provide two very desirable properties, besides the ability to capture the local community structures. Firstly, the random walk based local exploration is easy to parallelize. Several random walks can simultaneously explore different parts of the same network in different threads, processes and machines. Secondly, with the information obtained from short random walks, it is possible to accommodate small changes in the network structure without the need for global recomputation. 

**SkipGram Technique:** The updating procedure used in DeepWalk is very similar to the word appearance prediction in language modeling. SkipGram is a language model that maximize the co-occurrence probability of words appearing in the time window $s$ in a sentence. Here, when applying the SkipGram technique to the DeepWalk model, the nodes $u \in V$ in the network can be regarded as the words $w$ denoted in the equations aforementioned. Meanwhile, for the nodes sampled by the random walk model within the window size $s$ before and after node $v$, they will be treated as the words appearing ahead of and after node $v$. Furthermore, SkipGram assumes the appearance of the words (or nodes for networks) to be independent, and the above probability equations can be rewritten as follows:

$$P(u_{n-x}, u_{n-x+1}, \ldots, u_{n+1} \setminus \{u_n\}|x_{u_{n}}) = \prod_{i=n-x}^{n+s} P(u_i|x_{u_{n}}), \quad \text{(117)}$$

where $u_{n-x}, u_{n-x+1}, \ldots, u_{n+s}$ denotes the sequence of nodes sampled by the random walk model. The learning process of the SkipGram algorithm is provided in Algorithm 2, where we will enumerate all the co-locations of nodes in the sampled node series $v_{n-x}, v_{n-x+1}, \ldots, v_{n+s}$ by a random walk $W_n$ (starting from node $u$ in the network). With gradient descent, the representation of nodes with their neighbors representations can be updated with stochastic gradient descent. The derivatives are estimated with the back-propagation algorithm. However, in the equation, we need to have the conditional probabilities of the nodes and their representations. A concrete representation of the probability can be a great challenging problem. As proposed in [46], such a distribution can be learnt with some existing models, like logistic regression. However, since the labels used here denote the nodes in the network, it will lead to a very large label space with $|V|$ different labels, which renders the learning process extremely time consuming. To solve such a problem, some techniques, like Hierarchical Softmax, have been proposed which represents the nodes in the network as a binary tree and can lower done the probability computation time complexity from $O(|V|^2)$ to $O(\log |V|)$.

**Hierarchical Softmax:** In the SkipGram algorithm, calculating probability $P(u|x_{u_{n}})$ is infeasible. Therefore, in the DeepWalk model, hierarchical softmax is used to factorize the conditional probability. In hierarchical softmax, a binary tree is constructed, where the number of leaves equals to the network node set size, and each network node is assigned to a leaf node. The prediction problem is turned into a path probability maximization problem. If a path $(b_0, b_1, \ldots, b_{\log|V|})$ is identified from the tree root to the node $u$, i.e., $b_0 = \text{root}$ and $b_{\log|V|} = u$, then the probability can be rewritten as

$$P(u|x_{u_{n}}) = \prod_{l=1}^{\log|V|} P(b_l|x_{u_{n}}), \quad \text{(118)}$$

where $P(b_l|x_{u_{n}})$ can be model by a binary classifier denoted as

$$P(b_l|x_{u_{n}}) = \frac{1}{1 + \exp^{-x_{b_l}x_{u_{n}}}}. \quad \text{(119)}$$

Here the parameters involved in the learning process include the representations for both the nodes in the network as well as the nodes in the constructed binary trees.

### 8.2.2 LINE

To handle the real-world information networks, the embedding models need to have several requirements: (1) preserve the first-order and second-order proximity between the nodes, (2) scalable to large sized networks, and (3) able to handle networks with different links: directed and undirected, weighted and unweighted. In this part, we will introduce another homogeneous network embedding model, named LINE [67].

**First-order Proximity:** In the network embedding process, the network structure should be effectively preserved, where the node closeness is defined as the node proximity concept in LINE. The first-order proximity in a network denotes the local pairwise proximity between nodes. For a link $(u, v) \in E$ in the network, the first-order proximity denotes the weight of link $(u, v)$ in the network (or 1 if the network is unweighted). Meanwhile, if link $(u, v)$ doesn’t exist in the network, the first-order proximity between them will be 0 instead. To model the first-order proximity, for a given link $(u, v) \in E$ in the network $G$, LINE defines the joint probability between nodes $u$ and $v$ as

$$p_1(u, v) = \frac{1}{1 + \exp^{-x_u \cdot x_v}}, \quad \text{(120)}$$

where $x_u, x_v \in \mathbb{R}^d$ denote the vector representations of nodes $u$ and $v$ respectively.

Function $p_1(\cdot, \cdot)$ defines the proximity distribution in the space of $V \times V$. Meanwhile, given a network $G$, the empirical proximity between nodes $u$ and $v$ can be denoted as

$$\hat{p}_1(u, v) = \frac{w(u, v)}{\sum_{(v, u) \in E} w(v, u)} \cdot \log p_1(u, v), \quad \text{(121)}$$

To preserve the first-order proximity, LINE defines the objective function for the network embedding as

$$J_1 = d(p_1(\cdot, \cdot), \hat{p}_1(\cdot, \cdot)), \quad \text{(122)}$$

where function $d(\cdot, \cdot)$ denotes the distance between two distributions introduced proximity distribution and the empirical proximity distribution. By replacing the distance function $d(\cdot, \cdot)$ with the KL-divergence and omitting some constants, the objective function can be rewritten as

$$J_1 = - \sum_{(u, v) \in E} w(u, v) \log p_1(u, v). \quad \text{(123)}$$

By minimizing the objective function, LINE can learn the feature representation $x_u$ for each node $u \in V$ in the network.

**Second-order Proximity:** In the real-world social networks, the links among the nodes can be very sparse, where the first-order proximity can hardly preserve the complete structure information of the network. LINE introduce the concept of second-order proximity, where denotes the similarity between the neighborhood structure of nodes. Given a user pair $(u, v)$ in the network, the more common neighbors shared by them, the closer users $u$ and $v$ are in the network. Besides the original representation $x_u$ for node $u \in V$, the nodes are also associated with a feature vector representing its context in the network, which is denoted as $y_u \in \mathbb{R}^d$. Formally, for a given link $(u, v) \in E$, the probability of context $y_v$ generated by node $u$ can be represented as

$$p_2(v|u) = \frac{e^{x_u^t y_v}}{\sum_{v' \in V} e^{x_u^t y_{v'}}}. \quad \text{(124)}$$
Slightly different from first-order proximity, the second-order empirical proximity is denoted as

\[ p_2(v|u) = \frac{u_i(u,v)}{D(u)} \]  

(125)

By minimizing the difference between the introduced proximity distribution and the empirical proximity distribution, the objective function for the second-order proximity can be represented as

\[ J_2 = \sum_{u \in V} \lambda_u d(p_2(\cdot|u), \hat{p}_2(\cdot|u)), \]  

(126)

where \( \lambda_u \) denotes the prestige of node \( u \) in the network. Here, by replacing the distance function \( d(\cdot|\cdot) \) with the KL-divergence and setting \( \lambda_u = D(u) \), the second-order proximity based objective function can be represented as

\[ J_2 = -\sum_{(u,v) \in E} w_{(u,v)} \log p_2(v|u). \]  

(127)

**Model Optimization:** Instead of combining the first-order proximity and second-order proximity into a joint optimization function, LINE learns the embedding vectors based on Equations 123 and 127 respectively, which will be further concatenated together to obtain the final embedding vectors.

