A Survey on Social Media Anomaly Detection

Rose Yu [1]
qiyu@usc.edu

Huida Qiu [1]
huidaqiu@usc.edu

Zhen Wen [2]
zhenwen@us.ibm.com

Ching-Yung Lin [2]
chingyung@us.ibm.com

Yan Liu [1]
yanliu.cs@usc.edu

[1]Department of Computer Science
University of Southern California

[2] Watson Research Center
IBM

ABSTRACT

Social media anomaly detection is of critical importance to prevent malicious activities such as bullying, terrorist attack planning, and fraud information dissemination. With the recent popularity of social media, new types of anomalous behaviors arise, causing concerns from various parties. While a large amount of work have been dedicated to traditional anomaly detection problems, we observe a surge of research interests in the new realm of social media anomaly detection. In this paper, we present a survey on existing approaches to address this problem. We focus on the new type of anomalous phenomena in the social media and review the recent developed techniques to detect those special types of anomalies. We provide a general overview of the problem domain, common formulations, existing methodologies and potential directions. With this work, we hope to call out the attention from the research community on this challenging problem and open up new directions that we can contribute in the future.

1. INTRODUCTION

Social media systems provide convenient platforms for people to share, communicate, and collaborate. While people enjoy the openness and convenience of social media, many malicious behaviors, such as bullying, terrorist attack planning, and fraud information dissemination, can happen. Therefore, it is extremely important that we can detect these abnormal activities as *accurately* and *early* as possible to prevent disasters and attacks. Needless to say, as more social information becomes available, the most challenging question is what useful patterns could be extracted from this influx of social media data to help with the detection task.

By definition, anomaly detection aims to find "an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism" [29]. The common approach is to build a reference model, i.e., a statistical model that captures the generation process of the observed (or normal) data. Then for a new observation, we estimate its likelihood based on the reference model and predict the data as an "anomal" if the likelihood is below some threshold [11; 26; 24; 56; 77; 14].

In addition, the type of anomalies that we aim to detect vary significantly from applications to applications. Several algorithms have been developed specifically for social network anomaly detection on graph structure anomalies, e.g. power law models [2], spectral decomposition [45], scan statistics [54], random walks [51; 69], etc. The basic assumption of these algorithms is that if a social network has fundamentally changed in some important way, it is usually reflected in the individual communication change, i.e., some individuals either communicate more (or less) frequently than usual, or communicate with unusual individuals. However, this could be an over-simplification of the social media anomalies without considering several important aspects of social media data.

One of the challenges that differentiate social media analysis from existing tasks in general text and graph mining is the *social layer* associated with the data. In other words, the texts are attached to individual users, recording his/her opinions or activities. The networks also have social semantics, with its formation governed by the fundamental laws of social behaviors. The other special aspect of social media data is the *temporal perspective*. That is, the texts are usually time-sensitive and the networks evolve over time. Both challenges raise open research problems in machine learning and data mining. Most existing work on social media anomaly detection have been focused on the social perspective. For example, many algorithms have been developed to reveal hubs/authorities, centrality, and communities from graphs [37; 23; 63; 40]; a good body of text mining techniques are examined to reveal insights from user-generated contents [8; 58]. However, very few models are available to capture the temporal aspects of the problem [7; 28; 38], and among them even fewer are practical for large-scale applications due to the more complex nature of time series data.

Existing work on traditional anomaly detection [13; 16; 14; 70; 27; 68; 20; 42; 35; 76; 16; 72] have identified two types of anomalies: one is "univariate anomaly" which refers to the anomaly that occurs only within individual variable, the other is "dependency anomaly" that occurs due to the changes of temporal dependencies between time series. Mapping to social media analysis scenario, we can recognize two major types of anomalies:

- Point Anomaly: the abnormal behaviors of individual users
- Group Anomaly: the unusual patterns of groups of people

Examples of point anomaly can be anomalous computer users [59], unusual online meetings [31] or suspicious traf-

Table 1: Survey Structure

Point Anomaly Detection	
Activity-based	*Bayes one-step Markov*, *compression*[59], *multi-step Markov* [34], *Poisson process* [33], *probabilistic suffix tree* [67]
Graph-based (static graph)	*random walk* [48; 65], *power law* [2; 4] *hypergraph* [62; 61] *spatial autocorrelation*[66; 15]
Graph-based (dynamic graph)	*scan statistics* [54; 52], *ARMA process* [39] *MDL* [64; 3], *graph eigenvector* [32]
Group Anomaly Detection	
Activity-based	*scan statistics* [18; 25], *causal approach* [5] *density estimation* [73; 74; 49; 57]
Graph-based (static graph)	*MDL* [9; 43; 55] *anomalous substructure* [50; 22] *tensor decomposition* [46]
Graph-based (dynamic graph)	*random walk* [44], *t-partitie graph* [75; 36] *counting process* [30]

fic events [33]. Most of the existing work have been devoted to detecting point anomaly. However, in social network, anomalies may not only appear as individual users, but also as a group. For instance, a set of users collude to create false product reviews or threat campaign in social media platforms; in large organizations malfunctioning teams or even insider groups closely coordinate with each other to achieve a malicious goal. Group anomaly is usually more subtle than individual anomaly. At the individual level, the activities might appear to be normal [12]. Therefore, existing anomaly detection algorithms usually fail when the anomaly is related to a group rather than individuals.

We categorize a broad range of work on social media anomaly detection with respect three criteria:

1. Anomaly Type: whether the paper detects point anomaly or group anomaly
2. Input Format: whether the paper deals with activity data or graph data
3. Temporal Factor: whether the paper handles the dynamics of the social network

In the remaining of this paper, we organize the existing literature according to these three criteria. The overall structure of our survey paper is listed in table 1. We acknowledge that the papers we analyze in this survey are only a few examples in the rich literature of social media anomaly detection. References within the paragraphs and the cited papers provide broader lists of the related work.

We can also formulate the categorization in Table 1 using the following mathematical abstraction. Denote the time-dependent social network as $G = \{V(t), W_v(t), E(t), W_e(t)\}$, where V is the graph vertex, W_v is the weight on the vertex, E is the graph edge and W_e is the weight on the edge. Point anomaly detection learns an outlier function mapping from the graph to certain sufficient statistics $F : G \to R$. A node is anomalous if it lies in the tail of the sufficient statistics distribution. Group anomaly detection learns an outlier function mapping from the power set of the graph to certain sufficient statistics $F : 2^{|G|} \to R$. Activity based anomaly detection collapses the edge set $E(t)$ and weights $W_e(t)$ to be empty. Static graph-based approaches fix the time stamp of the graphs as one. Now each of the method summarized in the table is essentially learning a different F or using some projection (simplification) of the graph G. The projection trades-off between model complexity and learning efficiency.

2. POINT ANOMALY DETECTION

Point anomaly refers to the abnormal behaviors of individual users, which can be reflected in abnormal activity records such as unusually frequent access to important system files, or abnormal network communication patterns. Point anomaly detection aims to detect suspicious individuals, whose behavioral patterns deviate significantly from the general public. Based on the type of input, we can have activity-based point anomaly detection and graph-based point anomaly detection.

2.1 Activity-based Point Anomaly Detection

User activities are widely observed in social media, such as computer log-in/log-off records, HTTP access records, and file access records. Activity-based approaches assume that individuals are marginally independent from each other. The anomalousness of an individual is determined only by his own activities. A large body of literature are in the context of computer intrusion detection study. For example, [59] investigates the problem of detecting masquerades who disguise themselves as somebody else on the network. The paper collects user activities by looking at their UNIX commands records and manipulating the data to simulate masquerades.

Pioneering work for detecting masquerades fall into the framework of statistical hypothesis testing, e.g. [21; 34]. Different approaches are proposed including **Bayes one-step Markov**, **hybrid multi-step Markov** and **compression**. Here we omit other simple masquerade detection techniques such as uniqueness of the command as also compared in [21]. For Bayes one-step Markov method, it states the null hypothesis as a one-step Markov process and the alternative hypothesis as a Dirichlet distribution. The null hypothesis

assumes that the current time command C_{ut} of a user u relates to his previous command $C_{u,t-1}$. Mathematically speaking, $H_0 : P(C_{ut} = k|C_{u,t-1} = j) = p_{ukj}$, where p_{ukj} is the transition probability from command j to command k for user u. Then the algorithm computes the Bayes factor based on the hypothesis for each user $\bar{x}_u$ and set up a threshold with respect to $\bar{x}_u$ to detect anomalous masquerades. This approach models users independently and ignores the potential relationships among users.

As a direct generalization of Bayes one-step Markov, [34] builds a user model based on high-order Markov chains: **hybrid multi-step Markov**. It tests over two hypotheses. H_0 : commands are generated from the hybrid Markov model of u; H_1 : commands are generated from other users. The hybrid multi-step Markov method switches between the Markov model and the independence model. The Markov model assumes that a command depends on a set of previous commands, i.e.

$$P(C_{ut} = c_0|C_{u,t-1} = c_1, \cdots, C_{u,t-l} = c_l) = \sum_{i=1}^{l} \lambda_{ui} r_u(c_0|c_i)$$

where λ and r denotes the initial and transitional probability. For the independence model, it assumes that a user's commands are i.i.d samples from a multinomial distribution. The paper computes the test statistics by combining the statistics from two models. Similar to Bayes one-step Markov, hybrid multi-step Markov method sets up a threshold value on the test statistics to flag anomalies. Hybrid multi-step Markov method is able to capture the long-range dependence of the users' commands. However, it also suffers from higher computational cost. **Compression** takes a distinctive approach where it defines the anomaly score as the additional compression cost to append the test data to the training data. Formally, the score is $x = \text{compress}(\{C, c\}) - \text{compress}(C)$, where C is the training data, c is the testing data. The method applies the Lempel-Ziv algorithm for the compression operation. However, it can hardly capture the dependencies in the data instances.

[67] proposes **probabilistic suffix tree** (PST) to mine the outliers in a set of sequences S from an alphabet Σ. It makes Markov assumption on the sequences and encodes the variable length Markov chains with syntax similar to Probabilistic Suffix Automata. In PST, an edge is a symbol in the alphabet and a node is labeled by a string. The probabilistic distribution of each node represents the conditional probability of seeing a symbol right after the string label. An example of such PST is shown in Figure 1. The algorithm first constructs a PST and then computes a similarity measure score SIM_N based on marginal probability of each sequence over the PST. Then it selects the top k sequences with lowest SIM_N scores as outliers. Since PST encodes a Markov chain, which has been shown to have certain equivalence to the Hidden Markov model, the outliers detected by PST are similar to those using Markov model testing statistics. Though PST construction and SIM_N are relative cheap in computation, one drawback is that PST is pre-computed for a fixed alphabet. Pre-computation makes PST less adaptive to the unseen symbols outside of the alphabet or newly coming sequences, which basically requires recomputing the entire tree.

[33] investigates Markov-modulated **Poisson process** to

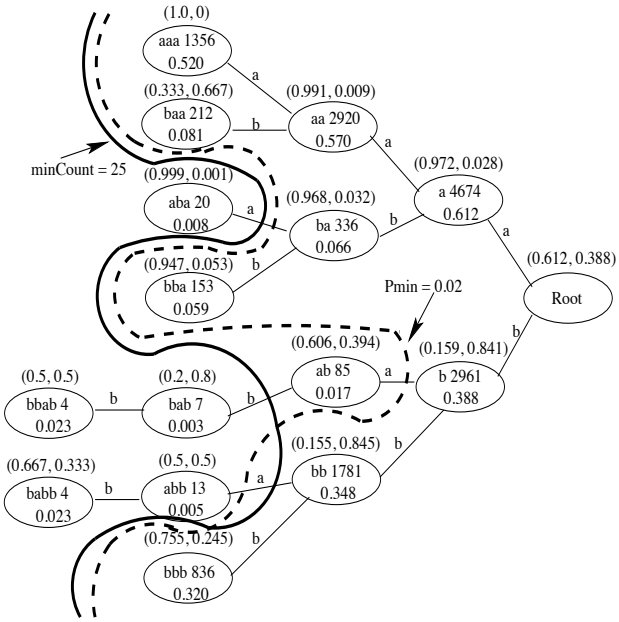


Figure 1: A example of PST. For each node, top array shows the probability distribution. Inside the node shows the label string, the number of times it appears in the data set and the empirical probability. [67]

address the specific problem of event detection on time-series of count data. The algorithm assumes the count at time t, denoted as $N(t)$, is a sum of two additive processes: $N(t) = N_0(t) + N_E(t)$, where $N_0(t)$ denotes the number of occurrences attributed to the "normal" behavior and $N_E(t)$ is the "anomalous" count due to an event at time t. More concretely, the periodic portion of the time series counts can be taken as "normal" behavior while the rare increase in the number of counts can correspond to the "anomalous" behavior. For both processes, the paper develops a hierarchical Bayesian model. In particular, the paper models periodic counting data (i.e. normal behavior) with a Poisson process and models rare occurrences (i.e. anomaly behavior) via a binary process. The algorithm then uses the MCMC sampling algorithm to infer the posterior marginal distribution over events. It uses the posterior probability as an indicator to automatically detect the presence of unusual events in the observation sequence. The paper applies the model to detect the events from free-way traffic counts and the building access count data. The method takes a full Bayesian approach as a principled way to pose hypothesis testing. However, it treats each time series as independent and fails to consider the scenario where multiple time series are inter-correlated.

Another application in social network anomaly detection is proposed in [31]. The paper proposes to detect unusual meetings by investigating the presence of meeting participants. Specifically, for each time stamp $t = 1, 2, \cdots$, the inputs are given as a snapshot of the network in the form of a binary string $x_t = (x_t(1), \cdots, x_t(n)) \in \{0, 1\}^n$, where $x_t(j) = 0$ or 1 indicates whether the jth person participated in the meeting at time t as well as the feedback from expert system with the correct labels $y_t \in \{-1, +1\}$. The algorithm outputs a binary label for each network state $\hat{y}_t \in \{-1, +1\}$

according to whether or not x_t is anomalous. Under the proposed two-stage framework, "filtering stage" estimates the model parameters and updates belief with the new observation. It builds an exponentially model driven by a time-vary parameter and learns the model parameter in an online fashion. "Hedging stage" compares the model likelihood of x_t as ζ_t with the critical threshold τ_t and flag anomalies if $\zeta_t > \tau_t$. After that, the online learning algorithm utilizes the feedback from an expert system to adjust the critical threshold value

$$\tau_{t+1} = \text{argmin}_\tau(\tau - \tau_t - \eta y_t 1_{\hat{y_t} \neq y_t})^2$$

It is easy to see from the construction of x_t that each person's participation is taken as an independent feature entry. Though this work highlights the network structure, the relational information utilized lies only between people and meetings, without considering the interaction among people themselves.

Generally speaking, activity-based approaches model the activity sequence of each user separately under certain Markov assumption. They locate the anomaly by flagging deviations from a user's past history. These approaches provide simple and effective ways to model user activities in a real-time fashion. The models leverage the tool of Bayesian hypothesis testing and detect anomalies that are statistically well-justified. However, as non-parametric methods, Markov models suffer from the rapid explosion in the dimension of the parameter space. The estimation of Markov transition probabilities becomes non-trivial for large scale data set. Furthermore, models for individual normal/abnormal activities are often ad hoc and are hard to generalize. As summarized by [60] in his review work on computer intrusion detection: "none of the methods described here could sensibly serve as the sole means of detecting computer intrusion". Therefore, exploration of deeper underlying structure of the data with fast learning algorithms is necessary to the development of the problem. Here we also refer interested readers to more general reviews of computer network anomaly detection [1; 41].

2.2 Graph-based Point Anomaly Detection

Social media contain a considerably large amount of relational information such as emails from-and-to communication, tweet/re-tweet actions and mention-in-tweet networks. Those relational information are usually represented by graphs. Some approaches analyze static graphs, each of which is essentially one snapshot of the social network. Others go beyond static graph and analyze dynamic graphs, which is a series of snapshot of the networks.

2.2.1 Static Graph

Compared with activity-based approaches, which simplify the social network as categorical or sequential activities of individuals, graph-based approaches further take into account the relational information represented by the graph. [50] immerses as one of the earliest work focusing on graph-based anomaly detection. It introduces two techniques for graph-based anomaly detection. One is to detect anomalous substructures within a graph and the other is to detect unusual patterns in distinct sets of vertices (subgraphs). Substructure is a connected component in the overall graph. Subgraph is obtained by partitioning the graph into distinct structures. Each substructure is evaluated using the Minimum Description Length metric for anomalousness. In real social graphs, intensive research efforts have been devoted to study the graph properties (see references in [10]). One famous example is the power law, which describes the relationship among various attributes, namely the number of nodes (N), number of edges (E), total weight (W) and the largest eigen-value of the adjacency matrix (λ).

Based on these observations, [2] proposes to study each node by looking at **power law** property in the domain of its "egonet", which is the subgraph of the node and its direct neighbors. For a given graph $\mathcal{G}$, denote the egonet of node i as $\mathcal{G}_i$, the paper describes the "OddBall" algorithm. The algorithm starts by investigating the number of nodes N_i, the weight W_i and number of edges E_i of the egonet $\mathcal{G}_i$. It then defines the normal neighborhoods patterns with respect to these quantities. For example, the authors report the Egonet Density Power Law (EDPL) pattern for N_i and E_i: $E_i \propto N_i^\alpha, \quad 1 \leq \alpha \leq 2.$; the Egonet Weight Power Law (EWPL) pattern for W_i and $E_i^\beta, \quad \beta \geq 1$. Given the normal patterns, the paper takes the distance-to-fitting-line as a measure to score the nodes in the graph. The algorithm can detect anomalous nodes whose neighbors are either too sparse (Near-star) or too dense (Near-clique). By studying both the total weight W and the number of edges E, it can detect anomalous nodes whose interactions with others are extremely intensive. By analyzing the relationship between the largest eigenvalue λ and the total weight W, it can detect dominant heavy link, or a single highly active link in an egonet. The "OddBall" algorithm builds on power law properties of complex networks, which haven been verified in various real world applications. Moreover, the fitting of power law and the calculation of anomaly score is computationally efficient, which makes the algorithm a good fit for large scale network analysis. However, the algorithm would easily fail if the network does not obey the power law, then the detected anomalies would be less meaningful. Also, the paper focuses only on the static network and generalization the algorithm to dynamic network is non-trivial.

Besides the power law, **random walk** is also adapted for graph-based anomaly detection between neighbors. The general idea is that if a node is hard to reach during the random walk, it is likely to be an anomaly. Random walk calculates a steady state probability vector, each element of which represents the probability of reaching other nodes. Following the idea of random walk, [65] focuses on the anomaly detection on bipartite graph, denoted as $G = \langle V_1 \bigcup V_2, E \rangle$, where node sets V_1 has k nodes, V_2 has n nodes and E are the edges between them. It detects anomalies by first forming the neighborhood and then computing the normality scores. During neighborhood formation stage, the algorithm computes the relevance score for a node $b \in V_1$ to $a \in V_1$ as the number of times that one visit b during multiple random walks starting from a. In this case, the steady state vector represents the probability of being reached from V_1 in a *random walk with restart* model, and the algorithm detects anomalies linked to the query nodes. Random walk model stresses the graph structure while ignores the nodes' attributes. Sometimes, it might be an over-simplification of the underlying network generating process, which would lead to high false positive ratio.

[48] uses similar random walk guideline to detect outliers in a database and proposes the "OutRank" algorithm. It first constructs a graph from the objects where each node repre-

sents a data object and each edge represents the similarity between them. For every pair of the objects $X, Y \in \mathbb{R}^d$, the algorithm computes the similarity $Sim(X, Y)$ and normalizes the resulting similarity matrix to obtain a random walk transition matrix S. Then it defines the following connectivity metric based on how well this node is connected to the other nodes:

Definition.(Connectivity) Connectivity $c(u)$ of node u at tth iteration is defined as follows:

$$c_t(u) = \begin{cases} a & \text{if } t = 0 \\ \sum_{v \in \text{adj}(u)} (c_{t-1}(v)/|v|) & \text{otherwise} \end{cases}$$

where a is its initial value, adj(u) is the set of nodes linked to node u, and $|v|$ is the node degree. This recursive definition of connectivity is also known as the power method for solving eigenvector problem. Upon convergence, the stationary distribution can be written as $c = S^T c$. The algorithm detects the objects (nodes) with low connectivity to other objects as anomalies. "OutRank" solves individual activity-based anomaly detection problem using a graph-based anomaly detection method. As a general outlier detection framework, it requires the construction of the graph from data objects. Thus its performance can heavily rely on the type of similarity measurement adopted for computing the edges. Despite a wealth of theoretical work in graph theory, standard graph representation only allows each edge to connect to two nodes, which cannot encode potentially critical information regarding how ensembles of networked nodes interacting with each other [62]. Given this consideration, a generalized **hypergraph** representation is formulated which allows edges to connect with multiple vertices simultaneously. In hypergraph, each hyperedge is a representation of a binary string, indicating whether the corresponding vertex participates in the hyperedge. Figure 2 provides an example for comparing the graph and the hypergraph representation of two observations 111111000 and 000101111, with p = 9, using a graph and a hypergraph. With the graph, representing one observation of an interaction requires multiple edges. With a hypergraph, one hyperedge suffices. Due to the mapping between binary strings and hyperedges, the paper formulates the graph-based anomaly detection problem in the corresponding discrete space. [62] and [61] address the problem of detecting anomalous meetings in very large social networks based on hypergraphs. In their papers, a meeting is encoded as a hyperedge $\mathbf{x}$ and $g(\mathbf{x})$ is the probability mass function of the meetings evaluated at $\mathbf{x}$. The distribution of the meetings is modeled as a two-component mixture of a non-anomalous distribution and an anomalous event distribution $g(\mathbf{x}) = (1-\pi)f(\mathbf{x}) + \pi\mu(\mathbf{x})$, with π as the mixture parameter. Then the paper learns the likelihood of each observation using variational EM algorithm with a multivariate Bernoulli variational approximation. The likelihood is subsequently used for the evaluation of the anomalousness. Hypergraph is specifically designed for high dimensional data in the graph. It provides a concise representation of the complex interactions among multiple nodes. But the representation only applies to binary relationships where an edge is either present or missing.

[66; 15] consider using **spatial auto-correlation** to detect local spatial outliers. We categorize them as graph-based approach because the spatial neighborhood defined in those methods resembles the neighborhood defined in graph. For each point o, the paper defines the Spatial Local Outlier Measure (SLOM) as $\tilde{d}(o) * \beta(o)$ to score its anomalousness. According to their definition, $\tilde{d}$ is the "stretched" distance between the point and its neighbors and β is the oscillating parameter. SLOM captures the spatial autocorrelation using $\tilde{d}$ and spatial heteroscedasticity(non-constant variance) with β. However, when the data is of high dimensions, the concept of neighborhood becomes less well-defined. The local anomaly defined in the proposed method using local spatial statistics would suffer from the "curse of dimensionality".

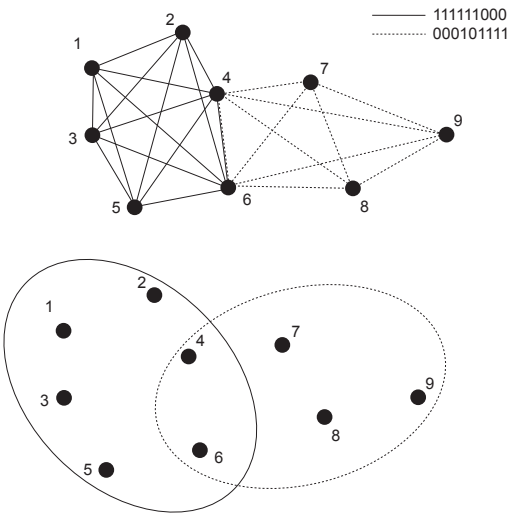


Figure 2: Modeling two observations, 111111000 and 000101111, with p = 9, using a graph (top) and a hypergraph (bottom). With the graph, representing one observation of an interaction requires multiple edges. With a hypergraph, one hyperedge suffices. The hypergraph is more efficient for storing/representing observations and more informative about the real structure of the data. [62]

Generally speaking, static graph-based approaches consider not only the activity of individual users but also their interactions. The common practice is to extract important node features from the graph, which relies heavily on feature engineering. Some algorithms import graph theoretical properties such as the power-law or the random walk into the analysis. However, those approaches usually make strong assumptions on the graph generating process, which can be easily violated in real world social networks.

2.2.2 Dynamic Graph

Social networks are dynamic in nature. Therefore, it is worthwhile to consider the problem of anomaly detection in a dynamic setting. A brief survey on dynamic network anomaly detection is elaborated in [6]. The survey characterizes the techniques employed for the problem into three groups: *Time Series Analysis of Graph Data, Anomaly Detection using Minimum Description Length, Window Based Approaches*. Based on this categorization, we review those anomaly detection approaches that incorporate the network

dynamics into their models.

Dynamic networks can be represented as a time series of graphs. A common practice is to construct a time series from the graph observations or substructures. [53] uses a number of graph topology distance measures to quantify the differences between two consecutive networks, such as weight, edge, vertex, and diameter. For each of these graph topology distance measures, a time series of changes is constructed by comparing the graph for a given period with the graph(s) from one or more previous periods. Given a graph $G = \{V, E, W_V, W_E\}$, the algorithm constructs a time series of changes for each graph topology distance measures. Each time series is individually modeled by an **ARMA process**. The anomaly is defined as days with residuals of more than two standard errors from the best ARMA model. The paper detects anomalies by setting up a residual threshold for the goodness of model fitting for time series. The proposed method in [53] is designed for change point detection. The performance of the proposed algorithm highly depends on how the graph topology distance measures are defined. Additionally, the distance measure is only able to capture the correlation between two consecutive time stamps rather than long-range dependencies.

Graph eigenvectors of the adjacency matrices is another form of the time series extracted from dynamic graph streams. In [32], the paper addresses the problem of anomaly detection in computer systems. Assume a system has N services, the paper defines a time evolving dependency matrix $D \in \mathbb{R}^{N \times N}$, where each element of the matrix $D_{i,j}$ is a function value relate to the number of service i's requests for service j within a pre-determined time interval. Given a time series of dependency matrices $D(t)$, the algorithm extracts the principal eigenvector $\mathbf{u}(t)$ of $D(t)$ as the "activity" vector, which can be interpreted as the distribution of the probability that a service is holding the control token of the system at a virtual time point. To detect anomalies, the authors define the typical pattern as a linear combination of the past activity vectors $\mathbf{r}(t) = c\sum_{i=1} Wv_i\mathbf{u}(t-i+1)$, where $\{v_i\}$ are the coefficients and c is the normalization constant. Then the algorithm calculates the dissimilarity of the present activity vector from this typical pattern. The anomaly metric $z(t)$ is defined as $z(t) = 1 - \mathbf{r}(t-1)^T\mathbf{u}(t)$. When the anomaly metric quantity $z(t)$ is greater than a given threshold, the system flags anomalous situation. Compared with representing graphs with edges, weights and vertices as in [53], eigenvectors capture the underlying invariant characteristics of the system and preserve good properties such as positivity, non-degeneracy, etc.

Besides time series analysis of the graph stream, **Minimum description length** (MDL) has been applied to anomaly detection as another way of characterizing the dynamic networks. [64] detects the change points in a stream of graph series. It introduces the concept of graph segment, which is one or more graph snapshots and the concept of source and destination partitions, which groups the source and destination nodes into clusters. Figure 3 illustrates those concepts in a three graph series. The rational behind the algorithm is to consider whether it is easier to include a new graph into the current graph segment or to start a new graph segment. If a new graph segment is created, it is treated as a change point. Given current graph segment $\mathcal{G}^{(s)}$, encoding cost c_o and a new graph $G^{(t)}$, the algorithm computes the encoding cost for $\mathcal{G}^{(s)} \bigcup \{G^{(t)}\}$ as c_n and $G^{(t)}$ as c. If $c_n - c_o < c$, the new graph is included in the current segment. Otherwise, $\{G^{(t)}\}$ forms a new stream segment and time t is a change point. To compute the encoding cost of a graph segment, the algorithm tries to partition the nodes in a segment into source and destination nodes so that the MDL for encoding the partitions is minimized. In this case, a change point indicates the time when the structure of the graph has dramatically changed. One limitation of this algorithm is that it can only handle unweighted graphs, which cannot encode the intensity of the communication between users. Thus, this method does not fit the situation when the communications of people suddenly increase while the topological structure stays unchanged. (e.g. a heated discussion starting to prevail in a social network).

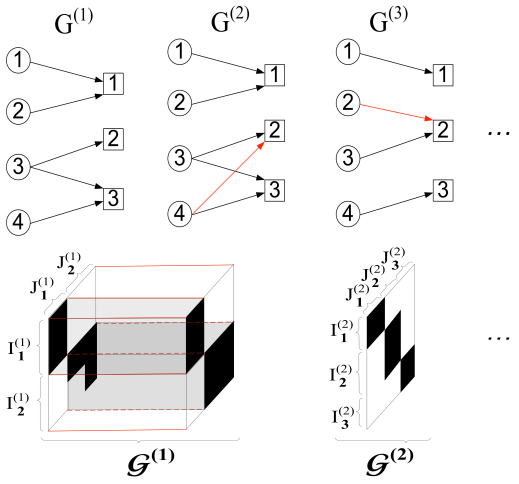


Figure 3: A graph stream with 3 graphs in 2 segments. First graph segment consisting of $G^{(1)}$ and $G^{(2)}$ has two source partitions $I_1^{(1)} = \{1,2\}$, $I_2^{(1)} = \{3,4\}$; two destination partitions $J_1^{(1)} = \{1\}$, $J^{(1)} =_2= \{2,3\}$. Second graph segment consisting of $G^{(3)}$ has three source partitions $I_1^{(2)} = \{1\}$, $I_2^{(2)} = \{2,3\}$, $I_3^{(2)} = \{4\}$; three destination partitions $J_1^{(2)} = \{1\}$, $J_2^{(2)} = \{2\}$, $J_3^{(2)} = \{3\}$. [64]

[3] addresses the categorical anomaly detection by pattern-based compression, which also adopts MDL-principle. It encodes a database with multiple code tables and searches for the best partitioning of features using MDL-optimal rule. With the natural property of code tables, the algorithm declares the anomaly by the pattern that has long code word, which are rarely used and have high compression cost. The method has been successfully generalized to a broad range of data. The use of multiple code tables to describe the data in the proposed algorithm exploits the correlations between groups of features. But the partition of the features into groups would impose unrealistic independence assumptions on the data.

For window-based approach, **scan statistics** is the mainstream method. The idea of scan statistics is to slide a small window over local regions, computing certain local statistic (number of events for a point pattern, or average pixel value for an image) for each window. The supremum or maximum of these locality statistics is known as the scan statistic. [54] specifically discusses a framework of using scan

statistics to perform anomaly detection on dynamic graphs. Specifically, the algorithm defines the scan region by considering the closed kth-order neighborhood of vertex v in graph $D = (V, E)$: $N_k[v; D] = \{w \in V(D) : d(v, w) \leq k\}$. Here distance $d(v, w)$ is the minimum directed path length from v to w in D. The induced subdigraph $\Omega(N)(N_k[v; D])$ is thus the scan region and any digraph invariant $\Psi_k(v)$ of the scan region is the locality statistics. For instance, the out degree of the digraph can be one such invariant locality statistics. The scan statistic $M_k(D)$ is the maximum locality statistic over all vertices. The algorithm applies hypothesis testing by stating the null hypothesis (normality) and the alternative hypothesis (anomaly). Digraphs with large scan statistic indicates the existence of the anomalous activity and are rejected under null hypothesis with certain threshold. Extension of scan statistics from standard graph to hypergraph representation is also examined in [52] for time-evolving graphs. The scan statistic is an intuitively appealing method to evaluate dynamic graph patterns. But one drawback of this type of method is the necessity to prespecify a window width before one looks at the data.

3. GROUP ANOMALY DETECTION

Group anomaly or *"collective anomaly"* detection in social network aims to discover groups of participants that collectively behave anomalously [12]. This is a challenging task due to three reasons: (1) we do not know beforehand any members of a malicious group; (2) the members of anomalous groups may change over time; (3) usually no anomaly can be detected when we examine individual member. Most existing algorithms can only address one or two of these challenges.

3.1 Activity-based Group Anomaly Detection

Activity-based group anomaly detection approaches usually assume that the group information is given beforehand and devote the most effort to model the activities within groups. Those approaches also imply that groups are marginal independent with each other.

[19] proposes a probabilistic model to detect group of anomalies in categorical data sets. It generalizes the spatial **scan statistic** in [54] for dynamic graphs to non-spatial data sets with discrete valued attributes. It uses Bayesian networks to model the relationship between the attributes and computes the group score for all subsets of the data S based on the model likelihood: $F(S) = \frac{P(Data|H_1(S)}{P(Data|H_0)}$. Under this definition, H_0 is the null hypothesis that no anomalies are present, and $H_1(S)$ is the alternative hypothesis specifying subset S is an anomalous group. Then it performs a heuristic search over arbitrary subsets of the data to find the groups that maximize the likelihood. At the final stage, it performs randomization testing to evaluate the statistical significance of the detected groups. For spatial data, the computation of scan statistics involves a definition of scanning region, which is often based on geographical properties. Non-spatial categorical data has the difficulty in defining local statistics based on geographical properties. Therefore, the efficient search heuristic is critical to the performance of algorithm. On the other hand, it lacks the solid theoretical justification and is sensitive to model mis-specification.

[18] considers the anomalies in categorical data sets and tries to detect anomalous attributes or combinations of attributes. The paper proposes two algorithms to test for anomalous records, i.e *Conditional Probability Test* and *Marginal Probability Test*. Conditional probability test uses conditional probability as the testing statistic. For two attributes a_t, b_t, the algorithm considers the ration $r(a_t, b_t) = \frac{P(a_t)P(b_t)}{P(a_t,b_t)}$. Marginal probability computes a quantity called the q-value, which is the cumulative probability mass of all the attributes q-val$(a_t) = \sum_{x \in X} P(x)$ where $X = \{x : P(x) \leq P(a_t)\}$. Q-value is in parallel with the p-value. This approach concerns with empirical distribution functions and is parameter free. But the underlying distributions of the attributes would heavily depend on the sample size of the data.

Another line of work formulates the group anomaly detection problem as a **density estimation** task. It imposes a hierarchical probabilistic model on the normal groups and estimates the distribution of the latent variables in the model. It evaluates the likelihood of the estimated latent variables for individual group and use it as a test statistic. The Multinomial Genre Model (MGM) proposed in [73] first investigates the problem following the paradigm of latent models. MGM models groups as a mixture of Gaussian distributions with different mixture rates. Formally, given M groups, each of which has N_m objects. MGM assumes that the object features $X_{m.n}$ are generated from a mixture of K Gaussian, $m = 1, 2, \cdots, M$, $n = 1, 2, \cdots, N_m$ with a set of stereotypical mixture rates χ. The mixture rates of the M groups belong to one of the stereotypical mixture rates in χ. Figure 3.1 depicts the graphical model of the proposed model. The method then performs Bayesian inference to estimate the density of the mixture rate for each group. Then group anomaly detection is conducted by scoring the mixture rate likelihood of each group. This method finds groups whose topic variables $\{Z_m, n\}$ are not compatible with any of the stereo- typical topic distributions in χ. In MGM, groups share the candidate topics β globally, which leads to bad performance when groups have different sets of topics. [74] further extends MGM to Flexible Genre Model (FGM) with more flexibility in the generation of topic distributions, as shown in Figure 4. The motivation of FGM is to allow each group to have its own topics. Specifically, they change the set of topics β from model hyper-parameters to random variables, depending on the genre parameter η. This extension enables the model to adapt to more diverse genres in groups.

Apart from the generative approach used in MGM and FGM, [49] takes a discriminative approach to estimate the density of the mixture model. It uses the same definition of group anomaly from [73] and represents groups as probability distributions. The authors consider kernel embedding of those probabilistic distributions. For two probabilities $\mathbb{P}_1$ and $\mathbb{P}_2$, the kernel on probability distributions is defined as $K(\mathbb{P}_1, \mathbb{P}_2) = \int \int k(x, y) d\mathbb{P}_i(x) d\mathbb{P}_j(y)$, where k is a reproducing kernel in reproducing kernel Hilbert space (RKHS). They generalize the technique of one-class support vector machine (OCSVM) for point anomaly detection to group anomaly detection. Similar to OCSVM with translation invariant kernels, the authors compute the kernel of Gaussian distributions and apply SVM in a probability measure space. Interestingly, the proposed one class support measure machine (OCSMM) algorithm has inherent correspondence to the kernel density estimation, which is theoretically more attractive. Compared with generative approaches in [73; 74], OCSMM does not make assumptions on the underlying distribution of the data and is generally less computational

expensive. However, due to the use of Gaussian RBF kernels and support vector machine, the algorithm is inevitably sensitive to the selection of kernels as well as the soft margin parameter.

[5] takes a casual approach to detect the contextual anomaly. The paper proposes to encode the variables in the Bayesian network and use probabilistic association rule to discover anomalies. The association rule builds upon two measures namely *support* and *confidence*. Support describes the prior probability of a variable while confidence represents the conditional probability. Given a state variable X and observations Y, the paper defines the two measures as follows $suppport(X = x_i) = P(X = x_i)$ and $confidence(X = x_i) = Pa(X = x_i|Y)$, where Pa is the parent nodes of X in the Bayesian network. The algorithm detects the domain specific anomalous patterns (DSAP) based on two probabilistic association rules: 1) low support and high confidence 2) high support and low confidence. Then it sorts the detected DSAPs according to sensitivity analysis scores and considers the top τ patterns with the lowest scores as output anomalies. Different from MGM or FGM, the proposed method operates on the general Bayesian network rather than a specific probabilistic model. The evaluation of support and confidence on each node is relatively cheap compared with full Bayesian inference. However, the detected causal anomalies would be ad hoc. The false positive rate would increase sharply with larger size Bayesian networks.

3.2 Graph-based Group Anomaly Detection

The most common observations we have in social networks are the individual attributes as well as ties among participants. Graph-based group anomaly detection techniques seek to jointly utilize these observations and detect anomalous groups in a unified framework.

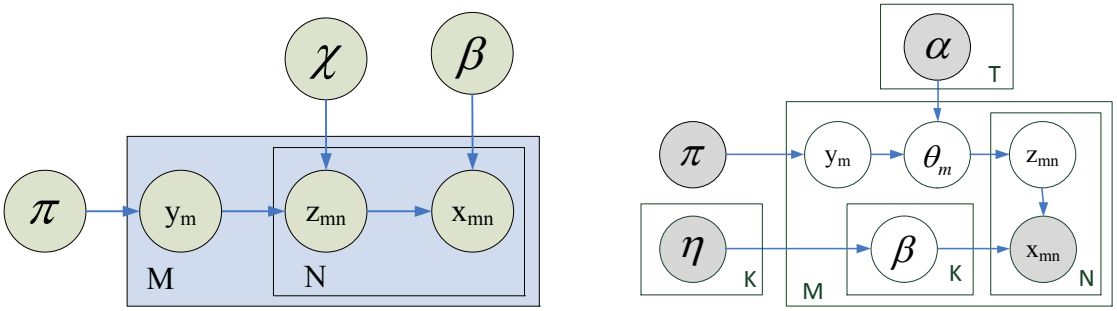
3.2.1 Static Graph

Anomalous edge detection has been proposed in [9] based on graph partitioning. The algorithm aims to detect anomalous edges that deviate from the overall clustering structure. The rationale behind this method is that if the removal of an edge can significantly make the graph easier to partition, then the two linked nodes may have an anomalous relation. The partitioning algorithm tries to find the best number of partitions so that the **Minimal Description Length** (MDL) needed to encode and transmit all the partitions of the graph is minimized. For a graph with n nodes, the paper defines the group mapping $\mathcal{G} : \{1, 2, ..., n\} \rightarrow \{1, 2, \cdots, k\}$ to assign nodes into k clusters. Thus, the *Total Encoding Cost* for the graph $T(D; k, \mathcal{G})$ in the form of the adjacency matrix $D = [d_{i,j}]$ depends on the number of the clusters k as well as the group mapping of the nodes $\mathcal{G}$. Anomalous edges are those edges whose removal would significantly reduce the total graph encoding cost. In the paper, the anomaly score of an edge is defined as the total encoding cost difference to transmit the new partitions when the edge is removed, i.e, "outlierness" of edge $(u, v) := T(D'; k, \mathcal{G}) - T(D; k, \mathcal{G})$. D and D' are equal of all edges except that $d'_{u,v} = 0$. Other similar work includes [43], which defines a *rarity* measure to discover unusual links, and [55], which uses a *Katz* measurement to statistically predict the likelihood of a link. Edge anomaly detection focus merely on pair-wise relationship and is not feasible for detecting more complicated anomalous behaviors with more than two people involved.

Finding **anomalous substructure** in graphs is another topic of attention. For example, in the scenario of email exchanges within a company, email correspondence between managers and their secretaries should be normal (frequent pattern), while email exchange between assembly line workers and secretaries could be an anomalous pattern. [50] presents an iterative expanding algorithm to look for rare substructures using their SubDue system [17]. Given a labeled graph, where each node has a label identifying its type, the system starts with a list holding 1-vertex substructures for each unique vertex label. It modifies the list by generating, extending, deleting or inserting vertices and edges. One central issue is how to measure the anomalousness of a substructure. Simply counting the number of occurrences for substructures is not enough, as larger substructures tend to have low occurrences. [50] intuitively defines a score for a substructure S in a graph G as $F_2 = Size(S) \cdot Occurrences(S, G)$, which is simply the product of the total number of nodes within a substructure and its occurrences. A smaller value of F_2 indicates a more abnormal substructure. Another issue of the problem is the computational complexity of the algorithm. Although [17] shows that in practice the system runs in polynomial time, theoretically it faces exponential number of substructures.

The pioneering work of [50] sees the rise of mining substructures in graphs. [46] leverages the structural information in the heterogeneous networks to detect unusual subgraph patterns. The algorithm encodes the graph using a tensor and focuses on finding the suspicious spikes via **tensor decomposition**. Formally, given an M-mode tensor $\mathcal{X}$ of size $I_1 \times I_2 \times \cdots \times I_M$, the algorithm performs CP decomposition of the tensor of rank R as $\mathcal{X} \approx \sum_{r=1}^{R} \lambda_r (a_r^{(1)} \times \cdots a_r^{(M)})$, where $\{a_r^{(i)}\}$ are rank-1 eigenscore vectors. The approximation would be exact when R equals the true rank of the tensor. Next the algorithm transforms the eigenscore vector plot (absolute value of eigenscore vs. attribute index) into the eigenscore histogram (absolute value of eigenscore vs. frequency count) and conducts spike detection on the histogram. The proposed approaches bridges graph mining and tensor analysis. Tensor decomposition is able to capture the complex structure in heterogeneous networks. But tensor decomposition problem itself can be NP-hard to solve. And the lack of explicit objective in the proposed anomaly detection framework would create difficulties in the final evaluation of the algorithm's performance.