In optimizing objective function 127, LINE needs to calculate the conditional probability \( P(\cdot|u) \) for all nodes \( u \in V \) in the network, which is computational infeasible. To solve the problem, LINE uses the negative sampling approach instead. For each link \( (u, v) \in \mathcal{E} \), LINE samples a set of negative links according to some noisy distribution.

Formally, for link \( (u, v) \in \mathcal{E} \), the set of negative links sampled for it can be represented as \( \mathcal{C}_{(u,v)} \subset V \times V \). The objective function defined for link \( (u, v) \) can be represented as

\[ \log \sigma(y_u^T \cdot x_v) + \sum_{(u', v') \in \mathcal{C}_{(u,v)}} \log \sigma(-y_u^T \cdot x_{u'}), \]  

(128)

where \( \sigma(\cdot) \) is the sigmoid function. The first term in the above equation denotes the observed links, and the second term represents the negative links drawn from the noisy distribution. Similar approach can also be applied to solve the objective function in Equation 123 as well. The new objective function can be solved with the asynchronous stochastic gradient algorithm (ASGD), which samples a mini-batch of links and then update the parameters.

### 8.2.3 node2vec

In LINE, the closeness among nodes in the networks is preserved based on either the first-order proximity or the second-order proximity. In a recent work, node2vec [22], the authors propose to preserve the proximity between nodes with a sampled set of nodes in the network.

**node2vec Framework:** Model node2vec is based on the SkipGram in language modeling, and the objective function of node2vec can be formally represented as

\[ \max \sum_{u \in V} \log P(\Gamma(u)|x_u). \]  

(129)

where \( x_u \) denotes the latent feature vector learnt for node \( u \) and \( \Gamma(u) \) represents the neighbor set of node \( u \) in the network.

To simplify the problem and make the problem solvable, some assumptions are made to approximate the objective function into a simpler form.

- **Conditional Independence Assumption:** Given the latent feature vector \( x_u \) of node \( u \), by assuming the observation of node in set \( \Gamma(u) \) to be independent, the probability equation can be rewritten as

\[ P(\Gamma(u)|x_u) = \prod_{v \in \Gamma(u)} P(v|x_u). \]  

(130)

- **Symmetric Node Effect:** Furthermore, by assuming the source and neighbor nodes have a symmetric effect on each other in the feature space, the conditional probability \( P(v|x_u) \) can be rewritten as

\[ P(v|x_u) = \frac{e^{x_u^T \cdot x_v}}{\sum_{v' \in V} e^{x_u^T \cdot x_{v'}}}. \]  

(131)

Therefore, the objective function can be simplified as

\[ \max_x \sum_{u \in V} [-\log Z_u + \sum_{v \in \Gamma(u)} x_v^T \cdot x_u], \]  

(132)

where \( Z_u = \sum_{v' \in V} e^{x_u^T \cdot x_{v'}} \). Term \( Z_u \) will be different for different nodes \( u \in V \), which is expensive to compute for large networks, and node2vec proposes to apply the negative sampling technique instead. The main issue discussed in node2vec is about sampling the neighborhood set \( \Gamma(u) \) from the network.

To overcome the shortcomings of BFS and DFS, node2vec proposes to apply random walk to sample the neighborhood set \( \Gamma(u) \) instead. Given a random walk \( W \), the node \( W \) resides at in step \( i \) can be represented as variable \( s_i \in V \). The complete sequence of nodes that \( W \) has resided at can be represented as \( s_0, s_1, \ldots, s_k \), where \( s_0 \) denotes the initial node starting the walk. The transitional probability from node \( u \) to \( v \) in \( W \) in the \( i \), \( k \) step can be denoted as

\[ P(s_i = v|s_{i-1} = u) = \begin{cases} \frac{w_{(u,v)}}{d_{l,v}}, & \text{if} \ (u, v) \in \mathcal{E}, \\ 0, & \text{otherwise,} \end{cases} \]  

(133)

where \( w_{(u,v)} \) denotes the normalized weight of link \( (u, v) \) in the network \( (w_{(u,v)} = 1 \) if the network is unweighted).

Traditional random walk model doesn’t take account for the network structure and can hardly explore different network neighborhoods. node2vec adapts the random walk model and introduce the 2-order random walk model with parameters \( p \) and \( q \), which will help guide the walk. In node2vec, let’s assume the walk just traversed link \( (t, u) \) and can go to node \( v \) in the next step. Formally, the transitional probability of link \( (u, v) \) is adjusted with parameter \( \alpha_{p,q}(t, v) \) (i.e., \( u \), \( w_{(u,v)} = \alpha_{p,q}(t, v) \cdot w_{(u,v)} \)), where

\[ \alpha_{p,q}(t, v) = \begin{cases} \frac{1}{p}, & \text{if} \ d_{t,v} = 0, \\ 1, & \text{if} \ d_{t,v} = 1, \\ \frac{1}{q}, & \text{if} \ d_{t,v} = 2, \end{cases} \]  

(134)

where \( d_{t,v} \) denotes the shortest distance between nodes \( t \) and \( v \) in the network. Since the walk can go from \( t \) to \( u \), and then from \( u \) to \( v \), the distance from \( t \) to \( v \) will be at most 2.

Parameters \( p \) and \( q \) control the walk transition sequence effectively, where parameter \( p \) is also called the return parameter and \( q \) is called the in-out parameter in node2vec.

### 8.3 Emerging Network Embedding

We have introduce several network embedding models in the previous sections already. However, when applied to handle real-world social network data, these existing embedding models can hardly work well. The main reason is that the network internal social links are usually very sparse in online social networks [67], which can hardly preserve the complete network structure. For a pair of users...
who are not directed connected, these models will not be able determine the closeness of these users’ feature vectors in the embedding space. Such a problem will be more severe when it comes to the emerging social networks [95], which denote the newly created online social networks containing very few social connections.

In this section, we will study the emerging network embedding problem across multiple aligned heterogeneous social networks simultaneously. To solve the problem, in this section, we will introduce a novel aligned heterogeneous social network embedding framework, named DIME proposed in [93].

8.3.1 Deep DIME-SH Model

DIME is based on the aligned auto-encoder model, which extends the traditional deep auto-encoder model to the multiple aligned heterogeneous networks scenario. In the application of DIME on a heterogeneous network $G(1)$, multiple node meta proximity matrices will be extracted (i.e., $\{P_{\Phi_0}(1), P_{\Phi_1}(1), \cdots, P_{\Phi_T}(1)\}$) based on the intra-network meta paths. As shown in the architecture in Figure 4 (either the left component for network 1 or the right component for network 2), about the same instance, DIME-SH (DIME-Single Heterogeneous Network) takes different feature vectors extracted from the meta paths $\{\Phi_0, \Phi_1, \cdots, \Phi_T\}$ as the input. For each meta path, a series of separated encoder and decoder steps are carried out simultaneously, whose latent vectors are fused together to calculate the final embedding vector $\mathbf{z}_i^{(1)} \in \mathbb{R}^d$ for user $u_i$ in $\mathbb{Y}(1)$. In the DIME-SH model, the input feature vectors (based on meta path $\Phi_k \in \{\Phi_0, \Phi_1, \cdots, \Phi_T\}$) of user $u_i$ can be represented as $\mathbf{x}_i^{(1)}$, which denotes the row corresponding to users $u_i$ in matrix $P_{\Phi_k}$ defined before. Meanwhile, the latent representation of the instance based on the feature vector extracted via meta path $\Phi_k$ at different hidden layers can be represented as $\{\mathbf{y}_i^{(1)}, \Phi_k, \mathbf{y}_i^{(2)}, \cdots, \mathbf{y}_i^{(T)}\}$.

One of the significant difference of model DIME-SH from traditional auto-encoder model lies in the (1) combination of various hidden vectors $\{\mathbf{y}_i^{(1)}, \Phi_k, \mathbf{y}_i^{(2)}, \cdots, \mathbf{y}_i^{(T)}\}$ to obtain the final embedding vector $\mathbf{z}_i^{(1)}$ in the encoder step, and (2) the dispatch of the embedding vector $\mathbf{z}_i^{(1)}$ back to the hidden vectors in the decoder step. As shown in the architecture, formally, these extra steps can be represented as

\[
\begin{align*}
\text{# extra encoder steps} & \quad \mathbf{z}_i^{1, o+1} = \sigma(\sum_{\Phi_k \in \{\Phi_0, \cdots, \Phi_T\}} \mathbf{W}_{i, \Phi_k}^{1, o} \mathbf{y}_i^{1, o} + \mathbf{b}_{\Phi_k}^{1, o+1}), \\
\text{# decoder steps} & \quad \mathbf{y}_i^{1, o} = \sigma(\mathbf{W}_{i, \Phi_k}^{1, o} \mathbf{z}_i^{1, o+1} + \mathbf{b}_{\Phi_k}^{1, o}),
\end{align*}
\]

What’s more, since the input feature vectors are extremely sparse (lots of the entries have value 0s), simply feeding them to the model may lead to some trivial solutions, like 0 vectors for both $\mathbf{z}_i^{(1)}$ and the decoded vectors $\mathbf{z}_i^{(1)}$. To overcome such a problem, another significant difference of model DIME-SH from traditional auto-encoder model lies in the loss function definition, where the loss introduced by the non-zero features will be assigned with a larger weight. In addition, by adding the loss function for each of the meta paths, the final loss function in DIME-SH can be formally represented as

\[
\mathcal{L}^{(1)} = \sum_{\Phi_k \in \{\Phi_0, \cdots, \Phi_T\}} \sum_{u_i \in \mathbb{V}} \left\| (\mathbf{x}_i^{(1), \Phi_k} - \mathbf{z}_i^{(1), \Phi_k}) \odot \mathbf{b}_i^{(1)} \mathbf{z}_i^{(1), \Phi_k} \right\|_2^2.
\]

8.3.2 Deep DIME Framework

By accommodating the embedding between the aligned networks, information can be transferred from the aligned mature network to refine the embedding result in the emerging network effectively. The complete architecture of DIME is shown in Figure 4, which involves the DIME-SH components for each of the aligned networks, where the information transfer component aligns these separated DIME-SH models together. To be more specific, given a pair of aligned heterogeneous networks $G = (G(1), G(2), (A(1,2))$ (with $G(1)$ as an emerging network and $G(2)$ as a mature network), the embedding results can be represented as matrices $\mathbf{Z}(1) \in \mathbb{R}^{|\mathbb{U}(1)| \times d(1)}$ and $\mathbf{Z}(2) \in \mathbb{R}^{|\mathbb{U}(2)| \times d(2)}$ for all the user nodes in $G(1)$ and $G(2)$ respectively. The $i$th row of matrix $\mathbf{Z}(1)$ (or the $j$th row of matrix $\mathbf{Z}(2)$) denotes the encoded feature vector of user $u_i$ in $G(1)$ (or $u_j$ in $G(2)$). If $u_i$ and $u_j$ are the same user, i.e., $(u_i, (1,2)) \in \mathbb{A}(1,2)$, by placing vectors $\mathbf{Z}(1)(i,:)$ and $\mathbf{Z}(2)(j,:)$ in a close region in the embedding space, information from $G(2)$ can be used to refine the embedding result in $G(1)$.

Information transfer is achieved based on the anchor links, and we only care about the anchor users. To adjust the rows of matrices $\mathbf{Z}(1)$ and $\mathbf{Z}(2)$ to remove non-anchor users and make the same rows correspond to the same user, DIME introduces the binary inter-network transitional matrix $\mathbf{T}(1,2) \in \mathbb{R}^{d(1) \times |\mathbb{U}(2)|}$. Entry $\mathbf{T}(1,2)(i,j) = 1$ iff the corresponding users are connected by anchor links, i.e., $(u_i(1), u_j(2)) \in \mathbb{A}(1,2)$. Furthermore, the encoded feature vectors for users in these two networks can be of different dimensions, i.e., $d(1) \neq d(2)$, which can be accommodated via the projection $\mathbf{W}(1,2) \in \mathbb{R}^{d(1) \times d(2)}$.

Formally, the introduced information fusion loss between networks $G(1)$ and $G(2)$ can be represented as

\[
\mathcal{L}^{(1,2)} = \left\| (\mathbf{T}(1,2)^T \mathbf{Z}(1) - \mathbf{Z}(2)) \right\|_F^2.
\]

The complete objective function of framework include the loss terms introduced by the component DIME-SH for networks $G(1)$, $G(2)$, and the information fusion loss. The latent embedding vectors achieved via DIME-SH on network $G(1)$ define the embedding vectors of user nodes in the emerging network.

![Figure 4: The DIME Framework.](image-url)
9. CONCLUSION AND FUTURE POTENTIAL DEVELOPMENTS

In this paper, we have introduced the current research works on broad learning and its applications on social media studies. This paper has covered 5 main research directions about broad learning based social media studies: (1) network alignment, (2) link prediction, (3) community detection, (4) information diffusion and (5) network embedding. These problems introduced in this chapter are all very important for many concrete real-world social network applications and services. A number of nontrivial algorithms have been proposed to resolve these problems, which have been talked about in great detail in this paper respectively.

Both the broad learning and social media mining are very promising research directions, and some potential future development directions are illustrated as follows.

1. Scalable Broad Learning Algorithms: Data generated nowadays is usually of very large scale, and fusion of such big data from multiple sources together will render the problem more challenging. For instance, the online social networks (like Facebook) usually involve millions even billions of active users, and the social data generated by these users in each day will consume more than 600 TB storage space (in Facebook). One of the major future development about the broad learning based social media mining is to develop scalable data fusion and mining algorithms that can handle such a large volume (of big data) challenge. One tentative approach is to develop information fusion algorithms based on distributed platforms, like Spark and Hadoop [26], and handle the data with a large distributed computing cluster. Another method to resolve the scalability challenge is from the model optimization perspective. Optimizing existing learning models and proposing new approximated learning algorithms with lower time complexity are desirable in the future research projects. In addition, applications of the latest deep learning models to fuse and mine the large-scale datasets can be another alternative approach for the scalable broad learning on social networks.