In the setting of fraudulent activity detection, [22] jointly considers anomalous substructure and the criteria of MDL. Specifically, they run the SUBDUE system with MDL heuristics to find the normative pattern in the graph. Instances of substructure are evaluated against the normative pattern with a match cost. Anomalous substructures are the ones with the lowest matches. Based on this definition of group anomaly, [22] presents three slightly different algorithms, i.e. GBAD-MDL, GBAD-P and GBAD-MPS to detect anomalies. These methods first find all the instances of frequent substructures and evaluate the frequency of the abnormal structure multiplied by the match cost. A key drawback of this method is that it assumes that the degree of nodes in a graph is uniformly distributed, which is almost impossible in most social networks. As shown in [47; 10], real graphs usually follow power law degree distribution instead of uniform distribution.



(a) Multinomial Genre Model

(b) Flexible Genre Model

Figure 4: Graphical Model Representation of Multinomial Genre Model and Flexible Genre Model for activity-based group anomaly detection.

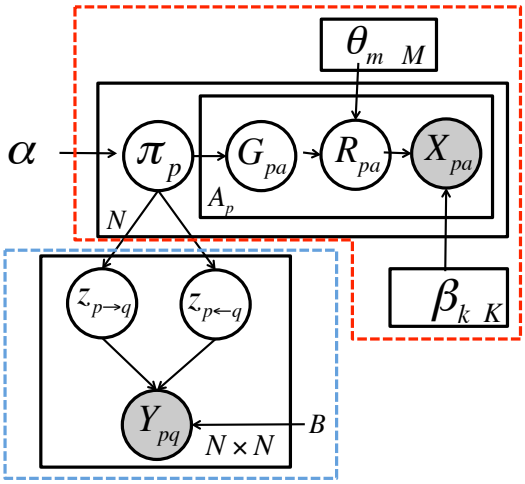


Figure 5: Plate representation for the GLAD model

In social media, two forms of data coexist: one is the point-wise data, which characterize the features of an individual person. The other is pair-wise relational data, which describe the properties of social ties. Density estimation methods for group anomaly detection [73; 74; 49] emphasize on the point-wise data and usually overlook the pair-wise relational data. Graph-based methods highlight the graph structure but usually fail to account for the attributes of individual nodes. Additionally, existing group anomaly detections algorithms are all two-stage approaches: (i) identify groups, (ii) detect group anomalies. This strategy assumes that the point-wise and pair-wise data are marginally independent. However, such independence assumption might underestimate the mutual influence between the group structure and the feature attributes. The detected group anomalies can hardly reveal the joint effect of these two forms of data.

With those considerations, [57] proposes to build an alla prima that can accomplish the tasks of group discovery and anomaly detection all at once. They develop a hierarchical Bayes model: the GLAD model, for detecting the group anomaly. The GLAD model utilizes both the pair-wise and point-wise data and automatically infers the group membership and the role at the same time. It models a social network with N individuals. Assuming that each person p is associated with a group identity G_p and a role identity R_p. By groups, it means the clusters that capture the similarity suggested by the pair-wise communications. By roles, it refers to the mixture components that categorize the point-wise feature values of the nodes. For simplification, they fix the number of groups as M and the number of roles as K. Figure 5 shows the plate notation for the GLAD model.

For each person p, he joins a group according to the membership probability distribution π_p. GLAD imposes a Dirichlet prior on the membership distribution. It is well known that the Dirichlet distribution is conjugate to the multinomial distribution. It assumes the pair-wise link $Y_{p,q}$ between person p and person q depends on the group identities of both p and q with the parameter B. Furthermore, it models the dependency between the group and the role using a multinomial distribution parameterized by a set of role mixture rate $\{\theta_{1:M}\}$. The role mixture rate characterizes the constitution of the group: the proportion of the population that plays the same role in the group. Finally, it models the activity feature vector of the individual X_p as the dependent variable of his role with parameter set $\{\beta_{1:K}\}$.

GLAD defines the group anomaly based on the role mixture rates, it scores the group anomalousness using

$$-\sum_{p\in G}\langle \log p(R_p|\Theta)\rangle_p$$

The most anomalous group will have the highest anomaly score. In practice, it approximates the true log likelihood with the variational log likelihood to get $-\sum_{p\in G}\langle \log p(R_p|\Theta)\rangle_q$. A limitation of GLAD is that it only models the static network. This might be restrictive if we want to further consider dynamic networks. Besides the anomaly group whose mixture rate deviates significantly from other groups, it is also interesting to study how the mixture rate evolves over time.

3.2.2 *Dynamic Graph*

Evolving networks can also provide insights into the temporal changes of groups. Detecting anomalously groups in dynamic graphs is more challenging, as the group structures

are not fixed and the unusual patterns in the group can also change.

[25] take a bipartite graph of individual entities and sequential ordered attributes as inputs and returns a group of entities whose attributes sequences are less likely to be generated from the proposed Markov chain model. One example of this type of anomaly is that several people constantly jump from companies to companies together. They track the complete history of employments and disclosures, and recognize the tribes that are closely related. Formally, the method requires bipartite graph $G = (R \bigcup A, E)$, where $R = \{r_i\}$ is the entity representatives, $A = \{a_j\}$ is the attributes and E are edges with time interval annotation. For each edge $e \in E$, $e = (r_i, a_j, tstart_{ij}, tend_{ij})$. The method begins by listing the co-worker relationships in the graph. Every pair $f_{ij} = (r_i, r_j)$ indicates the individuals that have worked together. This results in a new graph $H = (R, F)$, where edges in the new graph $F = \{f_{ij}\}$ is annotated with individuals attribute and history information. Then the paper defines a significance score for each edge, which measures the significance or the anomalousness of shared jobs. The algorithm proceeds by identifying significant edges and computing the significance score c for each of them. Then the proposed method picks a threshold d for the scores and prune all the edges f_{ij} for $c_{ij} < d$. After pruning, the connected components in the remaining graph (which should be quite sparse after the pruning) are regarded as anomalous groups, or tribes as referred in the paper. As also pointed by authors, the choice of scoring pairs constitutes the heart of the problem, thus posing difficulty in the selection

[71] directly analyze graph structures and efficiently track node proximity, which measures the relevance between two nodes in bipartite graphs. The paper defines a dynamic proximity score based on the probability to "random walk" from one to the other in the static graph. Low proximity to other nodes can in a way indicate anomaly. Their definition of dynamic proximity accounts for two important aspects of node relevance: proximity involves multiple snapshots of the graph; proximity does not drop over time. [71] extends this method to track anomalous nodes in time evolving graphs by defining a dynamic proximity metric. This dynamic proximity is derived from the edge and weight differences between graph snapshots and preserves a monotonicity property.

[44] proposes to detect the significant changing subgraphs. Given two consecutive snapshots of a graph G_{i-1} and G_i, the algorithm defines an *importance score* to measure the accumulative change of a node's closeness to its l-step neighbors (neighbors within l hops from the node) between two consecutive graph slices. In their context, **random walk** with restart is used to model the node relevance. The closeness of a pair of vertices v_j and v_k is defined as

$$\Pi^l(j,k) = \sum_{\tau: v_j = \rightarrow v_k; \text{length}(\tau) \leq l} p(\tau) c (1-c)^{length(\tau)}$$

where τ is a path from v_j to v_k whose length is length(τ) with transition probability $p(\tau)$. The importance score is therefore the summation of the closeness changes of v_j to the other nodes, defined as

$$VI_i(v_j) = \sum_{v_k \in V_i} |\Pi^l_{i-1}(j,k) - \Pi^l_i(j,k)|$$

Note that two consecutive graph slices G_i and G_{i+1} have the same set of nodes, but their edge set could be different. With the node closeness Π^l_i and the vertex importance score VI, the paper uses a strategy similar to density clustering to detect the significant subgraphs. Specifically, the algorithm puts the most important node in the current subgraph g, adds all of its l-step neighbors to a max-heap. As long as there exists a node whose closeness with node t exceeds certain threshold, the algorithm iteratively moves t from the heap into g. When the iteration terminates, g is regarded as the anomalous subgraph, and the algorithm proceeds to generate anomalous subgraphs for the next timestamp. The proposed algorithm detects subgraphs with significant change in edges as group anomalies. The incremental learning of nodes closeness changes makes the algorithm quite efficiently. However, the output subgraphs heavily rely on the threshold for the closeness, and there is no clear mapping between the nodes' closeness and anomalousness.

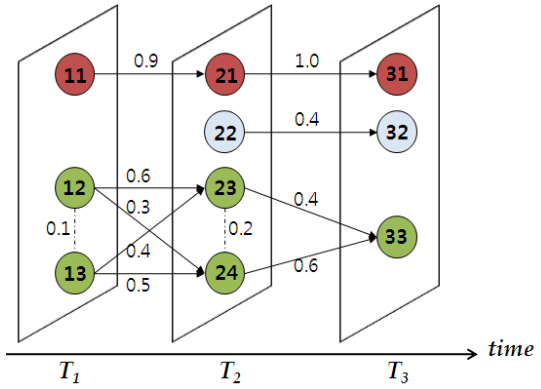


Figure 6: An example of t-partite graph constructed from a dynamic network (taken from [36]). Each circle at a time stamp T_i represents a cluster in the snapshot graph G_i.

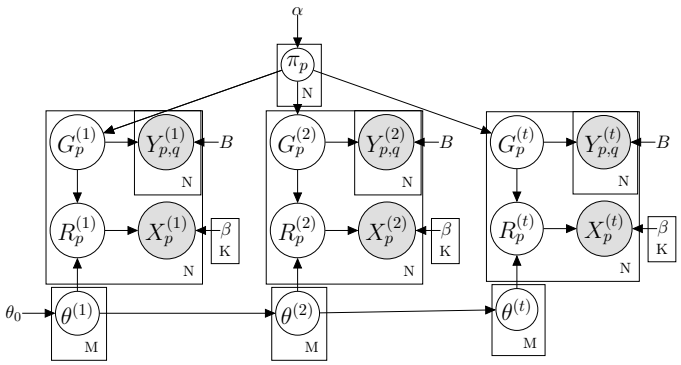


Figure 7: Plate representation for the Dynamic Group Latent Anomaly Detection (DGLAD) model

In another work which tries to detect changing communities by [36], the authors propose a two-stage density-based clustering algorithm *CHRONICLE*. The algorithm first clusters nodes in each snapshot graph G_i at time T_i using *structural similarity* σ, defined as $\sigma(v,w) = \frac{|N(v) \bigcap N(w)|}{\sqrt{|N(v)| \times |N(w)|}}$, where $N(v)$ is the neighborhood nodes of v. Then the algorithm replaces each cluster with one node to form a **t-partite**

graph, as shown in Fig. 6. In the t-partite graph, the edge weight between two nodes (dashed edges in Fig. 6) within a time stamp T_i denotes the number of edges between the two clusters, and the edge weight between two nodes from two consecutive time stamps is defined as the *Jaccard* similarity between the node sets of two clusters. In the second stage, *SCAN* is applied again on the t-partite graph. In Fig. 6 different colors represents different clusters in the t-partite graph. From these clusters we can clearly monitor the formation and dissolving of the groups, which could provide some hint on which groups are anomalously changing. However, the algorithm is originally designed for monitoring community evolution instead of anomalous changing group detection. It is not clear how the algorithm can be adapted for anomaly detection yet.

[30] also presents a two-stage method, which combines the Bayesian approach for discrete activity modeling with the graph analysis techniques for discovering anomalous structures. In the first stage identifies potentially anomalous nodes by conjugating Bayesian models for discrete time **counting processes**. Specifically, it models the number of communications made from i to j up until discrete time t denoted by $N_{ij}(t)$ as a counting process. It learns the distributions of the counts and use predictive p-value to evaluate the new observations for anomalous nodes. In the second stage, standard network inference tools are applied to the reduced subnetwork of the anomalous nodes identified from the first stage to uncover anomalous structure. Simulated cell phone communication as well as real-time press and media summary data are investigated to validate the method. This approach does not distinguish between point anomaly and group anomaly, hence hard to evaluate.

To further account for the dynamic nature of social media, [57] generalizes GLAD to the d-GLAD model as an extension for handling time series and formulate the problem as a change point detection task. The paper models the temporal evolution of the role mixture rate for each group with a series of multivariate Gaussian distributions. At a particular time point, the Gaussian has its mean as the value of the mixture rate. And the mixture rate of the next time point is a normalized sample from this Gaussian distribution. Since the model requires the mixture rate to be the parameters of a multivariate distribution over features, the authors apply a soft-max function to normalize the sample drawn from the multivariate Gaussian. The soft-max function is defined as $S(\theta_m) = \frac{\exp \theta_m}{\sum_m \exp \theta_m}$. When the total time length T equals one, d-GLAD reduces to the GLAD model. Figure 7 depicts the probabilistic graphical model of d-GLAD. The model is demonstrated successful in detecting change in topics of the scientific publications and party affiliation shift of US senators.

4. FUTURE RESEARCH

One challenge in anomaly detection is to distinguish between data errors and the "genuine" anomalies, i.e, those that were caused by the change in the underlying data distribution. As in most cases, it is very difficult to obtain the ground truth labels for the anomalies. We usually ignore the differences before conducting the anomaly detection. Only after we obtain the detection results and perform detailed analysis can we tease out the data error and recognize the "genuine" anomalies.

In summary, social media anomaly detection is still at an early stage. Most existing methods rely heavily on the specific application and self-defined anomalies. Some of the reviewed methods are originally designed for other related purposes, such as community monitoring and proximity tracking, instead of anomaly detection. In addition, many of the existing methods deal with memory-resident graphs, while real life social networks are often too large to fit into the memory. Distributed and online social network anomaly detection are two promising areas. In the case of very large social networks, new techniques for effectively summarizing the entire social network are also needed.

5. CONCLUSIONS

In this paper we present a survey of social media anomaly detection methods. Based on the type of target anomalies, these methods fall into two categories: point anomaly detection and group anomaly detection. Moreover, given the different formats of input information, they can be further classified into activity-based approaches and graph-based approaches. For the graph-based approaches, we divide the methodologies according to whether they consider the time dynamics of the social graph.

6. ACKNOWLEDGMENT

We would like to thank Dr. Sanjay Chawla from University of Sydney for his encouragement and detailed advice on the formulation of the anomaly categorization, distinguish between the data error and true anomalies. We also dedicate our acknowledgement to Dr. Huan Liu from Arizona State University who read this survey and provide valuable advice.

7. REFERENCES

[1] Tarem Ahmed, Tarem Ahmed, Boris Oreshkin, and Mark" Coates. Machine learning approaches to network anomaly detection. *In Proceedings of the second workshop on tackling computer systems problems with machine learning(SYSML)*, 2007.

[2] L Akoglu and M McGlohon. Anomaly detection in large graphs. *In CMU-CS-09-173 Technical*, (November), 2009.

[3] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos. Fast and reliable anomaly detection in categorical data. 2012.

[4] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Advances in Knowledge Discovery and Data Mining*, pages 410–421. Springer, 2010.

[5] Sakshi Babbar, Didi Surian, and Sanjay Chawla. A causal approach for mining interesting anomalies. In *Advances in Artificial Intelligence*, pages 226–232. Springer, 2013.

[6] C.C. Bilgin and B. Yener. Dynamic network evolution: Models, clustering, anomaly detection. Technical report, Technical Report, 2008, Rensselaer University, NY, 2010.

[7] David M. Blei and John D. Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 113–120, New York, NY, USA, 2006. ACM.

[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[9] Deepayan Chakrabarti. Autopart: parameter-free graph partitioning and outlier detection. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD '04, pages 112–124, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[10] Deepayan Chakrabarti and Christos Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1), June 2006.

[11] P.K. Chan and S.J. Stolfo. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD'98*, pages 164–168, 1998.

[12] V. Chandola, A. Banerjee, and V. Kumar. Outlier detection: A survey. *ACM Computing Surveys, to appear*, 2007.

[13] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection for discrete sequences: A survey. *IEEE Trans. on Knowl. and Data Eng.*, 24(5):823–839, May 2012.

[14] Varun Chandola, Varun Chandola, Arindam Banerjee, and Vipin" Kumar. Anomaly detection: A survey. 2007.

[15] Sanjay Chawla and Pei Sun. Slom: a new measure for local spatial outliers. *Knowledge and Information Systems*, 9(4):412–429, 2006.

[16] Haibin Cheng, Pang-Ning Tan, Christopher Potter, and Steven Klooster. Detection and characterization of anomalies in multivariate time series. In *SDM'09*, pages 413–424, 2009.

[17] Diane J. Cook and Lawrence B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, March 2000.

[18] K. Das, J. Schneider, and D.B. Neill. *Detecting anomalous groups in categorical datasets*. Carnegie Mellon University, School of Computer Science, Machine Learning Department, 2009.

[19] Kaustav Das, Jeff Schneider, and Daniel B Neill. Anomaly pattern detection in categorical datasets. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–176. ACM, 2008.

[20] Florence Duchêne, Catherine Garbay, and Vincent Rialle. Mining heterogeneous multivariate time-series for learning meaningful patterns: Application to home health telecare. *CoRR*, abs/cs/0412003, 2004.

[21] William Dumouchel. Computer Intrusion Detection Based on Bayes Factors for Comparing Command Transition Probabilities. Technical report, 1999.

[22] William Eberle and Lawrence Holder. Discovering structural anomalies in graph-based data. In *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, ICDMW '07, pages 393–398, Washington, DC, USA, 2007. IEEE Computer Society.

[23] Elena Erosheva, Stephen Fienberg, and John Lafferty. Mixed-membership models of scientific publications. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5220–5227, April 2004.

[24] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. *A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data*. Kluwer, 2002.

[25] Lisa Friedland and David Jensen. Finding tribes: identifying close-knit individuals from employment patterns. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 290–299, New York, NY, USA, 2007. ACM.

[26] Anup K. Ghosh and Aaron Schwartzbard. A study in using neural networks for anomaly and misuse detection. In *Proceedings of the 8th conference on USENIX Security Symposium - Volume 8*, SSYM'99, pages 12–12, Berkeley, CA, USA, 1999. USENIX Association.

[27] Valery Guralnik and Jaideep Srivastava. Event detection from time series data. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 33–42, New York, NY, USA, 1999. ACM.

[28] Steve Hanneke and Eric P. Xing. Discrete temporal models of social networks. In *Proceedings of the 2006 conference on Statistical network analysis*, ICML'06, pages 115–125, Berlin, Heidelberg, 2007. Springer-Verlag.

[29] D.M. Hawkins. *Identification of outliers*. Chapman and Hall, 1980.

[30] Nicholas A. Heard, David J. Weston, Kiriaki Platanioti, and David J. Hand. Bayesian anomaly detection methods for social networks. 11 2010.

[31] C. Horn and R. Willett. Online anomaly detection with expert system feedback in social networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 1936–1939. IEEE, 2011.

[32] Tsuyoshi Idé and Hisashi Kashima. Eigenspace-based anomaly detection in computer systems. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 440–449. ACM, 2004.

[33] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 207–216, New York, NY, USA, 2006. ACM.

[34] Wen H. Ju and Yehuda Vardi. A hybrid high-order markov chain model for computer intrusion detection. *Journal of Computational and Graphical Statistics*, 10(2):277–295, 2001.

[35] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 226–233, Washington, DC, USA, 2005. IEEE Computer Society.

[36] Min-Soo Kim and Jiawei Han. Chronicle: A two-stage density-based clustering algorithm for dynamic networks. In *Proceedings of the 12th International Conference on Discovery Science*, DS '09, pages 152–167, Berlin, Heidelberg, 2009. Springer-Verlag.

[37] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[38] Mladen Kolar, Le Song, Amr Ahmed, and Eric P. Xing. Estimating time-varying networks. *Annals of Applied Statistics*, 4(1):94–123, 2010.

[39] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. *SIGCOMM Comput. Commun. Rev.*, 34:219–230, August 2004.

[40] Theodoros Lappas, Kun Liu, and Evimaria Terzi. Finding a team of experts in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 467–476, New York, NY, USA, 2009. ACM.

[41] Ar Lazarevic, Ar Lazarevic, Aysel Ozgur, Levent Ertoz, Jaideep Srivastava, and Vipin" Kumar. A comparative study of anomaly detection schemes in network intrusion detection. *In Proceedings of the third SIAM international conference on Data Mining*, 2003.

[42] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[43] Shou-de Lin and Hans Chalupsky. Unsupervised link discovery in multi-relational data via rarity analysis. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM '03, pages 171–, Washington, DC, USA, 2003. IEEE Computer Society.

[44] Zheng Liu, Jeffrey Xu Yu, Yiping Ke, Xuemin Lin, and Lei Chen. Spotting significant changing subgraphs in evolving graphs. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 917–922, Washington, DC, USA, 2008. IEEE Computer Society.

[45] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007.

[46] Koji Maruhashi, Fan Guo, and Christos Faloutsos. Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 203–210. IEEE, 2011.

[47] Mary McGlohon, Leman Akoglu, and Christos Faloutsos. Weighted graphs and disconnected components: patterns and a generator. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 524–532, New York, NY, USA, 2008. ACM.