2. Multiple Sources Fusion and Mining: Current research works on multiple source data fusion and mining mainly focus on aligning entities in one single pair of data sources (i.e., two sources), where information exchange between the sources mainly rely on the anchor links between these aligned entities. Meanwhile, when it comes to fusion and mining of multiple (more than two) sources, the problem setting will be quite different and become more challenging. For example, in the alignment of more networks, the transitivity property of the inferred anchor links needs to be preserved [98]. Meanwhile, in the information transfer from multiple external aligned sources to the target source, the information sources should be weighted differently according to their importance. Therefore, the diverse variety of the multiple sources will lead to more research challenges and opportunities, which is also a great challenge in big data studies. New information fusion and mining algorithms for the multi-source scenarios can be another great opportunity to explore broad learning in the future.

3. Broader Learning Applications: Besides the research works on social network datasets, the third potential future development of broad learning and mining lies its broader applications on various categories of datasets, like enterprise internal data [100; 88; 103; 102], geo-spatial data [89; 77; 90], knowledge base data, and pure text data. Some prior research works on fusing enterprise context information sources, like enterprise social networks, organizational chart and employee profile information have been done already [100; 88; 103; 102]. Several interesting problems, like organizational chart inference [100], enterprise link prediction [88], information diffusion at workplace [103] and enterprise employee training [102], have been studied based on the fused enterprise internal information. In the future, these areas are still open for exploration. Applications of broad learning techniques in other application problems, such as employee training, expert location and project team formation, will be both interesting problems awaiting for further investigation. In addition, analysis of the correlation of different traveling modalities (like shared bicycles [89; 77; 90], bus and metro train) with the city zonings in smart city; and fusing multiple knowledge bases, like Douban and IMDB, for knowledge discovery and truth finding are both good application scenarios for broad learning research works.

10. REFERENCES


The DevOps Paradigm with Cloud Data Analytics for Green Business Applications

Michael J. Pawlish$^1$ and Aparna S. Varde$^2$
1. School of Business, Adelphi University, Garden City, NY, USA
2. Department of Computer Science, Montclair State University, Montclair, NJ, USA
mpawlish@adelphi.edu, vardea@montclair.edu

ABSTRACT
This paper reviews the emergence of the DevOps (development and operations) paradigm in the industry and the influence it has along with cloud based data management and analytics in the greening of business applications. It considers the geoscience domain as an example discussing usefulness in a GIS (geographic information system). Similar claims can be applied to other domains. Investigating the emergence of DevOps technologies and examining the dramatic shift in IT towards cloud and hybrid models for data analytics, the paper paints a picture of systems that have the ability to green their impact on society. It also addresses concerns from a privacy and security perspective and concludes with open issues for further research.

Keywords
Cloud Computing, Hybrid Models, DevOps, Geoscience, Green Energy, Privacy, Security

1. INTRODUCTION
In the last decade there has been an interesting and profound movement in the information technology (IT) sector that has resulted in the coming together of Development and Operations divisions of software companies [20]. This combined with cloud data analytics can cause rapid growth in IT. The emergence of this converging trend will continue to increase the efficiency and effectiveness of the IT industry, and simultaneously this trend will continue to have repercussions throughout society.

The increased speed of building and testing software on a cloud platform will continue to benefit society across industries and cultures by bringing together market tools for hopeful future solutions to challenges and problems [10]. Some of these benefits are discussed in this paper and the green or environmental aspects are highlighted in the work presented as a survey here.

This survey paper provides a background on the convergence of the Development and Operations divisions known as DevOps [1]. It thereby moves to present some of the green energy perspectives that includes cloud and hybrid models with reference to data analytics and IT management. For specific applications, it considers the geoscience domain, explaining the use of DevOps and cloud data analytics in the area of geoinformatics [16]. It considers the context of a geographic information system (GIS) in particular [8]. It also discusses some of the concerns with these advancements and challenges to include privacy and security. In addition, the paper puts forth some open issues with the potential for further research.

This survey paper would be interesting to IT professionals, data scientists as well as business executives. The geoscience community would potentially find it interesting as well. The IT industry would gain an insight into the material presented herewith from a green perspective while also being more acquainted with the focus on DevOps, i.e., merging development and operations. Data scientists would gain insight into the concerned research for defining specific problems based on the open issues and challenges discussed in the paper. Business executives would be able to make better decisions in terms of strategy planning and other such initiatives based on the latest developments and trends so as to maximize the efficiency of their systems and yet maintain the required standards of greenness. Moreover, the security and privacy issues here would interest business decision makers as well. Geoscientists would find all these aspects useful with specific reference to the geoinformatics applications considered herewith.

The rest of this paper is organized as follows. Section 2 discusses the emergence of the DevOps paradigm. Section 3 provides an insight into cloud and hybrid models. Section 4 delves into the green energy perspectives relevant to this survey. Section 5 focuses on the applications of this work in geoinformatics. Section 6 analyzes some of the concerns especially with reference to privacy and security. Section 7 gives the conclusions and outlines potential open issues for research.

2. EMERGENCE OF DEVOPS
The advent of the term DevOps has an interesting history. Until relatively recently the model for developing software and testing it rested in different divisions because most businesses experienced a phenomenon called a silo effect [20]. This refers to the prior organizational design where typically development departments were deliberately separated from operations departments for the delegation of work. The outcome was that upper management often had the problem of integrating the two distinct departments. In the development department the employees were focused on designing and writing software while in the operations department the focus was on testing the software for any design flaws. The result of this organizational design was that companies had large divisions with differing
goals that had problems of bureaucracy as well as longer time horizons to finalize the end products. This motivated the need for change. Consequently, management professionals found that by working with smaller teams and blending employees from both operations and development the problems stated herewith were reduced. The result was the appearance of DevOps teams [1, 20]. Given the goal of the end product being a better overall software package developed in a shorter time period, this new organizational design took off in the software industry. The timeline for development was supported by having the DevOps team work together more efficiently and effectively by integrating the former divisions. The overall result was a better end product that was obtained more rapidly. Moreover, when this effort was combined with the advancements in cloud technology it proved to be an interesting enhancement in the software industry.

3. CLOUD AND HYBRID MODELS

3.1 Cloud Services

The cloud, which utilizes a pay-as-you-go model, offers three main services: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-Service (SaaS). For example, Amazon EC2 is an IaaS, Microsoft Azure is PaaS and Google Docs is a SaaS [10]. In the public cloud deployment model, a feature such as multi-tenancy, that enables a single application to serve multiple customers, is found to be very useful [14]. The public cloud is a good way for businesses to cut IT costs, but still maintain their own specialized cloud or datacenter servers to cater to their needs. While the public cloud would not be suitable for sensitive information such as medical records, the private cloud could be a viable option for companies desiring to keep certain data private while also benefiting from some of the many features of the public cloud [10]. However, a private cloud cannot scale elastically like its public counterpart. In that sense, a shortcoming of the private cloud is its limited ability to dynamically provision resources to meet user demand, a feature in which the public cloud certainly excels.

3.2 Hybrid Approach

Given these pros and cons of the cloud models, a hybrid approach which combines some features of both public and private deployment methods can be a viable option for a lot of business processes. Businesses desiring to keep some data on their private cloud or data center servers while also benefiting from the ability of the public cloud to dynamically provision according to the demand would be wise to consider. The concerned business industry can utilize the on premise IT infrastructure for sensitive data while non-sensitive data is processed in the public cloud. Therefore, one could consider the hybrid approach as a great way to experience the best of both worlds.

In recent years, the cloud has become a more significant part of our lives, yet despite its rising popularity, studies have shown that many people are unaware that they are using the cloud. Estimates have shown that while 90% of global Internet users have used the cloud although only 29% are actually aware of it [6]. Some popular examples of cloud services are Google Drive, Google Docs, iCloud, Dropbox and Microsoft OneDrive. For example, Google Docs, which started as Google’s answer to Microsoft Office, is a widely used example of Software-as-a-Service (SaaS) under the cloud umbrella. The vendor, Google, hosts this application over a network like the Internet. People all over the world can access this application and use it without being concerned about saving the data on their local machine. The data is automatically saved to the cloud servers which can be readily accessed worldwide as long as the users are linked to

Figure 1: Outline of the DevOps Paradigm in Industry
a Google account for the data. Thus, from a business perspective, rather than installing and maintaining Microsoft Office on every computer in an office, companies can save resources such as money, HDD (hard disk drive) space and also time by utilizing a SaaS service such as Google docs.

### 3.3 IT Management and Data Analytics

SaaS applications are just a small aspect of the capabilities of the cloud. Businesses that subscribe to other cloud models can reap greater benefits, especially upstart businesses that cannot handle the initial capital investment of building and maintaining an on premise IT infrastructure. When starting a new business, instead of a company spending an excessive amount of time on backend requirements, such as server management, they can rent the amount of data they need for analysis by subscribing to a cloud data storage service. A common problem with upstart businesses is that the initial investment to build and maintain local servers is too great to handle, and sadly, hinders a potentially great company from attaining its true capacity. Indeed, the initial on premise IT investment alone can hinder an upstart business even before it gets an opportunity to fully begin. By adopting the cloud, an upstart business will have access to a scalable platform to meet their data analytics needs at any given time [10]. As their company grows, and they need more space to house their data, they can simply purchase more. In contrast, on premise IT infrastructures would need to physically build more servers, or find another way to manage their data.

The shift to the cloud has been booming over the past decade since commercial cloud services were offered. To reinforce this growing trend Figure 2 depicts the current market share of cloud services and the year-to-year growth rates for the five largest cloud providers according to the Synergy Research Group [17]. To be noted and presented previously in the paper is the dominance of Amazon with an almost 30% of the total market share of the five largest cloud providers. However, to also be noted are the large year-to-year growth rates, especially at Microsoft (96%) and Google (81%). This growth rate indicates the competitive nature of the cloud industry.

![Figure 2: Market Share and Revenue Growth of Cloud Services](image)

According to a study by Gartner, a research and advisory firm providing Information Technology related insight, on premise IT maintenance accounts for around 80% of IT expenditure [10]. As a result, most businesses running and managing their own IT infrastructure spend only 20% of their time on applications and tasks relating to the business. Indeed, the act of maintaining a local IT infrastructure is expensive, not only financially, but also in terms of getting the most out of employees. If this 80-20 rule were shifted in the other direction, the core business needs would be readily met. This is precisely where the adoption of cloud for data storage, processing and analysis can help an organization. Business owners entrusting cloud vendors to house the data of their companies can potentially enable their employees to focus more on core business tasks which will reap greater benefits for the industry.

### 3.4 Examples of Usage

When adopting the Cloud, businesses can focus on operating expenses, rather than capital expenses, such as building an on premise IT infrastructure. Operating expenses are beneficial to an organization because that gives the business the ability to terminate the costs at will. A simple example of operating expenses in relation to cloud data management would be an online retailer being able to purchase more cloud resources on Black Friday to accommodate for the influx of users. Once they get through the demanding day, they can simply terminate the server capacity they purchased from the cloud to store and analyze their data. In contrast, on premise IT infrastructures would need to build an IT infrastructure to handle the excess demand of Black Friday, which would be a capital expense. The expense of building and maintaining an on premise IT infrastructure could very likely offset the potential benefits of Black Friday revenue. Consequently, adopting the cloud for times such as these would be beneficial for any company looking to maximize its profit and minimize its investment.

Prior to adopting the Cloud, Gregory /Richochet, a fashion retailer in New Zealand, had an IT budget amounting to over $30,000 a year. These expenses covered server management, data processing for each of their stores, remote backup services and IT support for their network and head office. These expenses proved to be overwhelming, so they decided to adopt Vend, an online point-of-sale application, which is run under Rackspace, a cloud hosting provider. This resulted in incredible savings in storing, processing and analyzing data, as well as in other costs. Instead of paying over $30,000 a year, they only paid $350 a month, which amounted to $4200 a year. Consequently, their adoption of the cloud resulted in savings of over $25,000! Indeed, cloud data management can provide substantial savings and be of great benefit to small businesses as well as large enterprises.

Large cloud datacenters are able to provide huge savings to companies because of their ability to purchase in bulk. For instance, the cost of storage at a medium sized datacenter is $2.20 per gigabyte (GB) per month. In contrast, a large datacenter, like the ones cloud providers use, sees huge savings at rates of $0.40 per GB per month [4]. In addition, large cloud data centers can purchase power, network bandwidth, and hardware at 1/5 to 1/7 of the prices offered at medium-sized datacenters, which means that they can give huge savings to consumers and still enjoy a healthy profit [4]. Thus, we can see that the deployment of cloud and hybrid models for the management and analysis of data as well as other IT services distinctly proves beneficial.
4. GREEN ENERGY PERSPECTIVES

4.1 Datacenter Greenness

Over the past decade, global warming has become a concern for many in our society. Since the rise of the industrial revolution, mass production has utilized coal as part of manufacturing goods. This holds true for the computer industry as well. Many of our datacenters, as well as the cloud industry, are run by electrical utilities that rely on coal for power. These electrical generators that are based on coal emit harmful gases that have long-term implications to the environment, as studies have shown. As a result of both future pending legislation in the USA, and strategic factors to have less of an impact on the environment, many businesses, including several cloud vendors, have taken a green initiative. A central question is whether the cloud is less harmful to the environment when compared to using premise IT solutions. In short, the answer to that question seems to be that the cloud has less of an impact on the environment compared to the scenario of “what if all of the current cloud users were using private data centers” as explained below.

Datacenters consume large amounts of energy, about 1.2% of the electricity consumption in the United States and about half of that energy goes to cooling processors [2]. Since so much energy goes towards cooling the servers, a lot of energy is wasted, and consequently, the datacenters produce carbon emissions that could otherwise have been avoided. To avoid such emissions, the cloud could simply allocate their resources to fewer servers, especially those that are not being used could be switched off to save energy, cooling power and reduce global emissions as found in interesting studies worldwide [11]. Additionally, a 2007 study for Gartner estimated that the Information and Communication Technology (ICT) industry is responsible for 2% of the total global carbon dioxide (CO2) emissions. As one can imagine, these global CO2 emissions have become a cause for concern to many environmentalists. In a European Union report, they advocated that a decrease in emission volume of 15-30% is required by 2020. As such, many businesses, including cloud datacenters, need to do their part in decreasing the global emission volume.

While the cloud datacenters need to work towards reducing these emission volumes, “...a research study by Accenture shows that moving business applications to the cloud can reduce the carbon footprint of an organization”. According to this report, small businesses reduced their emissions by 90%, while it was found that large businesses reduced their emissions by 30-60%. Indeed, for the businesses passionate in their desire to preserve the environment, it would be wise to consider the cloud.