[48] H. D. K. Moonesinghe and Pang-Ning Tan. Outrank: a Graph-Based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools*, 17(1), 2008.

[49] Krikamol Muandet and Bernhard Schölkopf. One-class support measure machines for group anomaly detection. *stat*, 1050:1, 2013.

[50] C.C. Noble and D.J. Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003.

[51] Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 653–658, New York, NY, USA, 2004. ACM.

[52] DMY Park, C.E. Priebe, D.J. Marchette, and A. Youssef. Scan statistics on enron hypergraphs. *Interface*, 2008.

[53] Brandon Pincombe. Anomaly detection in time series of graphs using arma processes. *ASOR BULLETIN*, 24(4):2, 2005.

[54] Carey E. Priebe, John M. Conroy, David J. Marchette, and Youngser Park. Scan statistics on enron graphs. *Comput. Math. Organ. Theory*, 11(3):229–247, October 2005.

[55] Matthew J. Rattigan and David Jensen. The case for anomalous link discovery. *SIGKDD Explor. Newsl.*, 7(2):41–47, December 2005.

[56] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of pca for traffic anomaly detection. *SIGMETRICS Perform. Eval. Rev.*, 35(1):109–120, June 2007.

[57] Yu Rose, He Xinran, and Liu. Yan. Glad: Group anomaly detection in social media analysis. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014.

[58] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, pages 487–494, Arlington, Virginia, United States, 2004. AUAI Press.

[59] M. Schonlau, W. DuMouchel, W.H. Ju, A.F. Karr, M. Theus, and Y. Vardi. Computer intrusion: Detecting masquerades. *Statistical Science*, pages 58–74, 2001.

[60] Matthias Schonlau and Martin Theus. Detecting masquerades in intrusion detection based on unpopular commands. *Inf. Process. Lett.*, 76:33–38, November 2000.

[61] J. Silva and R. Willett. Detection of anomalous meetings in a social network. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, pages 636 –641, march 2008.

[62] J. Silva and R. Willett. Hypergraph-based anomaly detection in very large networks. 2008.

[63] Xiaodan Song, Ching-Yung Lin, Belle L. Tseng, and Ming-Ting Sun. Modeling and predicting personal information dissemination behavior. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 479–488, New York, NY, USA, 2005. ACM.

[64] Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 687–696, New York, NY, USA, 2007. ACM.

[65] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 418–425, Washington, DC, USA, 2005. IEEE Computer Society.

[66] Pei Sun and Sanjay Chawla. On local spatial outliers. In *Data Mining, 2004. ICDM'04. Fourth IEEE International Conference on*, pages 209–216. IEEE, 2004.

[67] Pei Sun, Sanjay Chawla, and Bavani Arunasalam. Mining for outliers in sequential databases. SIAM, 2006.

[68] Jun-ichi Takeuchi and Kenji Yamanishi. A unifying framework for detecting outliers and change points from time series. *IEEE Trans. on Knowl. and Data Eng.*, 18(4):482–492, April 2006.

[69] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 613–622, Washington, DC, USA, 2006. IEEE Computer Society.

[70] Hanghang Tong, Spiros Papadimitriou, Jimeng Sun, Philip S. Yu, and Christos Faloutsos. Colibri: fast mining of large static and dynamic graphs. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 686–694, New York, NY, USA, 2008. ACM.

[71] Hanghang Tong, Spiros Papadimitriou, Philip S. Yu, and Christos Faloutsos. Proximity Tracking on Time-Evolving Bipartite Graphs. 2008.

[72] Weng-Keen Wong, Andrew Moore, Gregory Cooper, and Michael Wagner. Bayesian network anomaly pattern detection for disease outbreaks. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*, pages 808–815, Menlo Park, California, August 2003. AAAI Press.

[73] L. Xiong, B. Poczos, J. Schneider, A. Connolly, and J. VanderPlas. Hierarchical probabilistic models for group anomaly detection. In *Proceedings of International Conference on Artifcial Intelligence and Statistics*, 2011.

[74] Liang Xiong, Barnabás Póczos, and Jeff G Schneider. Group anomaly detection using flexible genre models. In *NIPS*, pages 1071–1079, 2011.

[75] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger. Scan: a structural clustering algorithm for networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 824–833, New York, NY, USA, 2007. ACM.

[76] Dragomir Yankov, Eamonn Keogh, and Umaa Rebbapragada. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. *Knowl. Inf. Syst.*, 17:241–262, November 2008.

[77] Dianmin Yue, Xiaodan Wu, Yunfeng Wang, Yue Li, and Chao-Hsien Chu. A review of data mining-based financial fraud detection research. *2007 International Conference on Wireless Communications Networking and Mobile Computing*, (C):5514–5517, 2007.

The Internet of Things: Opportunities and Challenges for Distributed Data Analysis

Marco Stolpe
Artificial Intelligence Group, LS 8
Department of Computer Science
TU Dortmund, 44221 Dortmund, Germany
marco.stolpe@tu-dortmund.de

ABSTRACT

Nowadays, data is created by humans as well as automatically collected by physical things, which embed electronics, software, sensors and network connectivity. Together, these entities constitute the Internet of Things (IoT). The automated analysis of its data can provide insights into previously unknown relationships between things, their environment and their users, facilitating an optimization of their behavior. Especially the real-time analysis of data, embedded into physical systems, can enable new forms of autonomous control. These in turn may lead to more sustainable applications, reducing waste and saving resources.

IoT's distributed and dynamic nature, resource constraints of sensors and embedded devices as well as the amounts of generated data are challenging even the most advanced automated data analysis methods known today. In particular, the IoT requires a new generation of distributed analysis methods.

Many existing surveys have strongly focused on the centralization of data in the cloud and big data analysis, which follows the paradigm of parallel high-performance computing. However, bandwidth and energy can be too limited for the transmission of raw data, or it is prohibited due to privacy constraints. Such communication-constrained scenarios require decentralized analysis algorithms which at least partly work directly on the generating devices.

After listing data-driven IoT applications, in contrast to existing surveys, we highlight the differences between cloud-based and decentralized analysis from an algorithmic perspective. We present the opportunities and challenges of research on communication-efficient decentralized analysis algorithms. Here, the focus is on the difficult scenario of vertically partitioned data, which covers common IoT use cases. The comprehensive bibliography aims at providing readers with a good starting point for their own work.

1. INTRODUCTION

Every day, data is generated by humans using devices as diverse as personal computers, company servers, electronic consumer appliances or mobile phones and tablets. Due to tremendous advances in hardware technology over the last few years, nowadays even larger amounts of data are automatically generated by devices and sensors, which are embedded into our physical environment. They measure, for instance,

- machine and process parameters of production processes in manufacturing,
- environmental conditions of transported goods, like cooling, in logistics,
- temperature changes and energy consumption in smart homes,
- traffic volume, air pollution and water consumption in the public sector or
- puls and bloodpressure of individuals in healthcare.

The collection and exchange of data is enabled by electronics, software, sensors and network connectivity, that are embedded into physical objects. The infrastructure which makes such objects remotely accessible and connects them, is called the *Internet of Things (IoT)*. In 2010, already 12.5 billion devices were connected to the IoT [34], a number about twice as large as the world's population at that time (6.8 billion).

The IoT revolutionizes the Internet, since not only computers are getting connected, but physical things, as well. The IoT can thus provide us with data about our physical environment, at a level of detail never known before in human history [76]. Understanding the generated data can bring about a better understanding of ourselves and the world we live in, creating opportunities to improve our way of living, learning, working, and entertaining [34]. Especially the combination of data from many different sources and their automated analysis may yield new insights into existing relationships and interactions between physical entities, their environment and users. This facilitates to optimize their behavior. Automation of the interplay between data analysis and control can lead to new types of applications that use fully autonomous optimization loops. Examples will be shown in Sect. 3, indicating their benefits.

However, IoT's inherent distributed nature, the resource constraints and dynamism of its networked participants, as well as the amounts and diverse types of data are challenging even the most advanced automated data analysis methods known today. In particular, the IoT requires a new generation of distributed algorithms which are resource-aware and intelligently reduce the amount of data transmitted and processed throughout the analysis chain.

Many surveys (for instance, [3, 43, 78, 110]) discuss IoT's underlying technologies, others [37, 81] security and privacy

issues. Data analysis' role and related challenges are only covered shortly, if at all. Some surveys [1, 12, 23, 31] mention the problem of big data analysis and propose centralized cloud-based solutions, following the paradigm of parallel high performance computing. The authors of [40], [101] and [80] take a more things-centric perspective and argue for the analysis and compression of data before its transmission to a cloud. [8] identify the need for decentralized analysis algorithms, in addition. [100] present existing applications of well-known data analysis algorithms in an IoT context, highlighting decentralized data analysis as open issue concerning infrastructure. However, they do not address an algorithmic perspective.

To the best of our knowledge, our survey is the first one dealing with differences between cloud-based and decentralized data analysis from an algorithmic perspective. In Sect. 2, we elaborate on the role of data analysis in the context of the IoT. In Sect. 3, we show, how advanced levels of data analysis could enable new types of applications. Section 4 presents the challenges for data analysis in the IoT and argue for the need of novel data analysis algorithms. Like many other authors, we see the convenience and benefits of cloud-based solutions. However, we want to move further and enable data analysis even in resource-restricted situations (Sect. 5). In Sect. 6, we argue in favor of data reduction and decentralized algorithms in highly communication-constrained scenarios which existing surveys largely neglected, so far. We focus on communication-efficient distributed analysis in the vertically partitioned data scenario, which covers common IoT use cases. Section 7 presents future research directions. Finally, we summarize and draw final conclusions. The bibliography aims at providing readers with a good starting point for their own work.

2. THE INTERNET OF THINGS

The IoT consists of physical objects (or "things") which embed electronics, software, sensors, and communication components, enabling them to collect and exchange data. Physical things are no longer separated from the virtual world, but connected to the Internet. They can be accessed remotely, i.e. monitored, controlled and even made to act.

Ideas resembling the IoT reach back to the year 1988, starting with the field of ubiquitious computing. In 1991, Mark Weiser framed his ideas for the computer of the 21st century [106]. Weiser envisioned computers being small enough to vanish from our sight, becoming part of the background, so that they are used without further thinking. Rooms would host more than 100 connected devices, which could sense their environment, exchange data and provide human beings with information similar to physical signs, notes, paper, boards, etc. Devices would need self-knowledge, e.g., of their location. Many of Weiser's original ideas can still be found in current definitions of the IoT and requirements for according devices. For example, Mattern and Floerkemeier [68] enumerate similar capabilities needed to bridge the gap between the virtual and physical world. Objects must be able to communicate and cooperate with each other, which requires addressability, unique identification, and localization. Objects may collect information about their surroundings and they may contain actuators for manipulating their environment. Objects can embed information processing, featuring a processor or microcontroller, and storage capacity. Finally, they may interface to and communicate with humans directly or indirectly. In a report by Verizon [104], the IoT is defined as a machine to machine (M2M) technology based on secure network connectivity and an associated cloud infrastructure. Things belonging to the IoT follow the so called three "A"s. They must be *aware*, i.e. sense something. They must be *autonomous*, i.e. transfer data automatically to other devices or to Internet services. They also must be *actionable*, i.e. integrate some kind of analysis or control.

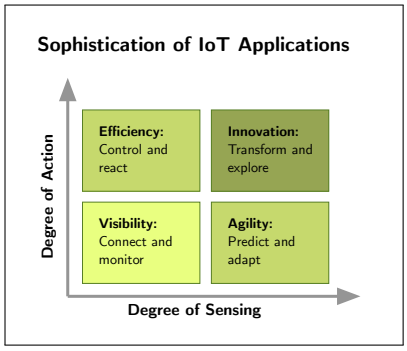


Figure 1: Sophistication levels of IoT applications [104]

The history of the IoT itself started in 1999, with the work on Radio-frequency identification (RFID) technology by the Auto-ID Center of the Massachusetts Institute of Technology (MIT) [34, 68]. The term "Internet of Things" was first literally used by the center's co-founder Kevin Ashton in 2002. In a Cisco whitepaper, Dave Evans [34] estimates that the IoT came into real existence between 2008 and 2009, when the number of devices connected to the Internet began to exceed the number of human beings on earth. Many of such devices were mobile phones, after in 2007, Steve Jobs had unveiled the first iPhone at Macworld conference. Since then, more and more devices are getting connected. It is estimated that by 2020, the IoT will consist of almost 50 billion objects [34].

The World Wide Web (WWW) fundamentally changed in at least four stages [34]. First, the web was called the Advanced Research Projects Agency Network (ARPANET) and foremost used by academia. The second stage was characterized by companies acquiring domain names and sharing information about their products and services. The "dot-com" boom may be called the third stage. Web pages moved from static to interactive transactional applications that allowed for selling and buying products online. The "social" or "experience" web marks the current fourth stage, enabling people to communicate, connect and share information. In comparison, Internet's underlying technology and protocols have gradually improved, but didn't change fundamentally. Now, connecting billions of physical things, crossing borders of entirely different types of networks poses new challenges to Internet's technologies and communication protocols. This is why the IoT was called the first evolution of the Internet [34].

As did the Internet, the IoT has the potential to change our lives in fundamental ways. Gathering and analysing data from many different sources in our environment may provide a more holistic view on the true relationships and

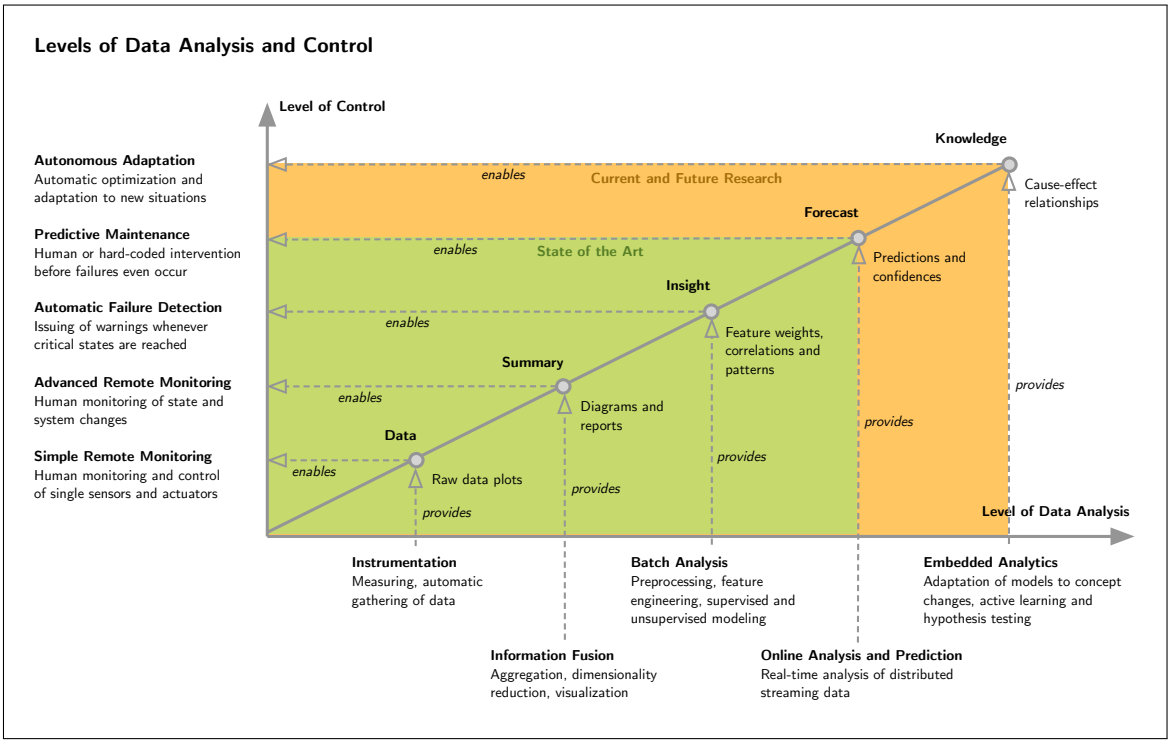


Figure 2: Relationship between data analysis and control

interactions between physical entities, enabling the transformation of raw data and information into long-term knowledge and wisdom [34]. The timely identification of current trends and patterns in the data could further support proactive behavior and planning, for instance by anticipating natural catastrophes, traffic jams, security breaches, etc. The IoT may also create new business opportunities. Potential benefits for companies are improved customer and citizen experience, better operation of machines and quality control, accelerating growth and business performance, as well as improving safety and a reduction of risk. Verizon estimates that by 2025, companies having adopted IoT technology may become 10% more profitable. Other sources predict profit increases by up to 80%. It is further estimated that the number of business to business (B2B) connections will increase from 1,2 billion in 2014 to 5,4 billion by 2020 [104]. The following section describes possible IoT applications in different sectors and points to the particular benefits that can be expected from automated data analysis.

3. DATA-DRIVEN IOT APPLICATIONS

In [25, 69], IoT applications are categorized by their level of *information and analysis* vs. their level of *automation and control*. A similar distinction is made in [104], which measures the sophistication of IoT applications by two factors, namely the *degree of action* and the *degree of sensing* (see Fig. 1). Applications falling into the lower left corner of the diagram in Fig. 1 already provide benefits given the ability to connect to and monitor physical things remotely. Giving objects a virtual identity independent of their physical location highly increases their visibility and can facilitate decision making based on smart representations of raw data. Applications located in the upper left corner of Fig. 1, in addition, use embedded actuators. Beyond pure monitoring, they enable remote control of physical things, thereby easing their management. Applications that analyse IoT generated data fall into the lower right corner of Fig. 1. Here, especially the combination of data from different physical objects and locations could provide a more holistic view and insights into phenomena that are only understood poorly, so far.

Though we agree with the previously presented categorizations, they don't show the dependency of advanced control mechanisms on data analysis. Data analysis could turn data into valuable information, which can then be utilized for building long-term knowledge and proactive decision making. Finally, merging analysis and control may lead to innovative new business models, products and services. We therefore propose the scheme in Fig. 2 which stresses the analysis. We structure the field along the dimensions of *control* and *data analysis*. The diagonal shows the milestones on the path to fully embedded analytics, which is put to good use in automatic system optimization.

The data gathered from single sensors for analysis enables simple remote monitoring applications. Here, the informed choice and placement of sensors during instrumentation depend on a well-defined analysis goal [91, 114]. Advanced applications move from the observation of single sensors to the monitoring of system and process states. This monitoring is based on the visualization of summary information obtained with the help of data analysis from multiple types of sensors and devices. The batch analysis of historical records finds correlations between features and relate them to a target value. Insights gained from this step may lead, for instance, to a better understanding of critical failure conditions and

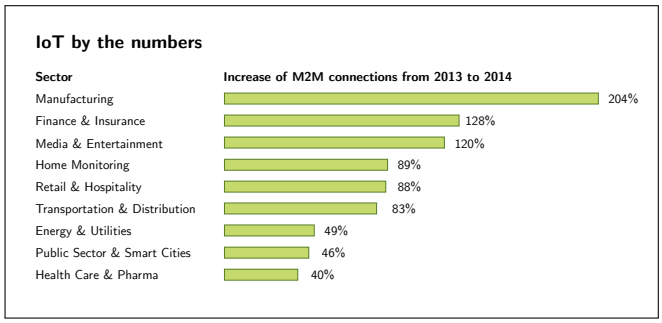


Figure 3: Increase of M2M connections in Verizon's network from 2013 to 2014 [104]

their automated detection. Prediction models derived from batch analysis may also be deployed for real-time forecasts. This is current state-of-the-art.

However, depending on the amount and rate of generated measurements, their preprocessing may become infeasable. Hence, current research focuses on distributed streaming analysis methods and the intelligent reduction of data directly at the sensors and devices themselves (see Sect. 4.3 and Sect. 5.2). Data analysis which is embedded into all parts of an IoT system will finally require the real-time derivation of models and an adaptation to changes in underlying data distributions and representations. This would in turn allow for a continuous and automated monitoring of changes in correlations. The full integration of data analysis and control introduces an automated conduction of cause-effect analysis by active testing of hypotheses, moving beyond the detection of correlations. Knowledge about causal relationships may then be used to autonomously adapt the relevant parameters in new situations. Limiting models and their use to a small selection of parameters saves memory, computing, and energy resources.

Figure 3 shows the increase of M2M connections for different business sectors in Verizon's network from 2013 to 2014. In the following, we present examples of specific IoT applications from the sectors mentioned at the beginning: Manufacturing, transportation and distribution, energy and utilities, the public sector and smart cities, as well as healthcare and pharma. We have ordered examples of each sector according to the different levels of data analysis and control as shown in Fig. 2 and have identified three main application types: Predictive maintenance, sustainable processes saving resources and quality control.

3.1 Manufacturing

The manufacturing sector supports the development of IoT by the provision of smart products. For instance, 43 million wearable bands were shipped in 2015 [20], and it is estimated that 20 million smart thermostats will ship by 2023 [74]. By 2016, smart products will be offered by 53% of manufacturers [77].

The sector not only produces devices, but also uses IoT technology itself. According to Fig. 3, the manufacturing sector is seeing the largest growth in terms of M2M connections in Verizon's network. Following the levels of Fig. 2, we now present types of industrial applications.