Although the emission of carbon by cloud datacenters is arousing concern among many environmental scientists, the cloud exhibits features that enable green computing [11]. Multi-tenancy is a key feature in the ability of the cloud to save energy and emit less greenhouse gases in the atmosphere [14]. In essence, multi-tenancy enables virtualization, which thereby helps the cloud group computing resources utilized by several businesses. In this way, the services that are used are consolidated to the fewest amounts of physical servers possible, enabling underutilized servers to be shut down temporarily. This means less energy is used from the servers and the cooling systems. As a result, significant reduction in carbon emissions can be achieved.

4.2 Cloud Features for Green Datacenters

There are four key features of the cloud that enable green computing: Dynamic provisioning, Multi-tenancy, Server utilization and Datacenter efficiency [11, 14]. These are briefly explained next.

1. Dynamic Provisioning: As mentioned earlier, the capability of the cloud to provision resources based on the amount of traffic it receives is a key feature to energy and cost savings. By utilizing only the server capacity that is necessary, less power and cooling resources are consumed in the datacenter. Thus, lower carbon emissions are released into the atmosphere.

2. Multi-tenancy: This refers to multiple companies being able to utilize the software installed on a single infrastructure, such as with SaaS services. This is much more energy efficient than utilizing the software installed on multiple infrastructures.

3. Server Utilization: Through adequate use of servers, the applications and resources can be hosted on a single server, which in turn leads to less energy being consumed by the datacenter. This implies less carbon emissions into the atmosphere.

4. Datacenter Efficiency: A PUE (Power Usage Effectiveness) level measures the efficiency of energy use. It is measured by the total amount of energy produced by the facility divided by the total amount of energy the IT equipment produces. Cloud datacenters can achieve PUE levels of 1.1 to 1.2, which is about 40% lower than the average individual datacenter.

The cloud has many features that precipitate green computing. While there is certainly room for improvement, such as ensuring datacenters only use energy that is necessary, overall, the cloud provides a service many green businesses can use.

4.3 Utilization Analysis of Datacenters

Additional information that is important to note from our research on private data centers is the amount of time the servers are actually being used, in other words, utilization rate. We have conducted research pertaining to this on a mid-size university located in the eastern part of the United States over a five-year period [13]. Presented herewith are Table 1 and Table 2 that introduce the concept of utilization rates on two hosts for the first half of a typical year. The year shown here is 2012. The average utilization rate is the amount of time the servers are active.

It is to be noted that the range of the average utilization rate from 29% to 42% translates to the servers not being used more than 50% of the time [13]. This is an important finding since if servers are designed with some type of sleep mode, there is a potential for energy savings. The two main energy costs are for the servers and for cooling. From our research, it has been found that the two main energy costs each account for approximately 50% of the energy costs. Designing servers with a sleep mode when not being utilized would have the dual benefit of lowering
the energy use and also decreasing the carbon footprint of the organization.

### Table 1: Utilization Rates for Host 1

<table>
<thead>
<tr>
<th>Month</th>
<th>Average Utilization rate</th>
<th>Monthly low</th>
<th>Monthly high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan.</td>
<td>38%</td>
<td>7%</td>
<td>86%</td>
</tr>
<tr>
<td>Feb.</td>
<td>34%</td>
<td>10%</td>
<td>85%</td>
</tr>
<tr>
<td>March</td>
<td>30%</td>
<td>7%</td>
<td>60%</td>
</tr>
<tr>
<td>April</td>
<td>35%</td>
<td>8%</td>
<td>68%</td>
</tr>
<tr>
<td>May</td>
<td>35%</td>
<td>10%</td>
<td>63%</td>
</tr>
<tr>
<td>June</td>
<td>29%</td>
<td>9%</td>
<td>60%</td>
</tr>
</tbody>
</table>

### Table 2: Utilization Rates for Host 2

<table>
<thead>
<tr>
<th>Month</th>
<th>Average Utilization rate</th>
<th>Monthly low</th>
<th>Monthly high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan.</td>
<td>42%</td>
<td>20%</td>
<td>86%</td>
</tr>
<tr>
<td>Feb.</td>
<td>35%</td>
<td>25%</td>
<td>90%</td>
</tr>
<tr>
<td>March</td>
<td>38%</td>
<td>21%</td>
<td>87%</td>
</tr>
<tr>
<td>April</td>
<td>35%</td>
<td>9%</td>
<td>82%</td>
</tr>
<tr>
<td>May</td>
<td>38%</td>
<td>21%</td>
<td>84%</td>
</tr>
<tr>
<td>June</td>
<td>42%</td>
<td>18%</td>
<td>90%</td>
</tr>
</tbody>
</table>

### Figure 3: Plots of kWh vs Time for Power Distribution Units

An additional concern that we have noticed in our research on private data centers is the balancing of servers on the Power Distribution Units (PDU) [13]. Presented in Figure 3 are the four PDUs that monitor the power going to the data center servers on an individual day. We have found that the power is consistent from day-to-day, however the power is not evenly balanced on the individual PDUs (as noticed by PDU 3 and 4 using more power per kWh). We believe that by balancing the power on the PDUs in a more even format would prolong the life of the equipment. It would possibly distribute the heat of the individual servers more evenly which would be more efficient.

### 5. GEO-INFORMATICS APPLICATIONS

#### 5.1 Cloud Data Storage for Geo-Informatics

The area of geo-informatics deals with processing and analyzing information of a geographic nature where spatial and temporal factors are crucial and location-specific services are likely to be needed [16]. It is an important area in the whole realm of geoscience in general. Cloud data management can play a significant role here. Some of the important trends in cloud data services have been presented herewith. Companies that implement cloud technology are using less hardware and licensed software. The reasons behind this shift are the freedom to reach the devices from anywhere, reduction of the investment on assets, lower costs and the ability to shift from a capital expenditure to an operational expense [4, 7].

An overall trend is that more companies will adopt a cloud strategy for storing and analyzing their data in geo-informatics applications and cloud spending will increase. It was foreseen that cloud market would continue to grow in 2016 which has already been witnessed. According to new projections, the global market will grow from its $49 billion that existed in 2016 to $67 billion in 2018. As more organizations shift to some form of cloud services to store and analyze their data, the demand will continue to grow and the amount of computing power will expand [7]. This will have a positive impact on the applications in geo-informatics since they will be able to utilize the increased potential of cloud technologies to enhance their location-specific services.

A second trend is that hybrid models will continue to be a popular option and grow at a rapid pace [13]. Public clouds offer cost savings and convenience, but some companies require the ability to add layers of security and customization, so they opt for a private cloud. Increasingly, purchasers are choosing the best of both worlds: a hybrid cloud that can house non-sensitive data on a public cloud while storing crucial systems and sensitive data on a private cloud or their own datacenter servers. Recently, private and hybrid cloud adoption rates were slightly less than public cloud growth. Moving forward, hybrid options will grow at a faster rate as more companies choose the convenience of the public cloud and the security of the private cloud. Geo-informatics applications will benefit from these hybrid options since they can maintain their critical data for highly sensitive location-specific services on a private cloud and outsource operations of a more generic nature for other less crucial location-based services to the public cloud, thereby incurring energy savings while yet catering to privacy and security issues.

Another major trend is that more staff will be involved in the cloud adoption decision process [7]. One of the most important aspects of cloud based data services is convenience. Armed with only a credit card, business unit leaders can instantly acquire the computing power they need to support expanded operations and scale up to handle major projects. This key aspect of cloud technology has revolutionized the industry. Other issues are that
education and training will be very important because as more companies adopt cloud technology, trained staff will be in demand for growing needs of the organization [7]. This will have an impact on geo-informatics by getting geoscientists and cloud professionals involved in decision-making and strategy planning along with corporate business executives.

5.2 Emergence of DevOps in Geo-Informatics

An important aspect of geo-informatics is GIS, i.e., Geographic Information System. Geo-informatics deals with the area of managing the geographic information in general with respect to collecting the data, extracting the knowledge and provisioning it to users [16]. A GIS deals with the more specific tasks of an IS, i.e., information system per se. It has specific goals with respect to various deliverables such as Demographic Analysis, Emergency Preparedness, Land Management and others [5, 8]. A broad range of these are illustrated in Figure 4 which depicts an overview of a GIS for a town in Michigan [5].

![Figure 4: Example of a GIS Snapshot for a town in MI, USA](image)

DevOps, a cultural and professional IT movement focused on changing the mindset of how organizations function, will certainly make a significant impact across many industries and companies in the near future [20]. An example of the broad impact DevOps has is in the Geographic Information System (GIS) area. The ability to portray maps on multiple layers has profound changes in the areas of engineering, demographic analysis and land management to just name a few. The DevOps paradigm provides faster delivery of product features, faster resolution of problems and continuous software delivery, all of which are advantageous in GIS applications. DevOps can benefit the product lifecycle of a company, its competitive advantage and its ability to meet customer needs more rapidly. Despite all of its benefits, DevOps is still in its early stages, because it is about making a significant cultural shift in how an organizations works and the tools the organization uses to develop its products.

A significant trend in the DevOps category is an increase in modular approaches to system building [15]. Previously, IT companies created monolithic products for their customers. However, the best practice today is to arrange small and agile teams to manage individual applications. Instead of huge teams working on one big application, a shift in acceptance in the software industry is occurring where small but agile groups are doing better work in individual applications. This is useful in building systems in geo-informatics where location-specific Development and Operations tasks can be carried out by smaller teams in those respective locations, thereby enhancing product delivery and functionality.

DevOps enable further advances in programmable infrastructure as a code. The idea of automation per se is not novel, however the ability to provision infrastructure easily and seamlessly has changed the nature of the game. As DevOps and the open-source software movement becomes more readily accepted, infrastructure can be programmed so that teams can develop the software and operate in the external environment simultaneously [20]. This is a huge advantage in geo-informatics where interacting with the external environment is essential and frequent.

An additional trend is that developers will take greater ownership of the entire product lifecycle. As DevOps-ready tools see more adoption and out-of-the-box functionality, the traditional silos between developers and operations will lose their importance. As many teams focus on continuous delivery and improvement, it means greater accountability and ownership from developer teams to build and run their solutions. DevOps leaders need to keep developers involved in the performance of the application and any issues that occur, as well as communicate that their job does not end after the application has been delivered [20]. This is beneficial to the designers of GIS since there are several aspects that need to be addressed with development and operations going hand in hand as evident from Figure 3. For example, in order to cater to a goal such as Demographic Analysis, it is important to develop the piece of software that captures the demographics of the region and then conduct the analysis with respect to specific questions that may be of interest to prospective users. Thus, DevOps teams with development and operations occurring in conjunction with each other are useful in this context. Developers need to be part of the entire lifecycle, and have complete visibility into its progress.

On the whole, DevOps will dramatically reduce the time to deploy any feature to production [15]. This trend is already occurring, as product methodologies transition from the traditional waterfall to more agile ways of working. As this transformation occurs, so also will the systems be more risk tolerant. Any changes are less likely to negatively impact the entire system, thus the time to address them will be reduced to hours and minutes, rather than days or weeks. Companies should start measuring how long it takes to deliver a bug fix or a feature to production, and monitor this on a weekly basis. This is a clear metric that can help monitor the success of DevOps [15]. It is particularly useful in geo-informatics applications where location-specific services could be subject to several changes as things progress. Thus, implementing those changes should be a fast process. Analysis of data in GIS and other geo-informatics applications needs to take into account rapid changes in order to reveal useful and timely results. For instance, consider Emergency Preparedness with reference to Figure 4. In order to
develop and operate the system with reference to this goal, it is crucial to incorporate changes that occur in emergency procedures in specific locations. Else, the system may not function well if it is not aware of novel features available, e.g., a new kind of evacuation procedure. Thus a good GIS should be able to incorporate significant changes during the developmental and operational phases. Hence, the emergence of DevOps is highly suitable to speed up processes here in order to meet the goals of the systems and deliver better outputs.

6. PRIVACY AND SECURITY

6.1 Significant Concerns

While the cloud enables scalability, reduces cost and has positive effects on the environment, one of the obstacles preventing businesses from adopting the cloud, is security. While the cloud has been gaining popularity in recent years, there is still some apprehension on adopting the service, for the fear that the data is not secure on any of the cloud deployment models, especially the public cloud [9]. The traditional security methods, such as basic user authentication and authorization, are no longer adequate. This is further aggravated when cloud data management is used in conjunction with DevOps. Since development and operations teams work together to deliver the outputs, the privacy and security issues would be spread across both these teams. We discuss these with specific reference to the SaaS, PaaS and IaaS services provided by the cloud that would be used by the DevOps teams in various applications.

SaaS gives consumers the ability to run the provider applications over a cloud infrastructure. With this model, the security lies with the cloud provider, not with the user. With respect to PaaS, developers do not have access to the underlying layers, thus the burden or security rests with the provider rather than the user. In contrast, IaaS, which serves as a foundation layer for the other delivery models, the security burden rests with the cloud user. While the providers still manage virtualization, servers, HDDs, networking and storage, IaaS users are responsible for their own security, operating system, applications and middleware.

One of the concerns with the cloud is the dependency that each cloud computing model has on each other. For instance, since PaaS and SaaS are hosted on top of IaaS, any security issue that occurs on the IaaS level could have dire consequences on the other levels as well. This is also true the other way around. For example, PaaS offers a platform to host SaaS services, thus the integrity of the PaaS platform is a factor in the SaaS security. A security concern of SaaS lies in Web applications and the manner in which its flaws can harm the SaaS layer or its breaches in security can occur through Web applications. Furthermore, while multi-tenancy is a key feature in green computing and cost reduction, there are security concerns. A key feature of multi-tenancy is the ability for multiple applications to be stored on the same physical server or database. This helps reduce cost and energy. However, it also comes with a security risk. “Since data from multiple tenants is likely to be stored in the same database, the risk of data leakage between tenants is high” [9]. This poses significant issues.

PaaS delivers applications over the Internet. It is a platform where developers can build and customize applications. An example of PaaS is Apprenda, which is a free PaaS service that provides developers with the tools to create mobile and cloud-based applications before using the application in their own on premise infrastructure [3]. While PaaS is a powerful and useful way to create applications, developers need to be mindful of the changes within the platform. Any changes to the PaaS components can compromise the security of the applications. This should be addressed especially taking into account the DevOps paradigm given that development and operations teams could be availing of multiple platforms over the cloud using PaaS while performing their respective tasks.