Simple remote monitoring applications increase visibility by embedding location-aware wireless sensors into products and wearables [104]. This allows for a continuous tracking of persons and assets, like available stock and raw materials, on- and offsite over cellular or satellite connections. [104] further mentions sensors which can detect hazards or security breaches by the instrumentation of products and wearables. Embedding sensors into production machinery will allow for the monitoring of individual machines with high granularity along the process chain. It should be added, however, that the automatic detection of such events necessarily requires an analysis and interpretation of measurements.

The aggregation of data from the same type of sensors supports the confidence in the accuracy of analysis results. Moreover, the fusion of data from different types of sensors advances remote monitoring of larger units, like systems, processes and their environment. For instance, [91] visually identify and quantify different types of productions modes in steel processing by summarizing multi-dimensional sensor data with algorithms for dimensionality reduction.

Models derived from heterogenous data sources by batch analysis may provide insights into the correlations between multiple dimensions of process parameters and a target value. According to [104], the timely identification of failure states can lead to less disruption and increase uptime in comparison to regular human maintenance visits and inspections. It should be added that once trained, data analysis models can often be made directly operational, and be used, for instance, for the automatic detection of critical patterns. For instance, learned models may be deployed early in the process for the automatic real time prediction of a product's final quality [91], allowing for timely human intervention. Here, resources might be saved by omitting further processing of already defect products. Based on human knowledge, control parameters might be adjusted such that a targeted quality level can still be reached. In the context of maintenance, the quantity to be predicted is machine wear or failure. The timely detection of anomalies and machine wear can help with reducing unplanned downtime, increasing equipment utilization and overall plant output [91, 104]. However, depending on the amount of generated data, batch analysis as well as preprocessing all data in real-time can be challenging [94]. Advanced applications therefore require the development of new kinds of data analysis algorithms (see Sect. 4.3 and Sect. 5.2).

Making data acquisition and analysis an integral part of production systems could finally allow for the long time observation of changes in correlations between process parameters and target variables. The importance of manufacturing for the adoption of IoT is emphasized by the German initiative "Industrie 4.0". It fosters the integration of production processes, IoT technology and cyber-physical systems into a so called *smart factory*. In this future type of factory, products can communicate with their environment, for instance with other products, machines and humans. In contrast to fixed structures and specifications of production processes that exist today, Reconfigurable Manufacturing Systems (RMS) derive case-specific topologies automatically based on collected data [16]. Hence, production will become more flexible and customized. Reactions to changes in customer demands and requirements may take only hours or minutes, instead of days. RMS might further support the active testing of hypotheses and targeted generation of new observations. The resulting variability of large numbers of observations

might then help with automatically distinguishing between random correlations of parameters and those the target variables truly depend on. Such knowledge could then be used for the automatic optimization and autonomous real-time adaptation of production processes and their parameters to new situations. The intelligent combination of data analysis and control can thereby lead to more sustainable systems which allow for major reductions in waste, energy costs and the need for human intervention [25, 91].

3.2 Transportation and Distribution

The sector of transportation and distribution belongs to the early adopters of IoT. Here, according to [104], important factors for the adoption of IoT technology are regulations and competition which force higher standards of efficiency and safety, as well as expectations of greater comfort and economy. From 2013 to 2014, the sector has seen a 83% increase of M2M connections in Verizon's network (see Fig. 3).

The instrumentation of vehicles enables simple remote monitoring applications that make it easier to locate and instruct fleets of cars, vans or trucks [45]. Logging driver's working hours, speed and driving behavior can improve safety and simplify compliance with regulations [104]. Customers can be regularly informed about the delivery times of anticipated goods. Even containers themselves are now equipped with boards of very restricted capacities, which open up opportunities of tracing and organizing the goods in a logistic chain of storage and delivery [103].

Another example for new types of applications is the UBER smartphone app which indicates the location of passengers calling a taxi to nearby drivers and uses surge pricing to fulfill demands for more taxis.

Advanced remote monitoring applications use data analysis to aggregate data and may provide summaries of fleet movements on a larger scale, like the average number of vehicles traveling certain routes, thereby facilitating resource planning [53].

Instrumentation allows car manufacturers deeper insights into the use of their cars. Models derived by the batch analysis of data gathered from many cars could automatically be deployed inside cars to identify or predict failure conditions. These models may also provide information about the relationships between failures and underlying causes. According to [104], such information would allow to pre-emptively issue recalls, improve designs to iron out problems, and better target new features to driver and market preferences. Intelligence built into vehicles, like proximity lane sensors, automatic breaking, head lamps, wipers, and automated emergency calls can increase road safety [104].

Advanced applications, like autonomously driving vehicles [5], require the embedded real-time analysis of data directly inside the vehicle. In addition, information sent by nearby infrastructure, like traffic signals, traffic signs, street lamps, road works or local weather stations might be taken into account (see also Sect. 3.4). For navigation, vehicles may remotely access current information on street maps.

At a larger scale, data gathered from many vehicles and infrastructure could be analysed and used to instruct vehicles beyond their individual driving decisions. [54] developed a sophisticated distributed analysis of local data from vans of a fleet, which allows to manage the overall fleet. Work orders can be allocated in real-time more efficiently, adopting to drivers, reacting to order changes, or other events. The effects on cutting fuel costs, leading to more sustainable vehicles and distribution systems has been shown [72]. Similarly, through timely diagnostics, predictive analytics, and the elimination of waste in fleet scheduling, the rail industry is looking to achieve savings of 27 billion dollars globally over 15 years [35].

3.3 Energy and Utilities

In the sector of energy distribution, IoT applications range from telematics for job scheduling and routing, to bigger ones extending the life of electricity infrastructure [104]. According to Fig. 3, the energy sector has seen an estimated growth of 49% in the number of M2M connections from 2013 to 2014.

Concerning remote monitoring, the energy sector was the first to introduce SCADA (supervisory control and data acquisition). Smart meters increase visibility by providing more granular data. Thereby they reduce the incovenience and expense of manual meter readings or estimated bills. Further, advanced remote monitoring provides more accurate views of capacity, demand and supply over different smart homes, made possible by visualizing summary information obtained from data analysis [55, 65, 116]. Based on such information, sustainability may be improved through better resource planning and cutting energy theft. According to [104], in 2014, 94 million smart meters were shipped worldwide and it is predicted that by 2022, the number of smart meters will reach 1.1 billion. One target of the European Union is to replace 80% of meters by smart meters by 2020, in 28 member countries.

Beyond monitoring applications, the batch analysis of data from smart homes may help with giving recommendations for saving energy and enable more sophisticated energy management applications [116]. Oil and gas companies can cut costs and increase efficiency by early predicting the failure of artificial components, local weather conditions, and the automated start up and shutdown of equipment [104]. On a larger scale, the smart grid connects assets in the generation, transmission and distribution infrastructures. Especially in recent years, energy use has become harder to predict, due to a decentralization of energy production. The prediction of wind power [99] and photovoltaic power [108] is important in order to better understand grid utilization. Data analysis may increase efficiency and optimize the infrastructure [55]. The embedded real-time analysis of data could enable even more sustainable distributed energy generation models in which highly autonomous systems react dynamically to changes in energy demand and distribute energy accordingly.

3.4 Public Sector

In the public sector, M2M connections have grown by 46% from 2013 to 2014 according to Fig. 3. It is estimated that by 2050, 66% of humans will live in urban areas [102] and 75% of world's energy use is taking place in cities [104]. The IoT promises the delivery of more effective services to citizens, like citizen's participation, controlling crime, the protection of infrastructure, keeping power and traffic running, and building sustainable developments with limited resources [104]. The IoT thus enables municipal leaders to make their communities safer and more pleasant to live, and to deal better with demographic changes [104].

The instrumentation of cities with sensors may lead to more

sustainable resource usage by simple remote monitoring applications. For instance, currently it takes 20 minutes on average to find a parking space in London [104] and 30% of congestion in cities is caused by people looking for a parking space [84]. The smart city of Santander [86] has instrumented, among others, parking lots. Their space utilization could be tracked and provided as information to smart phone apps. Advanced applications may also identify trends and anomalies in parking data [115]. Similar tracking apps could support car-sharing or unattended rental programs that offer on-demand access to vehicles by the hour [104]. More advanced remote monitoring applications could indicate the crowdedness of neighboring cities by aggregating data with the help of data analysis. Using real-time analysis, they might as well give direct recommendations, for instance which city to visit for more relaxed shopping.

Resource savings can also be expected from a more sustainable management of water. IBM offers an intelligent software for water management that uses data analysis for visualization and correlation detection [50]. According to IBM, the software helps to manage pressure, detect leaks, reduce water consumption, mitigate sewer overflow and allows for a better management of water infrastructure, assets and operations.

Currently, up to 40% of municipal energy costs come from street lighting [109]. The European Union has set a target to reduce CO_2 emissions of professional lighting by 20 million tons by 2020 [104]. Predictive models obtained through data analysis enable smart streetlights that automatically adjust their brightness according to the expected volume of cars and weather conditions. In a case study it was shown that the city of Lansing, Michigan, could thereby cut the energy and maintenance costs of street lighting by 70% [88, 104].

Further resources might be saved by using more intelligent transportation and traffic systems. Predicting traffic flow on the basis of past data that has been measured by sensors in the streets offers drivers an enhanced routing. The German government estimated a daily fuel consumption in Germany due to traffic jams of 33 millions of liter, a waste of time in the range of 13 million hours and concludes that traffic jams are responsible for an economic loss of 259 million Euro per day. For instance, the SCATS system [83] provides traffic flow data for different junctions throughout Dublin city. Simple remote monitoring can provide data about the current traffic flow to individual drivers by plotting counts of cars on a digital street map. The batch analysis of traffic data could help with determining factors causing traffic jams, which in turn might be used by traffic managers to adapt the street network accordingly. For the City of Dublin, traffic forecast derived from a spatio-temporal probabilistic graphical model, was exploited for smart routing [62]. In the future, such recommendations may be as well given to autonomously driving vehicles (see also Sect. 3.2).

Embedding data analysis everywhere in a city and combining the data from multiple heterogenous systems and other cities may even provide larger value. Such combination could provide a holistic view of everything, like energy use, traffic flows, crime rate and air pollution [104]. Correlations and relationships between seemingly unrelated variables are not necessarily obvious. For instance, according to the broken windows theory, the prevention of small crimes such as vandalism helps with preventing more serious crimes. However, critics state that other factors have more influence on crime rate. Up to now, such theories are hard to test and validate, since studies conducted by humans can only focus on a limited number of influence factors and might be biased. The instrumentation of many different cities and areas could increase the number of observations and help with obtaining more objective and statistically significant results. Long time observation of many different variables and active hypothesis testing, for instance by giving recommendations to city planners, may help with the detection of causes that underly phenomena. The insights gained may then enable better policy decisions.

3.5 Healthcare and Pharma

According to Fig. 3, healthcare has seen the smallest growth in M2M connection from 2013 to 2014. Similarly, Gartner estimates that it will take between five and 10 years for a full adoption of the IoT by health care. This slow adoption rate may be explained by strict requirements for keeping data of patients private and secure [42], with the IoT posing many challenges for privacy and security (see also Sect. 4). Despite such difficulties, the number and possible impact of IoT applications in healthcare is large.

The instrumentation of healthy citizens as well as patients, devices or even whole hospitals with different kinds of sensors enables different kinds of remote monitoring applications. It starts with consumer-based devices for personal use. In two years, there will be 80 million wearable health devices [42], like fitness trackers and smart watches. New kinds of devices are able to monitor not only the number of steps taken or calories, but also pulse rate, blood pressure, or blood sugar levels. The aggregation of these kinds of different information requires data analysis [36]. Monitoring might promote healthy behavior through increased information and engagement [57]. In addition, physicians may get more holistic pictures of their patients' life styles, which eases diagnosis [18].

Monitoring can be done remotely and continuously in real time, beyond office visits, with patients staying at home [18, 25, 70]. Emergencies can be detected early, like with breath pillows for children or Ion mobility spectrometry combined with multi-capillary columns (MCC/IMS) that can give immediate information about the human health status or infection threats [47]. In the case of chronic illnesses, practitioners get early warning of conditions that would lead to unplanned hospitalizations and expensive emergency care [25, 42,45,57]. Monitoring alone could reduce treatment costs by a billion dollars annually in the US [25]. According to [42], estimates show a 64% drop in hospital readmissions for heart failure patients whose blood pressure and oxygen saturation levels were monitored remotely. Similarly, at-risk elderly individuals may longer stay in their own homes. Here, remote monitoring can reassure loved ones by detecting falls or whether an indivdual got out of bed in the morning, or whether an individual took his or her medicine [57].

Monitoring may as well help with drug management and the detection of fraudulent drugs in the supply chain, by incorporating RFID tags in medication containers and finally embedding technology in the medication itself [45]. In hospitals, medical equipment like MRIs and CTs can be connected and remotely monitored, helping with maintenance, replenishing supplies and reducing expensive downtime [42].

While conditions based on a few measurements may be detected automatically based on hard-wired rules, the detec-

tion of more complex patterns necessarily requires the analysis of data.
Data analysis is also needed, if we want to identify critical patterns in patient's vital parameters [51, 60] or in movements through hospitals and optimize flow [42]. The analysis of multi-dimensional data is necessary for discovering dependencies between many variables, like, e.g., the duration of treatments and waiting times at other wards. Data analysis provides doctors with insights of scientific value, taking the data gathered by many individuals as population-based evidence [57]. Clinical and nonclinical data of larger population samples may help to understand the unique causes of a disease. Finally, data analysis that was directly embedded into devices like electrocardigrams (ECG) or wireless electrocardiograms (WES) could help with the detection of emergency cases in real-time [24].

4. DATA ANALYSIS CHALLENGES

The previous section has given many examples of applications in diverse sectors, showing that advanced levels of control not only require the instrumentation of devices, but also an analysis of the acquired data. These examples support our view expressed in Fig. 2 that it is data analysis which enables advanced types of control. Unfortunately, the IoT poses new challenges to data analysis. The following sections present problems in terms of security and privacy, technical issues as well as algorithmic challenges which require research on new types of data analysis methods.

4.1 Security and Privacy

Despite IoT's anticipated positive effects, it also poses risks for our security and privacy. Especially sectors that deal with highly personalized information, such as healthcare (see Sect. 3.5), require according means for the secure and privacy-preserving processing of data. Apart from having to make existing data analysis code more secure, analysis can as well provide solutions to decrease existing threats.

Security. The biggest security risk of IoT stems from its biggest benefit, namely the connection of physical things to a global network. In the past, security breaches were mostly restricted to the theft and manipulation of data *about* physical entities. However, the IoT allows for a direct control of the physical entities *themselves*, many of which belonging to critical infrastructures in sectors previously mentioned. Without security measures, malware like viruses could easily spread through many of IoT's connected networks, potentially resulting in disasters at a global scale [32, 37].
Data analysis algorithms can be made secure by design. However, existing code bases weren't necessarily designed and implemented with security in mind. In the past, algorithms could be expected to run mostly in environments which weren't publicly accessed. Further, the way how data has been input into analysis software was relatively controlled. With the IoT, analysis code will run on devices directly exposed to an open network environment and is thus suspectible to malicious hacking attempts. It will be much harder to ensure that data originates from trustworthy sources and is in appropriate format. Hackers might gain access to sensors and other embedded devices [32, 37, 81], or install rogue devices that interfere with existing network traffic [81]. Hence, it becomes more and more important to make data analysis code more robust by penetration testing [33] and differentiate hacking attempts from usual sensor failure. Also, legal liability frameworks must be established for algorithms whose decisions are fully automated [25].
At the same time, data analysis might provide solutions for the automatic detection or even prevention of security breaches. For instance, outlier and novelty detection algorithms which examine deviations from normal behavior have already been used successfully in fields like intrusion or malware detection [10, 17].

Privacy. Another of IoT's challenges is the protection of citizens' privacy. As Mark Weiser already stated in 1991, "hundreds of computers in every room, all capable of sensing people near them and linked by high-speed networks, have the potential to make totalitarianism up to now seem like sheerest anarchy" [106]. Since it became known that intelligence agencies of democratic states are spying at other friendly states and their citizens [96], the topic of privacy has developed an especially high brisance. It also plays a large role in business sectors where data is highly personalized. For instance, data in healthcare must be especially protected.
One problem is that with small embedded devices vanishing from our sight, people might not even recognize that data about them is getting acquired. Further, it may not be entirely clear how data given away will be combined later on and what can then be derived from it. For instance, as research on learning from label proportions [79, 93] suggests, information that seems harmless all by itself, like public election results, may become problematic once it is combined with data from other sources, such as social web sites.
It is important to mention, however, that several of the aforementioned benefits from data analysis can be achieved without highly personalized data [41]. For instance, disease research based on population-based evidence (see Sect. 3.5) would yield the same results with anonymized observations. If that doesn't suffice and enough samples are present, data can further be aggregated to guarantee k-anonymity [95]. Related is the problem of learning from label proportions [79, 93]. Where more privacy is needed, the challenge consists of developing distributed analysis algorithms that derive a model without exchanging individualized records between different networked nodes (for instance, see [27]).

4.2 Technical Challenges

Technical challenges of IoT mainly concern networking technology, devices interoperability, as well as increasing the lifetime and range of wireless battery-powered devices. Here, we list the technical problems that every application of data analysis has to face.

Data Understanding. One envisioned scenario for the analysis of IoT generated data is that as people connect new devices to the IoT, their data is automatically getting analysed, together with the data of other already existing devices. Data analysis being successful, however, depends much on the correct preprocessing of data, which in turn depends on the types and ranges of features of observations. This information can be estimated from the data. However, it can be difficult to assess the quality of such estimations without ground truth. For instance, outlier detection al-

gorithms may indicate measurements which occur only seldom. However, without additional background knowledge provided by experts, it is impossible to determine automatically if values are still inside physically meaningful ranges or caused by sensor failure. Similarly, peak detection algorithms might wrongly identify noise as relevant patterns. These problems could easily be solved if manufacturers made their sensors and embedded devices queryable and provided meta data, e.g. meaningful ranges and noise levels of theirs sensors.

Standardization. The ability to query sensors and devices for meta information requires standardized protocols. A similar standardization is needed for the exchange of raw data. Especially in industry, closed systems with proprietary data formats complicate the exchange of data between distributed components and make automated data analysis unnecessarily difficult [91]. Similarly important would be a standardization of user interfaces for data analysis tools. As Mark Weiser already noted in [106], technology becomes unobtrusive once its user interfaces are as uniform and consistent as possible. In contrast, today the user interface of operating systems and applications often is their most distinguishing property and therefore a unique selling point. Hence, a wide adoption of common standards requires that profits made from IoT technology outweigh potential losses caused by the lacking individualization of products.

Porting existing code bases. As Sect. 4.1 already discussed, existing code bases for data analysis must be made more robust to operate in hostile network environments. In addition, as more and more data analysis algorithms can be expected to run directly on embedded and mobile devices, existing code and related libraries need to be ported to these platforms. The implementation language of choice for embedded devices is C/C++. In contrast, much data analysis code is written in Java and Python, whose virtual machines and interpreters require too many resources to run on small embedded devices like sensors. Currently, the same algorithms must therefore be implemented in many different versions, making the reuse of existing code more difficult. Beyond modification of existing code bases, the IoT poses several challenges that require research on new algorithms, as described in the next section.

4.3 Algorithmic Challenges

Manual inspection of IoT generated data is possible only in simple cases. Normally, since the amount of data generated by single sensors becomes too high, the analysis needs to be fully automated. Further, the combination of data from many heterogenous sources leads to high-dimensional datasets that cannot be easily visualized or examined by humans.

Automated data analysis methods have been developed in the fields of signal processing and computer vision [29], statistics [46], artificial intelligence [82], machine learning [71], data mining [44] and databases [39], to name just some text books. Among them are sophisticated methods that can generalize over raw data, deriving *models* that describe patterns and relationships which statistically hold on expectation also for unseen observations. Such methods will be called *learning algorithms* in the following. Unsupervised learning algorithms find general patterns and relationships in the data. Supervised algorithms find such patterns in relation to a specified target value, which at best should be given as label for each observation. The difficulty in both cases is that the model must be derived only from a given finite *sample* of the data, while the probability distribution generating the data is unknown (for a more formal definition of the problem, see [46]). Many learning algorithms assume the sample to be given as a single batch which can be processed in a random access fashion, potentially making several passes over the data. Observations are assumed to have a relatively homogenous structure and fixed representation.

The IoT poses new challenges to data analysis. At the data generating side, devices are often highly resource-constrained in terms of CPU power, available main memory, external storage capacity, energy and available bandwidth. Algorithms working at the data generating side must take these constraints into account. Also the underlying data distribution may change which is known as *concept drift* [117]. For instance, due to wear, the accuracy of sensors may decrease. At the receiving side, e.g. a data center, the combination of data from many different sources may create huge masses of heterogenous data. It is estimated that in total, the IoT will generate 4.4 trillion GB by 2020 [75]. Hence, the problem consists of having to analyse *big data* [67, 76], which is characterized by large *volume* (terabytes or even petabytes of data), *heterogenity* (different sources and formats) and *velocity* (speed of generated data). High volume and velocity prohibit several passes over the data, and thus require new types of algorithms. In addition to the big data problem, the analysis of IoT data are distributed and asynchronous. Just to illustrate an effect of this particular setting, let us look at IoT devices dynamically entering or leaving the network. This contradicts an assumption underlying almost all data analysis approaches, namely that the representation of observations, e.g. the number of features, does not change over time.

5. DISTRIBUTED DATA ANALYSIS

The requirements of algorithms for the analysis of IoT generated data are largely determined by the hardware and network environment in which they are expected to run. Depending on volume and rate of data generation, as well as the particular analysis problem, data must either be already preprocessed and analyzed at the generating side, on network middleware or sent to a data center. Each scenario comes with its own set of advantages and disadvantages, constraints and particular challenges. Based on specifications found on websites of cloud providers and manufacturers, we have compiled a list of computing environments and device's properties for a quick and easy comparison in Fig. 4.

The current focus is on the centralization of data in the cloud and its analysis by high performance computing [19, 23, 31, 43, 76]. Cloud computing allows for highly scalable distributed systems that solve tasks in parallel by means of virtualization. Virtual instances of nodes in a network are independent from the particular physical nodes they run on. Hence, new instances can easily be added and removed depending on current computational demands. Computation follows the paradigm of parallel computing in so far as modern frameworks shield programmers as much as possible from the intricate details of distributed systems. For

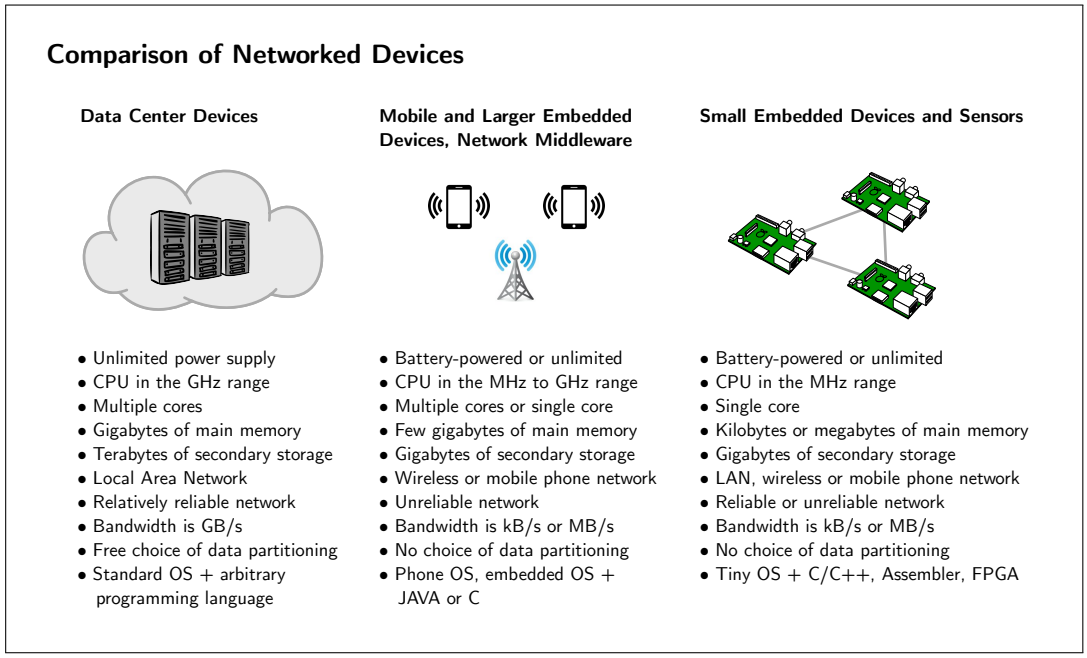


Figure 4: Comparison of computing environments and device types

instance, the scheduling and execution of code, the creation of threads or processes, synchronization as well as message passing are handled automatically. Failures that can occur in distributed systems are taken care of by redundancy and the automatic rescheduling of processes. The main task for programmers is to divide their problem into smaller subproblems which can be worked on in parallel. How and where code is executed is mostly transparent, giving the impression of a single big machine instead of many nodes.

As more and more devices are getting connected, existing network hardware and infrastructure will no longer suffice to handle the expected network traffic [19, 25, 26, 70, 76]. Whenever the rate of data generation is higher than available bandwidth, data must be analysed on the generating devices themselves or at least be reduced *before* transmission into the cloud [40, 80, 101]. In the following, algorithms that process or analyse data directly where it is acquired will be called *decentralized*. In case they need another node for coordination, data and computation are at least splitted between local nodes and the coordinator. Decentralized algorithms which need no coordinator and exchange information only with local peer nodes will be called *fully decentralized*. Ideally, decentralized analysis algorithms should exchange less information than all data between nodes.

The next section presents the ideas and constraints of current cloud-based data analysis approaches in more detail, while the following section discusses the need for decentralized data analysis algorithms in more communication-constrained scenarios.

5.1 Data Centers and Cloud Computing

One option for the analysis of IoT generated data is its centralization at a data center. Cloud computing solutions are offered by different service providers. They allow for an easy and cost-efficient upscaling of computing and storage resources. Depending on the rate of data generation, there exist two different models of data processing: Data may either be stored and analysed as a batch, or it must be processed directly as a stream.

Batch analysis. Huge data masses which do not fit in one server require the distribution of data over different connected storage devices. This is, for instance, accomplished by saving chunks of arriving data in a distributed file system such as HDFS [85]. Once the data is stored, it can be analysed as a batch by distributed algorithms that solve tasks cooperatively. Each machine in a data center may have multiple cores, which algorithms can exploit for parallel execution. CPUs are in the gigahertz (GHz) range and main memory has several gigabytes. Machines are usually connected in a local area network (LAN) where connections are relatively reliable. Technologies such as Infiniband and 100 Gigabit Ethernet allow for high bandwidths which are comparable to direct main memory accesses. Reading from dynamic random access memory (DRAM) can be about one order of magnitude faster than reading from external storage mediums, like solid-state drives (SSDs). A reorganization of data would therefore be an expensive operation. Hence, it is desirable to read data from disk only once. This can be achieved by moving code to the machine storing the data and executing it locally.

The distributed batch analysis of data is currently supported by different frameworks. Hadoop [107] is a popular framework. It follows the map and reduce paradigm known from functional programming, where the same code is executed on different parts of the data and the results are then merged. Map and reduce is especially well-suited for problems that are data parallel. This means that tasks can work independently from each other on different chunks of data, reading it only once, without synchronization or managing state. The

paradigm lends itself well for data analysis algorithms which process subsets of observations or features only once. Some algorithms for counting, preprocessing and data transformation fall into this category.
More advanced data analysis algorithms, especially learning algorithms, often require the combination of data from different subsets. They also need to make several passes over the data, and synchronize shared model parameters. For instance, the k-Means clustering algorithm [64] repeatedly assigns observations to a globally maintained set of centroids. Similarly, many distributed optimization algorithms used in data analysis maintain a globally shared set of model parameters (see also [13]). In map and reduce, distributed components are assumed to be stateless. One way to maintain state between iterations would be to access, for instance, a database server which is external to the Hadoop framework. However, this would require the unnecessary and repeated transmission of state over the network. For the implementation of stateful components, lower level frameworks like the Message Passing Interface (MPI) [2] or ZeroMQ [49] are usually better suited. These frameworks allow for long running stateful components and full control over which data is to be sent over the network.
Distributed variants of well-known data analysis algorithms, like k-Means clustering [64] and random forests [15], have been implemented in the Apache mahout [98] framework that works on top of Hadoop. However, the framework contains only few algorithms, as research on distributed data analysis algorithms for high performance computing is still ongoing.

Analysis of streaming data. Whenever batch processing isn't fast enough to provide an up-to-date view of the data, it must be processed as a stream [9,26]. The Lambda architecture by Marz [67] is a hybrid of batch and stream processing. The batch layer regularly creates views on historical data. The speed layer processes current data items which come in while batch jobs are running, and creates up-to-date views for this data. Both views are combined at a service layer, which provides a single view on the data to users. A disadvantage of the Lambda architecture is that algorithms must be designed and implemented for different layers. Kreps [58] therefore proposed the Kappa architecture, in which all data is treated as a stream.
Several frameworks support the development of streaming algorithms (for one framework and an overview, see [9]). Related analysis algorithms are still an active area of research [38] and are currently implemented in different frameworks [7, 30, 97].

The centralization of all data in the cloud offers several benefits. The often complicated network infrastructure needed for distributed computing as well as the corresponding machines are fully managed by the provider. Due to providers' expert knowledge, security risks might decrease. Customers pay only for those services they really use, such that it becomes easier and less costly to accomodate for spikes in network traffic. As long as the data analysis algorithms to be executed and their components can be fully parallelized, scalability is just a matter of adding new machines.
However, the centralization of all data also poses risks for privacy and may have disadvantages. In the case of data theft, all data may suddenly become accessible. Further, the cloud itself poses a single point of failure. Whenever data is generated at a higher rate than can be transmitted, either due to a limited bandwidth or high latency, the cloud can become a bottleneck for real-time analysis and control. Such cases require the local processing and reduction of data directly at the data generating side, as argued for in the next section.

Table 1: Data transfer rates of different technologies

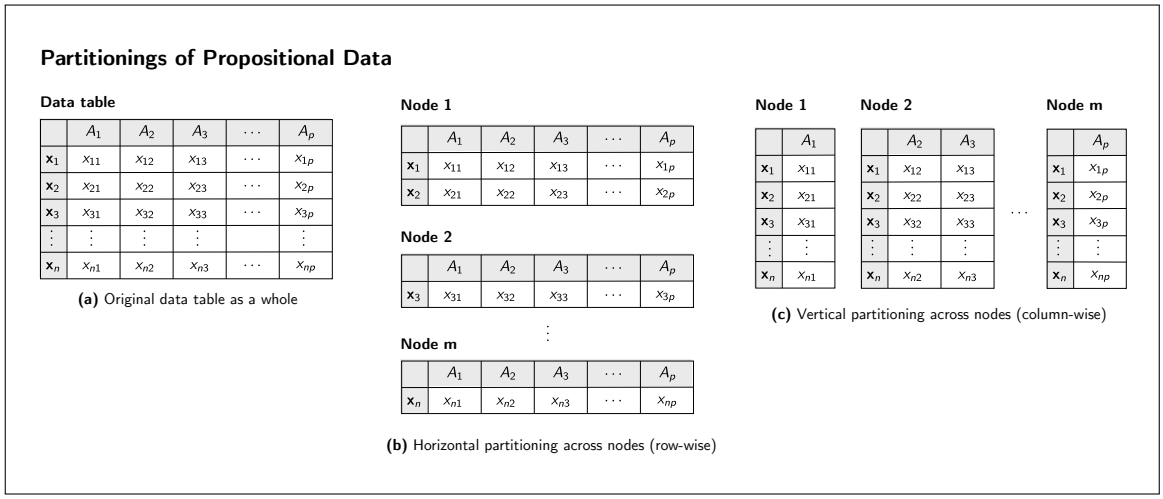
Technology	Rate		Type
EDGE	237.0	kB/s	Mobile Phone
UMTS 3G	48.0	kB/s	Mobile Phone
LTE	40.75	MB/s	Mobile Phone
802.15.4 (2.4 GHz)	31.25	kB/s	Wireless
Bluetooth 4.0	3.0	MB/s	Wireless
IEEE 802.11n	75.0	MB/s	Wireless
IEEE 802.11ad	900.0	MB/s	Wireless
Solid-state drive (SSD)	600.0	MB/s	Storage
eSATA	750.0	MB/s	Peripheral
USB 3.0	625.0	MB/s	Peripheral
VDSL2	12.5	MB/s	Broadband
Ethernet	1.25	MB/s	Local Area
Gigabit Ethernet	125.0	MB/s	Local Area
100 Gigabit Ethernet	12.5	GB/s	Local Area
Infiniband EDR 12x	37.5	GB/s	Local Area
PC4-25600 DDR4 SDRAM	25.6	GB/s	Memory

5.2 Communication-constrained Scenarios

A central analysis of IoT generated data requires its transmission over a network. However, due to technical limitations, the transmission of *all* data to a central location, like a data center, is not always possible. Either the data generating devices themselves are highly communication-constrained, or the available bandwidth is too limited. Moreover, there exist cases where privacy concerns, security concerns, business competition or political regulations prohibit the centralization of all data.

Communication-constrained devices. One of mobile devices' biggest constraint is that they are battery powered. Devices having much less computational power, like embedded devices or smart sensors, can be battery powered as well, even if they aren't mobile. Sending and receiving data is known to be one of the most energy draining operations on mobile devices [22] and smart sensors [63]. Hence, communication must be traded off against computation.

Limitations of bandwidth. There exist several scenarios in which the available bandwidth does not suffice to transmit all data to a central location. IoT generated data may stem from devices that are connected wirelessly. Table 1 shows typical transfer rates for different kinds of network technologies and bus systems. It becomes apparent that wireless networks provide much lower bandwidths than LANs which are used in data centers. For instance, ZigBee networks based on IEEE 802.15.4, a specification for personal area networks consisting of small, low-power digital radios, have a data transmission rate of only 31.25 kB/s. Mobile devices, like smartphones or tablets, are relatively powerful in



(a) Original data table as a whole

(b) Horizontal partitioning across nodes (row-wise)

(c) Vertical partitioning across nodes (column-wise)

Figure 5: Common types of data partitioning

terms of computation and available main memory (see also Fig. 4). They easily may generate data at higher rates than can be transmitted over mobile telephone interfaces. Other applications, like those in earth science [112] or telescopes in physics [11], produce masses of data whose transmission over satellite connections is in the range of years. Masses of data are also generated by high throughput applications, like Formula One racing [89], which require a real-time analysis of large amounts of data [26]. Similarly, analysis and control in manufacturing can have real-time constraints [91,94]. In cases where reaction times lie in the range of a few seconds, it seems risky to send production parameters first into the cloud for preprocessing and analysis, which then computes an answer. Depending on latency, which can be high with Internet based services, the answer may come too late. Finally, bandwidth becomes more limited with more network participants. With the IoT, those will likely increase as more and more devices are getting connected to the same network segments [76]. According to [70], "how to control the huge amount of data injected into the network from the environment is a problem so far mostly neglected in the IoT research".

Privacy concerns and regulations. Privacy concerns and regulations may entirely prohibit the transmission of data to a central location. Or, privacy-preserving algorithms may transmit data, but not the original records. Further, network usage might be constrained by political or business regulations, such that data cannot be centralized. Other issues concern security and fail-safe operation. Centralized systems pose single points of failure. The more control is depending on data and its analysis, the more important it is to guarantee its delivery. In the cloud computing scenario, service provider and client may secure their end points, but usually have no control over the transmission of packets in between. A smart factory sending all its data into the cloud, depending on a timely analysis for real-time operation, might come to a complete standstill in case of a network failure. Even if the cloud is not available, continuous local operation should at least be possible.

In all of the aforementioned cases, data must be directly analysed on the generating devices themselves and be reduced before transmission (see also [8, 26, 40, 80]). For instance, as shown in [63], the reduction of data before transmission with the help of autogressive models reduced the energy consumption of smart sensors (MEMS) by factors up to 11. Similar reductions could be achieved with edge mining [40], whose authors argue purely in favor of local data preprocessing. However, local transformations and models may not suffice to capture dependencies between highly correlated measurements from different sensors. In such cases, decentralized algorithms are needed which build a global model based on messages exchanged between peer nodes or with a coordinator node. Such algorithms will necessarily need to be designed differently from distributed algorithms running in a data center. There, network technology allows for transfer rates resembling those of main memory accesses. Moreover, it may be freely decided how data is getting stored and partitioned across machines. New storage and compute nodes may be dynamically added to the network, based on demand. However, on the data generating side, the kind of data partitioning as well as the network structure are usually application dependent and given as fixed. Especially the type of data partitioning can have a large influence on learning and the amount of data that needs to be communicated, as shown in the following section.

6. TYPES OF DATA PARTITIONING

Data for learning is often given as a sample S of n observations, i.e. $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. For the following discussion, w.l.o.g. it is assumed that observations are represented in *propositional* form, i.e. described by a finite set of p different *features* $A_1, \ldots, A_p$ (also called *attributes*). Feature values are stored in columns of a data table, with one observation per row (see Fig. 5a). In distributed settings, data from this table may be spread across nodes in two different ways [21].

Horizontal partitioning. In the *horizontally partitioned data* scenario (see Fig. 5b), data about observation, i.e. rows of the data table, are distributed across nodes $j = 1, \ldots, m$.

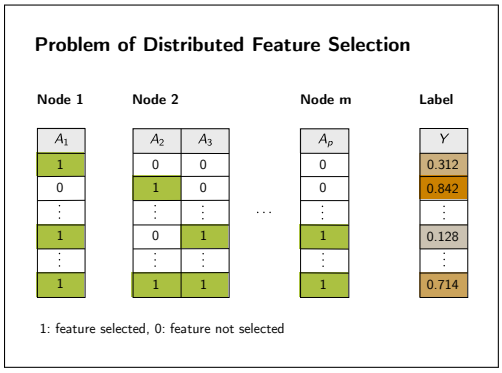


Figure 6: Which features provide most information about the target concept?

All observations share the *same features*.

Horizontally partitioned sets of observations may be seen as skewed subsamples of a dataset that would result from centralizing and merging all observations. Hence, the distributed learning task consists of building a global model from such local samples, with as few communication between nodes as possible. Observations may be assumed to be independent and identically distributed, which for instance is exploited by learning algorithms that merge summary information independently derived from each subsample. In general, there exist many distributed learning algorithms for the scenario (for instance [14, 27, 52, 65]), though only few algorithms are truly suited for small devices (for a more detailed treatment, see [6,90]). Communication costs for the scenario are well understood in the sense that bounds have been established for different classes of learning problems [4, 113]. For instance, [4] show that a distributed perceptron, which is a linear classifier, can find a consistent hypothesis in at most $O(k(1+\alpha/\gamma^2))$ rounds of communication, k being the number of nodes, supposed that data is α-well-spread and all points have margin at least γ with the separating hyperplane.

An example task for learning in the horizontally partitioned data scenario is link quality prediction in wireless sensor networks (WSNs). We may assume that factors influencing link quality are the same across different wireless sensor nodes, i.e. recorded features provide information about the same underlying concept to be learned. However, the distributions of observations may differ for different parts of the network. For instance, in certain parts the link quality could be better than in other parts. The question is how to learn a global model which represents the distribution over all observations across nodes, without having to transfer all observations to a central node.

Vertical partitioning. In the *vertically partitioned data* scenario (see Fig. 5c), feature values of observations, i.e. columns of the data table, are distributed across nodes $j = 1, \ldots, m$. Shared is only the index column, such that it is known which features belong to which observation. This might require a continuous tracking of objects, which in the IoT would be realized through globally unique identifiers for each entity.

The columns distributed over nodes constitute subspaces of the whole instance space. These subspaces and their individual components (e.g. features), in supervised learning including the target label, have a dependency structure that is usually unknown before learning. Learning in the scenario may thus be seen as a combinatorial problem of exponential size: Which subset of features provides the most information about the target concept (see also Fig. 6)? In supervised learning, this is also known as the *feature selection* [87] problem, whereas in unsupervised learning similar problems occur in *subspace clustering* [59]. Several techniques have been developed to tackle the exponential search space [56]. Most of them are highly iterative and assume that features can be freely combined with each other. In a decentralized setting, however, such combination requires the costly transmission of column information between nodes in each iteration step and is thus prohibited. Hence, current approaches [28,61,92] circumvent such problems by making explicit assumptions on the conditional joint dependencies of features, given the label.

In the context of the IoT, learning in the vertically partitioned data scenario is relevant and common. The problem occurs whenever a state or event is to be detected or predicted, based on feature values assessed at different nodes. What exactly constitutes a single observation then is application dependent. A common use case are spatio-temporal prediction models, which use measurements of devices at different locations. Measurements may be related to each other by the time interval in which they occur. The following list gives examples of applications:

- In manufacturing, one is interested in predicting the final product quality as early as possible [91,94], based on process parameters and measurements at different production steps. Similarly, the optimization of process flow could benefit from a prediction of the time it takes to assemble a product, based on the current filling of queues and machine parameters at different locations on a shop floor. In both cases, a single observation consists of features, like sensor measurements and machine parameters, that are distributed and assessed at different locations. Depending on the granularity of control to be achieved, predictions must be either given after minutes, seconds or maybe also milliseconds. The more time-constrained the application, the more it might benefit from decentralized local processing.
- Products are assembled from parts delivered by different suppliers [105]. Optimal planning and scheduling of assembly steps depend on a correct and continuous estimation of parts' delivery times. Those again are determined by production and transportation parameters of individual suppliers. For instance, the delivery of a particular part might be delayed due to the maintenance of a single production unit at one supplier. Assembly time of a product is thus a global function depending on local information (features) from different suppliers, i.e. observations for learning this function are vertically partitioned. Even if it was technically feasable to centralize the raw production and transportation data from all suppliers for analysis, it would be unnecessary if the global function depended only on a few local features. Moreover, due to privacy concerns, it is unrealistic that suppliers would provide raw data about their processes. Hence, a decentralized

algorithm is needed that derives a global model from local data, at the same time preserving privacy.

- The smart grid requires a continuous prediction of energy demand [55,65], based on local information about energy usage at different smart homes [116]. Here, observations might represent the whole state of the energy grid, and consist of vertically partitioned features at different locations describing local states. Instead of centralizing raw meter readings from ten thousands of households, communication could be spared by an aggregation of local data or a combination of predictions from locally trained models.