IaaS provides virtualized computer resources such as servers, storage and networks. In IaaS, cloud users have greater control of their security, however, the Cloud provider controls the network and storage infrastructure. One aspect is that the use of virtualization is a security concern, since there are more points of entry for attacks than physical servers. In addition, since virtual machines on the same server share CPU, memory etc., if there is any malicious machine located on the same server, it could retrieve sensitive information. This is of great concern while using infrastructure as a service. DevOps teams need to take into account virtualization and related aspects while they function together in the industry. Security measures need to be proposed to deal with such issues.

6.2 Research Perspectives

Researchers and developers have often investigated the privacy and security concerns associated with the use of cloud technologies and have discussed potential solutions in the respective applications. For example, Tancer et al. [18] have presented some useful insights on these with respect to the use of the Medical Markup Language, MML, on the cloud. They consider issues such as: consistent availability of cloud systems when tasks are crucial; storage of private medical records on the public cloud; and differences of laws in countries across which medical records would be accessed for practice and research. Some of the arguments applied to cloud storage and processing of the crucial medial data therein [18] could also pertain to sensitive information in other fields such as geoscience. Accordingly, users of many systems such as a GIS need to be aware of such concerns while deploying cloud services, public and / or private, in their systems. Often these concerns can be addressed and solved by the respective developing organizations before providing cloud data services to the users. The responsibility therefore should ideally be assigned to the providers as opposed to users regarding the cloud services and their associated concerns.

Doctoral students in data management and related areas such as data mining find it interesting to conduct research on issues pertaining to cloud technologies. An overview of such dissertation research is presented in [19]. It is found here that many security and privacy issues arouse interests among doctoral candidates. They address these with respect to solutions such as modeling trust in a cloud context; conducting data perturbation to conceal identity while yet maintaining authenticity of results; and incorporating domain-specific aspects from a user perspective. Much of this work presents the potential for future research as well. We will discuss this more in the next section.

The DevOps paradigm being relatively new does not seem to be very extensively researched with respect to privacy and security. However, we advocate that particularly when being used along with cloud data management, the privacy and security concerns can be further heightened due to multiple professionals from different spheres working together and also due to the use of
shared software services, multiple platforms and infrastructure issues. This mandates further research in the areas of cloud data analytics along with DevOps to cater to modern business needs.

As with every technological advancement, there are pros and cons. While the cloud offers several great features for cost reduction and energy efficiency, there are security concerns within each cloud computing model that prospective users should know. Outlining some of the security issues is not an attempt to deter anyone from adopting the cloud, but rather, to arm them with the knowledge necessary to make an informed decision. If a business does decide to adopt the cloud, they should be aware of all the potential pros and cons and thereby outline their strategies. Moreover, if the business is heading towards the adaptation of DevOps in their functioning, they need to understand that some of the privacy and security concerns become even more pronounced. Hence, they need to make business decisions accordingly.

It is important to note that DevOps provides faster and more effective solutions and end products on the whole [15, 20]; cloud data management is beneficial from a green energy perspective [10, 11, 13]; using the two in conjunction would imply balancing productivity very well with greenness; yet privacy and security issues are very prominent [18, 19] especially while using these technologies together. This motivates the need for further research on several grounds, including privacy and security issues, greener solutions and the enhanced usage of DevOps along with cloud services for data management and analytics.

7. CONCLUSIONS AND OPEN ISSUES

The merging of DevOps and cloud technology has brought forth great strides in the software industry that have strong repercussions throughout all industry and modern society. The switch towards cloud usage for storing, processing and analyzing data has increased a trend towards eco-efficiency in datacenter operations that has resulted in less energy being used.

The trend towards more eco-efficient cloud providers is documented by generally lower PUE (Power Usage Effectiveness) ratios in cloud providers than private datacenters. In addition, the organizational hangs in both the move towards the cloud and the merging of the development and operations departments into DevOps has brought forth faster response times to the building of software. Increasingly, the advancement of software is becoming more important in the daily lives of individuals in modern society, and all types of organizations.

Considering DevOps, the datacenter industry and cloud services there are a number of open issues for research as listed below:

- Investigate specific DevOps case studies on where the development and operations departments have been merged to form more effective organizations; and contrast these with the traditional methods
- Address current developments in the cloud market with the leading five cloud companies presented herewith.
- Find solutions to each of the security issues pertaining to the IaaS, SaaS and PaaS services along with DevOps.
- Incorporate heightened privacy concerns especially for location-specific applications such as GIS.
- Apply more enhanced virtualization procedures in organizations that head towards gaining eco-efficiency and reducing the carbon footprint of operations in various datacenters.
- Perform a detailed comparison of several datacenters incorporating DevOps with cloud and hybrid models.
- Study the differing ways in which a datacenter can go into sleep mode by powering down the servers that would be of interest to save energy and investigate these with respect to the DevOps paradigm.

Thus, while the advancements documented in this survey paper are central to IT management and data analytics, there are still open issues that need to be addressed. The security of the cloud will continue to be a concern that must be investigated, especially given the emergence of the DevOps paradigm and some sensitive applications, e.g., location-specific GIS services. The rules and regulations, as well as their enforcement will be an issue that society will need to monitor and enforce into the future. It is expected that this survey paper will be useful to data scientists, IT professionals, business executives and potentially geoscientists as well. The open issues presented herewith would provide the opportunities for joint research in these areas that would further enhance various technologies in DevOps, green IT, geo-informatics and cloud data analytics.

8. ACKNOWLEDGMENTS

The authors thank Cem Davutoglu and Joseph Grillo from the Department of Computer Science at Montclair State University for their inputs in some parts of this survey. The authors’ research in related areas has been partly supported by a grant from PSE&G (Public Services Electric and Gas Company) in NJ, USA.

9. REFERENCES

[1] Agile Admin, What is DevOps, theagileadmin.com/


About the authors:

Michael Pawlish is currently an adjunct professor at Adelphi University in the Robert B. Willunstad School of Business located in Garden City, NY. He obtained his PhD in Environmental Management from Montclair State University, NJ, USA and his MBA from San Francisco State University, CA, USA. His research interests are business-sustainability focused operations including eco-efficient datacenter processes and sustainable entrepreneurship. He has publications in these areas spanning multidisciplinary venues encompassing Computer Science, Environmental Management and Business. His PhD dissertation has been supported by a research grant from PSE&G in NJ.

Aparna Varde is a tenured Associate Professor of Computer Science at Montclair State University, NJ, USA; and Doctoral Faculty in their Environmental Management PhD program. She obtained her PhD and MS from Worcester Polytechnic Institute, MA, USA and her BE in Computer Engineering from University of Bombay, India. Her research interests span Data Mining, Artificial Intelligence and Database Systems with emphasis on multidisciplinary work. Her current research projects entail commonsense knowledge, smart cities and cloud data mining. She has around 80 publications in journals such as AIEDAM, MTAP, TKDD, IJAC, SIGMOD Record, SIGKDD Explorations; and conferences/workshops of AAAI, SIGMOD, CIKM, KDD, SDM, ICDM, ICDE, EDBT etc. She has served as a panelist for NSF in their division of Intelligent and Information Systems; a reviewer for journals (TKDD, TKDE, DMKD, VLDBJ, DKE, AIEDAM etc.) and PC member of conferences (EDBT, ICDM, CIKM, KDD etc.). Her research is funded by grants from organizations such as NSF, PSE&G and other private companies.