- Centralized traffic management systems analyse traffic based on data from a hard-wired mesh of distributed presence sensors [83]. While easy to design, centralized systems pose a single point of failure in case of an emergency. With the addition of new sensors, they may become a bottleneck, due to limited bandwidth. Further, the maintenance of hard-wired sensors can be expensive in case of failure, due to required construction work. A more decentralized system could consist of cheap wireless sensors. Those may be attached to existing infrastructure, like traffic lights, signs and street lights. Traffic lights may then adjust themselves, based on the prediction of traffic flow at neighboring junctions. The flow measurements at each individual junction can be interpreted as vertically partitioned features of a single observation describing the current state of all sensors. The learning task is to derive prediction models from these distributed flow measurements, without transmission of all data to a central server [92].

- In healthcare, diagnoses of illnesses depend on many factors, like a patient's health care records, parents' illnesses and current health parameters such as puls, blood pressure, measurements from a blood sample, an electroencephalogram or other specialized information. With IoT technology, even more data becomes available through fitness trackers or dieting apps (see also Sect. 3.5). The features describing a single patient are thus distributed over different locations, like several physicians, medical centers, and now even devices or social websites. The centralization of all data poses a threat to patients' privacy. Hence, the learning task is to derive a global model for diagnosis from local data, without transmission of raw data between locations. The features of diagnoses from different geographical locations over certain time intervals could then be combined to predict, for instance, epidemics and their spread at a larger scale (see also [73]). Again, the features from different locations over the same time intervals constitute vertically partitioned observations.

7. RESEARCH QUESTIONS

The number of communication-efficient distributed data analysis methods for the vertically partitioned is much smaller than those for horizontally partitioned data. There are many open research questions, which mainly concern the relationship between accuracy and communication costs. Therefore, we first define how communication costs are measured and what it means for an algorithm to be communication-efficient. Then, an overview of typical components that vertically distributed algorithms may consist of is given. It is shown that the schema is general enough to cover common designs of distributed algorithms. Finally, open issues and research questions are formulated that concern communication-efficient learning.

7.1 Communication Costs and Efficiency

In most publications on distributed data analysis, *communication costs* are the total payload transmitted measured in bits, i.e. excluding meta data, like packet headers. The authors of [40] argue for a measurement of communication costs by the number of transmitted packets. Although the number of packets in certain cases might be a more exact measure than the payload in bits, it is highly dependent on chosen network protocols and the underlying network technology. Similar to measuring the run-time of algorithms in seconds, it would make the comparison of results from different publications very difficult. A fair comparison would require building the exact same network with the same hardware and configuration. A solution could be network simulators, however, there doesn't seem to exist a commonly agreed on standard between different scientifc communities. At least for batch transmissions of data, the number of packets to be sent is proportional to the payload in bits. From there, we follow the argumentation in [40] that a reduction of packets may reduce congestion and collisions on networks with large amounts of traffic. This in turn reduces the number of acknowledgements and retransmissions, which should enable better use of available bandwidth (i.e. higher transmission rates or more network participants).

Central analysis requires the transmission of all data (or at least all preprocessed data) to the coordinator node. We define a learning method to be *communication-efficient* if less data than the whole dataset (optionally after local preprocessing) is exchanged between local nodes and an optional coordinator node. Method A is called *more communication-efficient* than method B, if A is communication-efficient and its communication costs are less than those of B.

The amount of data communicated per observation during learning may differ from the amount communicated when making an actual prediction. It should be noted that in the vertically partitioned data scenario, at least *some* data must be communicated for detecting a global state or predicting a global event. Further, the supervised learning of local models may require the transmission of label information from a coordinator. This is different from a horizontal partitioning of data, where each local node contains all the necessary information (i.e. feature values and often also the label).

7.2 Distributed Setting and Components

Figure 7 gives an overview of the setting in the vertically partitioned data scenario and the distributed components that algorithms may be designed of. Given are $m+1$ networked nodes $j = 0, \dots m$, where nodes $1, \dots, m$ are called *local nodes* and $j = 0$ denotes a *coordinator node*. No assumptions are made on network topology or technology. Further, "local" and "coordinator" are to be understood as *roles* that physical nodes can have, and may change depending on context.

Each local node acquires raw values, like sensor measurements. Those may be locally preprocessed and transformed

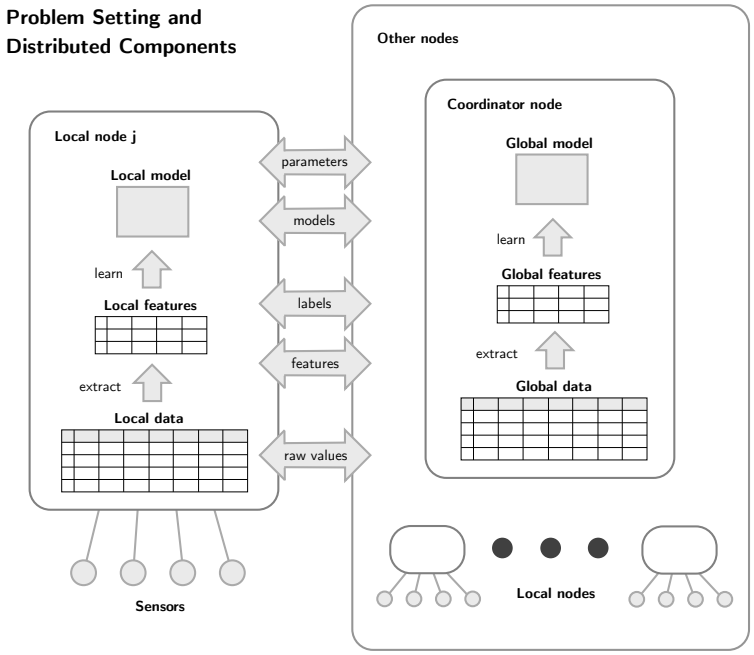


Figure 7: Problem setting and distributed components

into features for learning. It is assumed that the features of the same observations are vertically partitioned across the local nodes. Distributed components of learning algorithms may do further local calculations on such features, and might build or update local models. Once or iteratively, depending on algorithm, local nodes will either transmit raw values, features, models or predictions of such models to other nodes, which in turn may preprocess the received data, and do further calculations on them, like build or update a global model, fuse predictions, etc. The setting as described is general enough to cover the following common approaches for designing distributed algorithms:

Central analysis Each local node transmits all of its raw values to a coordinator node for further analysis. This may include the stages of preprocessing, feature extraction and model building. This is in principle what cloud-based data processing proposes [19,23,31,43,76]: While the coordinator may consist itself of distributed components and solve the analyis problem in parallel, from the perspective of local nodes it looks like a single machine where all data is getting centralized. The design is not decentralized, as data and processing aren't split between local nodes and coordinator node, but all processing is done at the coordinator node.

Local preprocessing, central analysis Local nodes preprocess raw values and transform them into a representation for learning. The representations are sent to a coordinator, which builds a global model based on them. While according to our former definition, this design is decentralized, its form is very rudimentary, as most of the processing is still done at the coordinator. Depending on the processing capabilities of local nodes and the particular learning task, such a design might be the only viable option. The design fits ideas mentioned in [40, 80, 101], whose authors' propose to reduce data locally before sending it to the cloud for analysis. Privacy-preserving Support Vector Machine (SVM) algorithms like [66, 111] also follow this design, but are not necessarily communication-efficient.

Model consensus Local nodes iteratively try to reach consensus on a set of parameters among each other (peer-to-peer), or on a set of parameters they share with a coordinator node. At the end, each local node (or only the coordinator) has a global model. As [13] demonstrate, many existing analysis problems can be cast into a consensus problem and then be solved, for instance, with the Alternating Direction Method of Multipliers (ADMM). Algorithms of this sort are working fully decentralized, but are working iteratively and may transmit more than the original data, depending on their convergence properties.

Fusion of local models Each local node preprocesses its own data and builds a local model on it. Such models are then transmitted to a coordinator node or to peer nodes, which fuse them to a global model. These algorithms are working decentralized, as data and the load of processing are shared among all nodes. In the vertically partitioned data scenario, using a global model usually requires the transmission of feature values of observations whenever a prediction is to be made. An example algorithm would be [48].

Fusion of local predictions Each local node preprocesses its own data and builds a local model on it. Whenever a prediction is to be made, only the predictions

are transmitted from local nodes, and fused at the coordinator or other peer nodes according to a fusion rule. This could be, for instance, a majority vote over predictions. Local nodes each transmit only one value during prediction, but a fusion rule may not be as accurate as a global model, depending on data distribution and learning task. Examples would be [61, 92].

While aforementioned approaches are common, there exist hybrids also covered by the setting shown in Fig. 7. For instance, in [28] local models are used to detect local outliers, which are then checked against a global model that was derived from a small sample of all data.

According to our previous definition, the examples of distributed algorithms given above all learn from vertically partitioned data in a decentralized fashion. However, not all are communication-efficient. Apart from the two mentioned privacy-preserving SVMs which might send more data than the whole dataset, the model consensus based algorithms may send more data as well, depending on the number of iterations during optimization. The design of communication-efficient decentralized algorithms in the vertically partitioned data scenario leaves many open research questions of which some are presented in the following.

7.3 Open Questions

Despite first successes in the development of communication-efficient algorithms for the vertically partitioned data scenario, there are still many open research questions left:

- Data analysis knows many different kinds of tasks, like dimensionality reduction, classification, regression, clustering, outlier detection or frequent itemset mining. How does the task influence communication costs when the data is vertically partitioned? And how does the design change with the task?
- As first results suggest, the accuracy of communication-efficient algorithms in the vertically partitioned data scenario very much depends on the data being analysed. What influence have different data distributions on the communication costs and accuracy of algorithms? How is the design of algorithms affected?
- What are bounds on communication, i.e. how much information must be at least and at most communicated to learn successfully from vertically partitioned data?
- How can the supervised learning of local models be made more communication-efficient in cases where labels do not reside on the local nodes, but must first be transmitted to them? For instance, how can we learn from aggregated label information?
- Many existing data analysis algorithms can easily work on different numbers of observations, but expect the number of features to be fixed. How can algorithms that work on observations with features from different sensors deal with the dynamic addition and removal of sensors, i.e. features?

Beyond those questions, there are open issues concerning distributed data analysis algorithms in general, i.e. also those that work on horizontally partitioned data or in the cloud. For instance, methods for feature selection, the optimization of hyper parameters and validation are highly iterative and work on different subsets of features and observations in each iteration. How can we adapt these algorithms in such a way that the same data isn't repeatedly sent over the network or read from external storage? As the previous questions demonstrate, there is still a lot of research to do before data analysis and the IoT will become seamlessly integrated.

8. SUMMARY

After a short introduction to the IoT, it was argued for data analysis being an essential part of it. By giving examples from different sectors, it was shown that already remote monitoring applications may benefit from a summarization of data with the help of data analysis. Complex applications require more advanced and autonomous control mechanisms. These in turn depend on advanced data analysis methods, like those that can analyse data in real-time, adapt to changing concepts and representations and test hypotheses actively. Beyond security, privacy and technical problems, especially algorithmic challenges need to be tackled before such advanced applications will become a reality.

Distributed cloud-based algorithms follow the paradigm of parallel high performance computing. The cloud might seem like the most convenient and powerful solution for the analysis of IoT generated big data, which is expected to have large volume, high velocity and high heterogenity. However, without substantial advances in network technology, bandwidth will become more and more scarce with each new device getting connected. The transmission of all data into the cloud can already be infeasable, due to limited energy, bandwidth, high latency or due to privacy concerns and regulations. Communication-constrained applications require decentralized analysis algorithms which at least partly work directly on the devices generating the data, like sensors and embedded devices. A particularly challenging scenario is that of vertically partitioned data, which covers common IoT use cases, but for which not many data analysis algorithms exist so far. The main research question is how to design communication-efficient decentralized algorithms for the scenario, while at the same time preserving the accuracy of their centralized counterparts.

Several works achieved impressive resource savings by reducing data with the help of analysis directly on embedded devices and sensors. In the field of data analysis, research on communication-efficient decentralized algorithms is active, as several given citations demonstrate. It seems surprising that many other surveys focus mostly on cloud-based analysis solutions, ignoring the up-coming challenges of communication-constrained IoT applications. We hope to have closed this gap by our work and providing a comprehensive bibliography. We think that in the future IoT, cloud-based and decentralized data analyis solutions will coexist and complement each other.

9. ACKNOWLEDGEMENTS

This work has been supported by the DFG, Collaborative Research Center SFB 876 (http://sfb876.tu-dortmund.de/), project B3.

10. REFERENCES

[1] C. Aggarwal, N. Ashish, and A. Sheth. The Internet of Things: A Survey From The Data-Centric Perspective. In C. C. Aggarwal, editor, *Managing and Mining Sensor Data*. Springer, Berlin, Heidelberg, 2013.

[2] Argonne National Laboratory. The Message Passing Interface (MPI) standard. `http://www.mcs.anl.gov/research/projects/mpi/`, 2015. [Online; accessed 2015-12-15].

[3] L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A survey. *Comput. Netw.*, 54(15):2787–2805, 2010.

[4] M.-F. Balcan, A. Blum, S. Fine, and Y. Mansour. Distributed Learning, Communication Complexity and Privacy. In *JMLR: Workshop and Conference Proceedings, 25th Annual Conference on Learning Theory*, 2012.

[5] W. Bernhart and M. Winterhoff. Autonomous Driving: Disruptive Innovation that Promises to Change the Automotive Industry as We Know It. In J. Langheim, editor, *Energy Consumption and Autonomous Driving: Proc. of the 3rd CESA Automotive Electronics Congress*. Springer, 2016.

[6] K. Bhaduri and M. Stolpe. Distributed Data Mining in Sensor Networks. In C. Aggarwal, editor, *Managing and Mining Sensor Data*. Springer, Berlin, Heidelberg, 2013.

[7] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer. MOA: Massive Online Analysis. *J. Mach. Learn. Res.*, 11:1601–1604, Aug. 2010.

[8] S. Bin, L. Yuan, and W. Xiaoyi. Research on Data Mining Models for the Internet of Things. In *Proc. of the Int. Conf. on Image Analysis and Signal Processing (IASP)*, pages 127–132, 2010.

[9] C. Bockermann. *Mining Big Data Streams for Multiple Concepts*. PhD thesis, TU Dortmund, Dortmund, Germany, 2015.

[10] C. Bockermann, M. Apel, and M. Meier. Learning SQL for Database Intrusion Detection Using Context-Sensitive Modelling. In U. Flegel, , and D. Bruschi, editors, *Proc. of the 6th Int. Conf. on Detection of Intrusions and Malware (DIMVA)*, pages 196–205. Springer, Berlin, Heidelberg, 2009.

[11] C. Bockermann, K. Brügge, J. Buss, A. Egorov, K. Morik, W. Rhode, and T. Ruhe. Online Analysis of High-Volume Data Streams in Astroparticle Physics. In *Proc. of the European Conf. on Machine Learning (ECML), Industrial Track*. Springer, 2015.

[12] A. Botta, W. de Donato, V. Persico, and A. Pescapé. Integration of Cloud computing and Internet of Things: A survey. *Future Gener. Comp. Sy.*, 56:684–700, 2016.

[13] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, Jan. 2011.

[14] J. Branch, C. Giannella, B. Szymanski, R. Wolff, and H. Kargupta. In-network outlier detection in wireless sensor networks. *Knowl. Inf. Sys.*, 34(1):23–54, 2012.

[15] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[16] M. Brettel, N. Friederichsen, M. Keller, and M. Rosenberg. How Virtualization, Decentralization and Network Building Change the Manufacturing Landscape: An Industry 4.0 Perspective. *Int. Journ. of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 8(1):37–44, 2014.

[17] A. Buczak and E. Guven. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys & Tutorials*, 2015.

[18] N. Bui and M. Zorzi. Health Care Applications: A Solution Based on the Internet of Things. In *Proc. of the 4th Int. Symp. on Applied Sciences in Biomedical and Communication Technologies*, ISABEL '11, pages 131:1–131:5. ACM, 2011.

[19] D. Burrus. The Internet of Things is Far Bigger Than Anyone Realizes. `http://www.wired.com/insights/2014/11/the-internet-of-things-bigger/`, 2014. [Online; accessed 2016-02-16].

[20] Canalys. Wearable band shipments set to exceed 43.2 million units in 2015. `http://www.canalys.com/newsroom/wearable-band-shipments-set-exceed-432-million-units-2015`, 2014. [Online; accessed 2016-04-04].

[21] D. Caragea, A. Silvescu, and V. Honavar. Agents that learn from distributed dynamic data sources. In *Proc. of the Workshop on Learning Agents*, 2000.

[22] A. Carroll and G. Heiser. An Analysis of Power Consumption in a Smartphone. In *Proc. of the 2010 USENIX Conf. on USENIX Ann. Technical Conf. (USENIXATC)*, USA, 2010. USENIX Association.

[23] F. Chen, P. Deng, J. Wan, D. Zhang, A. Vasilakos, and X. Rong. Data Mining for the Internet of Things: Literature Review and Challenges. *Int. J. Distrib. Sen. Netw.*, 2015:12:12–12:12, Jan. 2015.

[24] M. Chen, Y. Ma, J. Wang, D. Mau, and E. Song. Enabling Comfortable Sports Therapy for Patient: A Novel Lightweight Durable and Portable ECG Monitoring System. In *IEEE 15th Int. Conf. on e-Health Networking, Applications and Services (Healthcom)*, pages 271–273, 2013.

[25] M. Chui, M. Löffler, and R. Roberts. The Internet of Things. `http://www.mckinsey.com/insights/high_tech_telecoms_internet/the_internet_of_things`, Mar. 2010. [Online; accessed 2016-02-16].

[26] F. Combaneyre. Understanding Data Streams in IoT. http://www.sas.com/en_us/whitepapers/understanding-data-streams-in-iot-107491.html, 2015. [Online; accessed 2016-02-23].

[27] K. Das, K. Bhaduri, and H. Kargupta. A local asynchronous distributed privacy preserving feature selection algorithm for large peer-to-peer networks. *Knowledge and Information Systems*, 24(3):341–367, 2009.

[28] K. Das, K. Bhaduri, and P. Votava. Distributed Anomaly Detection Using 1-class SVM for Vertically Partitioned Data. *Stat. Anal. Data Min.*, 4(4):393–406, Aug. 2011.

[29] E. Davies. *Computer and Machine Vision: Theory, Algorithms, Practicalities*. Academic Pr, Inc., 2012.

[30] G. De Francisci Morales and A. Bifet. SAMOA: Scalable Advanced Massive Online Analysis. *J. Mach. Learn. Res.*, 16(1):149–153, Jan. 2015.

[31] M. Díaz, C. Martín, and B. Rubio. State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing. *Journal of Network and Computer Applications*, 2016.

[32] J. Dixon. Who Will Step Up To Secure The Internet of Things? http://techcrunch.com/2015/10/02/who-will-step-up-to-secure-the-internet-of-things/, 2015. [Online; accessed 2016-02-16].

[33] P. Engebretson. *The Basics of Hacking and Penetration Testing*. Elsevier/Syngress, 2nd edition, 2013.

[34] D. Evans. The Internet of Things – How the Next Evolution of the Internet Is Changing Everything. https://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf, Apr. 2011. [Online; accessed 2015-11-19].

[35] P. Evans and M. Annunziata. Industrial Internet: Pushing the Boundaries of Minds and Machines. http://www.ge.com/docs/chapters/Industrial_Internet.pdf, 2012. [Online; accessed 2016-04-04].

[36] T. Fawcett. Mining the Quantified Self: Personal Knowledge Discovery as a Challenge for Data Science. *Big Data*, 3(4):249–266, Jan. 2016.

[37] D. Fletcher. Internet of Things. In M. Blowers, editor, *Evolution of Cyber Technologies and Operations to 2035*, pages 19–32. Springer International Publishing, 2015.

[38] J. Gama. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition, 2010.

[39] H. Garcia-Molina, J. Ullman, and J. Widom. *Database Systems: The Complete Book*. Pearson Education Limited, 2nd edition, 2013.

[40] E. Gaura, J. Brusey, M. Allen, R. Wilkins, D. Goldsmith, and R. Rednic. Edge Mining the Internet of Things. *IEEE Sensors Journal*, 13(10):3816–3825, 2013.

[41] F. Gianotti and D. Pedreschi, editors. *Mobility, Data Mining and Privacy*. Springer, 2007.

[42] J. Glaser. How The Internet of Things Will Affect Health Care. http://www.hhnmag.com/articles/3438-how-the-internet-of-things-will-affect-health-care, Jun. 2015. [Online; accessed 2016-02-23].

[43] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Gener. Comput. Syst.*, 29(7):1645–1660, Sep. 2013.

[44] J. Han and M. Kamber. *Data Mining*. Morgan Kaufmann, 2nd edition, 2006.

[45] B. Harpham. How the Internet of Things is changing healthcare and transportation. http://www.cio.com/article/2981481/healthcare/how-the-internet-of-things-is-changing-healthcare-and-transportation.html, Sep. 2015. [Online; accessed 2016-02-16].

[46] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd edition, 2009.

[47] A.-C. Hauschild, T. Schneider, J. Pauling, K. Rupp, M. Jang, J. Baumbach, and J. Baumbach. Computational Methods for Metabolomic Data Analysis of Ion Mobility Spectrometry Data - Rviewing the State of the Art. *Metabolites*, 2(4):733–755, 2012.

[48] C. Heinze, B. McWilliams, and N. Meinshausen. DUAL-LOCO: Preserving privacy between features in distributed estimation. In *Proc. of the 19th Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, JMLR: Workshop and Conference Proceedings, 2016.

[49] P. Hintjens. *ZeroMQ*. O'Reilly, USA, 2013.

[50] IBM. IBM Intelligent Water: Water management software with analytics for improved infrastructure and operations. http://www-03.ibm.com/software/products/en/intelligentwater, 2016. [Online; accessed 2016-04-01].

[51] M. Imhoff, R. Fried, U. Gather, and V. Lanius. Dimension Reduction for Physiological Variables Using Graphical Modeling. In *AMIA 2003, American Medical Informatics Association Annual Symposium, Washington, DC, USA, November 8-12, 2003*, 2003.

[52] M. Kamp, M. Boley, D. Keren, A. Schuster, and I. Scharfman. Communication-Efficient Distributed Online Prediction by Decentralized Variance Monitoring. In T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, editors, *Proc. of the European Conf. on Machine Learning and Principles and Practice of Knowledge Discovery (ECML/PKDD)*, pages 623–639. Springer, 2014.

[53] H. Kargupta, R. Bhargava, K. Liu, M. Powers, P. Blair, S. Bushra, J. Dull, K. Sarkar, M. Klein, M. Vasa, and D. Handy. VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring. In *Proc. of the SIAM Int. Conf. on Data Mining (SDM)*, chapter 28, pages 300–311. 2004.

[54] H. Kargupta, K. Sarkar, and M. Gilligan. MineFleet: an overview of a widely adopted distributed vehicle performance data mining system. In *Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 37–46, 2010.

[55] A. Khan, A. Mahmood, A. Safdar, Z. Khan, and N. Khan. Load forecasting, dynamic pricing and DSM in smart grid: A review. *Renew. Sust. Energ. Rev.*, 54:1311–1322, 2016.

[56] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[57] R. Krawiec, J. Nadler, P. Kinchley, E. Tye, and J. Jarboe. No appointment necessary: How the IoT and patient-generated data can unlock health care value. `http://dupress.com/articles/internet-of-things-iot-in-health-care-industry/`, Aug. 2015. [Online; accessed 2016-02-16].

[58] J. Kreps. Questioning the Lambda Architecture. `http://radar.oreilly.com/2014/07/questioning-the-lambda-architecture.html`, 2014. [Online; accessed 2015-12-15].

[59] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering High-dimensional Data: A Survey on Subspace Clustering, Pattern-based Clustering, and Correlation Clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1:1–1:58, Mar. 2009.

[60] V. Lanius and U. Gather. Robust online signal extraction from multivariate time series. *Comput. Stat. Data An.*, 54(4):966–975, 2010.

[61] S. Lee, M. Stolpe, and K. Morik. Separable Approximate Optimization of Support Vector Machines for Distributed Sensing. In P. Flach, T. D. Bie, and N. Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *LNCS*, pages 387–402, Berlin, Heidelberg, 2012. Springer.

[62] T. Liebig, N. Piatkowski, C. Bockermann, and K. Morik. Predictive Trip Planning - Smart Routing in Smart Cities. In *Proc. of the Workshop on Mining Urban Data at the Int. Conf. on Extending Database Technology*, pages 331–338, 2014.

[63] J. Long, J. Swidrak, M. Feng, and O. Buyukozturk. Smart Sensors: A Study of Power Consumption and Reliability. In E. Wee Sit, editor, *Sensors and Instrumentation, Volume 5: Proc. of the 33rd IMAC, A Conf. and Exposition on Structural Dynamics*, pages 53–60. Springer, 2015.

[64] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297, 1967.

[65] R. Mallik and H. Kargupta. A Sustainable Approach for Demand Prediction in Smart Grids using a Distributed Local Asynchronous Algorithm. In *Proc. of the Conf. on Data Understanding (CIDU)*, pages 1–15, 2011.

[66] O. Mangasarian, E. Wild, and G. Fung. Privacy-preserving Classification of Vertically Partitioned Data via Random Kernels. *ACM Trans. Knowl. Discov. Data*, 2(3):12:1–12:16, Oct. 2008.

[67] N. Marz and J. Warren. *Big Data - Principles and best practices of scalable realtime data systems*. Manning, 2014.

[68] F. Mattern and C. Floerkemeier. From the Internet of Computers to the Internet of Things. In K. Sachs, I. Petrov, and P. Guerrero, editors, *From Active Data Management to Event-based Systems and More*, pages 242–259. Springer-Verlag, Berlin, Heidelberg, 2010.

[69] M. May, B. Berendt, A. Cornuejols, J. Gama, F. Giannotti, A. Hotho, D. Malerba, E. Menesalvas, K. Morik, R. Pedersen, L. Saitta, Y. Saygin, A. Schuster, and K. Vanhoof. Research Challenges in Ubiquitous Knowledge Discovery. In Kargupta, Han, Yu, Motwani, and Kumar, editors, *Next Generation of Data Mining (NGDM)*, pages 131–151. CRC Press, 2009.

[70] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7):1497–1516, 2012.

[71] T. Mitchell. *Machine Learning*. Mcgraw-Hill Education Ltd, 1997.

[72] K. Morik, K. Bhaduri, and H. Kargupta. Introduction to data mining for sustainability. *Data Min. Knowl. Disc.*, 24(2):311–324, Mar. 2012.

[73] R. Moss, A. Zarebski, P. Dawson, and J. McCaw. Forecasting influenza outbreak dynamics in melbourne from internet search query surveillance data. *Influenza and Other Respiratory Viruses*, page n/a, Feb. 2016.

[74] Navigant Research. Shipments of Smart Thermostats Are Expected to Reach Nearly 20 Million by 2023. `https://www.navigantresearch.com/newsroom/shipments-of-smart-thermostats-are-expected-to-reach-nearly-20-million-by-2023`, 2014. [Online; accessed 2016-04-04].

[75] Oracle Corporation. Energize Your Business with IoT-Enabled Applications. `http://www.oracle.com/us/dm/oracle-iot-cloud-service-2625351.pdf`, 2015. [Online; accessed 2016-02-16].

[76] Oracle Corporation. Unlocking the Promise of a Connected World: Using the Cloud to Enable the Internet of Things. `http://www.oracle.com/us/solutions/internetofthings/iot-and-cloud-wp-2686546.pdf`, 2015. [Online; accessed 2015-12-15].

[77] Oxford Economics. Manufacturing Transformation: Achieving competetive advantage in changing global marketplace. `http://www.oxfordeconomics.com/Media/Default/Thought%20Leadership/executive-interviews-and-case-studies/PTC/Manufacturing%20Transformation%20130607.pdf`, 2013. [Online; accessed 2016-04-04].

[78] D. Partynski and S. Koo. Integration of Smart Sensor Networks into Internet of Things: Challenges and Applications. In *Proc. of the IEEE Int. Conf. on Green Computing and Communications (GreenCom) and IEEE Internet of Things (iThings) and IEEE Cyber, Physical and Social Computing (CPSCom)*, pages 1162–1167, 2013.

[79] G. Patrini, R. Nock, T. Caetano, and P. Rivera. (Almost) No Label No Cry. In *Advances in Neural Information Processing Systems (NIPS)*, number 27, pages 190–198. Curran Associates, Inc., 2014.

[80] Y. Qin, Q. Sheng, N. Falkner, S. Dustdar, H. Wang, and A. Vasilakos. When things matter: A survey on data-centric internet of things. *J. Netw. Comput. Appl.*, 64:137–153, 2016.

[81] R. Roman, J. Zhou, and J. Lopez. On the features and challenges of security and privacy in distributed internet of things. *Computer Networks*, 57(10):2266–2279, 2013.

[82] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2013.

[83] SCATS. Sydney Coordinated Adaptive Traffic System. `http://www.scats.com.au/`, 2013. [Online; accessed 2015-08-19].

[84] D. Shoup. Free Parking or Free Markets. *Cato Unbound - A Journal of Debate*, 2011. [Online; accessed 2016-04-04].

[85] K. Shvachko, H. K., S. Radia, and R. Chansler. The Hadoop Distributed File System. In *IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pages 1–10, 2010.

[86] SmartSantanderSantander. Future Internet Research & Experimentation. `http://www.smartsantander.eu`, 2016. [Online; accessed 2016-04-01].

[87] U. Stanczyk and L. Jain, editors. *Feature Selection for Data and Pattern Recognition*. Studies in Computational Intelligence. Springer, 2015.

[88] W. Stephenson. IntelliStreets: Digital Scaffolding for 'Smart' Cities. `http://www.huffingtonpost.com/w-david-stephenson/intellistreets_b_1242972.html`, 2012. [Online; accessed 2016-04-04].

[89] J. Stierwalt. Formula 1 and HANA: How F1 Racing is Pioneering Big Data Analytics. `http://jeremystierwalt.com/2014/01/29/formula-1-and-hana-how-f1-racing-is-pioneering-big-data-analytics/`, 2014. [Online; accessed 2016-02-16].

[90] M. Stolpe, K. Bhaduri, and K. Das. Distributed Support Vector Machines: An Overview. In *Solving Large Scale Learning Tasks: Challenges and Algorithms*, volume 9580 of *LNCS*. 2016. [to appear].

[91] M. Stolpe, H. Blom, and K. Morik. Sustainable Industrial Processes by Embedded Real-Time Quality Prediction. In J. Lässig, K. Kerstin, and K. Morik, editors, *Computational Sustainability*, volume 9570 of *LNCS*, pages 207–251. Springer, Berlin, Heidelberg, 2016.

[92] M. Stolpe, T. Liebig, and K. Morik. Communication-efficient learning of traffic flow in a network of wireless presence sensors. In *Proc. of the Workshop on Parallel and Distributed Computing for Knowledge Discovery in Data Bases (PDCKDD)*, CEUR Workshop Proceedings, page (to appear). CEUR-WS, 2015.

[93] M. Stolpe and K. Morik. Learning from Label Proportions by Optimizing Cluster Model Selection. In *Proc. of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 3, pages 349–364, Berlin, Heidelberg, 2011. Springer-Verlag.

[94] M. Stolpe, K. Morik, B. Konrad, D. Lieber, and J. Deuse. Challenges for Data Mining on Sensor Data of Interlinked Processes. In *Proceedings of the Next Generation Data Mining Summit (NGDM) 2011*, 2011.

[95] L. Sweeney. K-anonymity: A Model for Protecting Privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, Oct. 2002.

[96] J. Tapper. Obama administration spied on German media as well as its government. 2015. [Online; accessed: 2016-03-30].

[97] The Apache Software Foundation. Apache Flink: Scalable Batch and Stream Data Processing. `http://flink.apache.org/`, 2015. [Online; accessed 2015-12-15].

[98] The Apache Software Foundation. mahout. `http://mahout.apache.org/`, 2015. [Online; accessed 2016-03-30].

[99] N. Treiber, J. Heinermann, and O. Kramer. Wind Power Prediction with Machine Learning. In J. Lässig, K. Kersting, and K. Morik, editors, *Computational Sustainability*, volume 9570 of *LNCS*. Springer, 2016.

[100] C.-W. Tsai, C. Lai, M. Chiang, and L. Yang. Data Mining for Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials*, 16(1):77–97, 2014.

[101] C.-W. Tsai, C.-F. Lai, and A. Vasilakos. Future Internet of Things: open issues and challenges. *Wirel. Netw.*, 20(8):2201–2217, 2014.

[102] United Nations. World Urbanization Prospects. `http://esa.un.org/unpd/wup/Publications/Files/WUP2014-Report.pdf`, 2014. [Online; accessed 2016-04-04].

[103] A. K. R. Venkatapathy, A. Riesner, M. Roidl, J. Emmerich, and M. ten Hompel. PhyNode: An intelligent, cyber-physical system with energy neutral operation for PhyNetLab. In *Proc. of the Europ. Conf. on Smart Objects, Systems and Technologies, Smart SysTech*, 2015.

[104] Verizon. State of the Market: The Internet of Things 2015. http://www.verizonenterprise.com/state-of-the-market-internet-of-things/, 2015. [Online; accessed 2015-10-22].

[105] C. Wang, Z. Bi, and L. D. Xu. IoT and Cloud Computing in Automation of Assembly Modeling Systems. *IEEE T. Ind. Inform.*, 10(2):1426–1434, 2014.

[106] M. Weiser. The Computer for the 21st Century. *Sci. Am.*, 265(9), 1991.

[107] T. White. *Hadoop: The Definitive Guide*. O'Reilly, USA, 2nd edition, 2011.

[108] B. Wolff, E. Lorenz, and O. Kramer. Statistical Learning for Short-Term Photovoltaic Power Predictions. In J. Lässig, K. Kersting, and K. Morik, editors, *Computational Sustainability*, volume 9570 of *LNCS*. Springer, Berlin, Heidelberg, 2016.

[109] E. Woods. Smart Street Lights Face Financial Hurdles. https://www.navigantresearch.com/blog/smart-street-lights-face-financial-hurdles, 2012. [Online; accessed 2016-04-04].

[110] L. Xu, W. He, and S. Li. Internet of Things in Industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.

[111] H. Yunhong, F. Liang, and H. Guoping. Privacy-Preserving SVM Classification on Vertically Partitioned Data without Secure Multi-party Computation. In *5th Int. Conf. on Natural Computation (ICNC)*, volume 1, pages 543–546, Aug. 2009.

[112] J. Zhang, D. Roy, S. Devadiga, and M. Zheng. Anomaly detection in MODIS land products via time series analysis. *Geo-spatial Information Science*, 10(1):44–50, 2007.

[113] Y. Zhang, J. Duchi, M. Jordan, and M. Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS) 26*, pages 2328–2336. Curran Associates, Inc., 2013.

[114] Y. Zhao, R. Schwartz, E. Salomons, A. Ostfeld, and H. Poor. New formulation and optimization methods for water sensor placement. *Environmental Modelling & Software*, 76:128–136, 2016.

[115] Y. Zheng, S. Rajasegarar, C. Leckie, and M. Palaniswami. Smart car parking: Temporal clustering and anomaly detection in urban car parking. In *IEEE 9th Int. Conf. on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 1–6, 2014.

[116] K. Zhou and S. Yang. Understanding household energy consumption behavior: The contribution of energy big data analytics. *Renew. Sust. Energ. Rev.*, 56:810–819, 2016.

[117] I. Žliobaitė, M. Pechenizkiy, and J. Gama. An Overview of Concept Drift Applications. In N. Japkowicz and J. Stefanowski, editors, *Big Data Analysis: New Algorithms for a New Society*, pages 91–114. Springer International Publishing, 2016.

MultiClust 2013: Multiple Clusterings, Multi-view Data, and Multi-source Knowledge-driven Clustering

[Workshop Report]

Ira Assent[1], Carlotta Domeniconi[2], Francesco Gullo[3], Andrea Tagarelli[4], Arthur Zimek[5]

[1]Dept. of Computer Science, Aarhus University, Denmark
ira@cs.au.dk

[2]George Mason University, USA
carlotta@cs.gmu.edu

[3]Yahoo Labs, Spain
gullo@yahoo-inc.com

[4]DIMES, University of Calabria, Italy
tagarelli@dimes.unical.it

[5]Ludwig-Maximilians-Universität München, Germany
zimek@dbs.ifi.lmu.de

ABSTRACT

In this workshop report, we give a summary of the MultiClust workshop held in Chicago in conjunction with KDD 2013. We provide an overview on the history of this workshop series and the general topics covered. Furthermore, we provide summaries of the invited talks and of the contributed papers.

1. INTRODUCTION

Multiple views and data sources require clustering techniques capable of providing several distinct analyses of the data. The cross-disciplinary research topic on multiple clustering has thus received significant attention in recent years. However, since it is relatively young, important research challenges remain. Specifically, we observe an emerging interest in discovering multiple clustering solutions from very high dimensional and complex databases. Detecting alternatives while avoiding redundancy is a key challenge for multiple clustering solutions. Toward this goal, important research issues include: how to define redundancy among clusterings; whether existing algorithms can be modified to accommodate the finding of multiple solutions; how many solutions should be extracted; how to select among far too many possible solutions; how to evaluate and visualize results; and eventually how to most effectively help the data analysts in finding what they are looking for. Recent work tackles this problem by looking for non-redundant, alternative, disparate, or orthogonal clusterings. Research in this area benefits from well-established related areas, such as ensemble clustering, constraint-based clustering, frequent pattern mining, theory on result summarization, consensus mining, and general techniques coping with complex and high dimensional databases. At the same time, the topic of multiple clustering solutions has opened novel challenges in these research fields.

Overall, this cross-disciplinary research endeavor has recently received significant attention from multiple communities. The MultiClust workshop is a venue to bring together researchers from the above research areas to discuss issues in multiple clustering discovery.

MultiClust 2013 was the 4th in a series of workshops. The first MultiClust workshop was an initiative of Xiaoli Fern, Ian Davidson, and Jennifer Dy and was held in conjunction with KDD 2010 [7]. The successful workshop series continued with the 2nd MultiClust workshop at ECML PKDD 2011 [10] and the 3rd MultiClust workshop at SIAM Data Mining 2012 [11]. Additionally, an upcoming special issue of the Machine Learning Journal is dedicated to the MultiClust topics. The aim of this special issue is to establish an overview of recent research, to increase its visibility, and to link it to closely related research areas.

Furthermore, in 2012, the 3Clust workshop was held in conjunction with PAKDD [6]. It had a slightly different perspective but is very related to the MultiClust workshop topics. Therefore, the organizers of 3Clust and some organizers of previous MultiClust workshops teamed up for the 4th

MultiClust workshop at KDD 2013, giving more emphasis not only on emerging issues in the areas of clustering ensembles, semi-supervised clustering, subspace/projected clustering, co-clustering, and multi-view clustering, but in particular on discussing new and insightful connections between these areas. The vision is to make progress towards a unified framework that reconciles the different involved variants of the clustering problem.

2. SUMMARY OF THE WORKSHOP

2.1 Invited Talks

At MultiClust 2013 we had two very inspiring invited talks, by Michael R. Berthold (University of Konstanz, Germany) and by Shai Ben-David (University of Waterloo, Canada).

Michael discussed his approach to learning in "Parallel Universes", with a focus on the application area of bio-chemical and medical research (drug discovery). In this area, data objects are represented in very different, heterogeneous feature spaces and complex data types such as molecular structures or sequences, resulting also in different notions of similarity. Objects that could be similar (and should be clustered) in one of the representations might be very dissimilar in another representation. At the same time, different data representations are of different quality and partly faulty, outdated, unreliable or just noisy.

Learning (or clustering) in parallel universes is similar but also different in some crucial aspects from related approaches to clustering. If the data objects are represented in a high-dimensional but essentially homogeneous, numeric feature space, we have a global similarity measure that can be used for clustering. For multiple feature spaces of heterogeneous nature, many notions of similarity would be required. This is different from feature selection for clustering, where one would choose the most informative or useful subset of attributes. For a specific subset, there is usually no interpretation possible. Feature selection approaches select a subset of features from one, large universe and serve as preprocessing for subsequent learning algorithms. Similarly, projected clustering or subspace clustering selects subsets of features for each cluster however not as a preprocessing but as an integral step of the clustering procedure. Nevertheless, the features of the complete feature space are thought to belong to the same universe and the projected clustering or subspace clustering algorithm works on this complete, single feature space to select appropriate subsets. For clustering in parallel universes, different subsets of features are separated semantically from each other by their different nature in the first place. The most similar notion is multi-view or multi-represented learning; the idea there, however, is that the same concept can be learned in different representations and, especially in the setting of co-learning, the learning process in one representation can help or guide the learning process in another representation. In multi-instance learning, finally, the same object can have different representations in the same feature space, for example, a molecule can have different 3D confirmations.

For the specific approach of learning in parallel universes, Michael discussed some example approaches. Fuzzy cu-Means [13] is an adaptation from the fuzzy k-means family to the setting of parallel universes, where representations in some universes can be completely noisy. For the example of neighborgram clustering [5], Michael demonstrated the possibilities for the domain scientist to understand decisions of the algorithm and gain insights by an illustrative view of the results.

The other invited talk by Shai was focused on the gap between theory and practice in clustering. Although clustering is one of the most widely used tools in data analysis and exploration, it is not clear a priori what a good clustering is for a dataset. For many datasets, different clustering solutions can be equally meaningful. How to turn clustering in an actually well-defined task depends on the application, i.e., the domain expert can add some bias, expressing domain knowledge. How to formalize such a bias is the motivating question for the points Shai was taking in his talk [14; 1; 2]. In particular, he discussed (1) general properties of the input-output functionality of clustering paradigms, (2) quality measures for clusterings, and (3) measures for the clusterability of data.

1. In general terms, if we consider functions that take as input a dissimilarity function over some domain S (or, alternatively, a matrix of pairwise "distances" between points in the domain), and provide as output a partition of S, we would like to have properties that can distinguish "clustering" functions from other functions that output partitions. The ideal theory would define sets of properties to distinguish major clustering paradigms from each other. This could even work hierarchically. Shai showed examples of sets of properties defining single-linkage clustering, and sets of properties defining linkage clustering.

2. A different approach for defining clustering paradigms is given by measures of clustering quality. These can also be analyzed with an axiomatic approach. Shai names the properties "scale invariance", "consistency", "richness", and "isomorphism invariance" as a consistent set of properties and names many clustering quality measures that satisfy these axioms.

3. Finally, clusterability can be seen as applying clustering quality measures on optimal clustering solutions for a dataset.

As can be seen, the two invited talks covered very different perspectives, Michael taking a perspective from his practical application, Shai sharing thoughts from a theoretical point of view. This broadness of aspects is a good reflection of the scope of the MultiClust workshop series.

2.2 Research Papers

The contribution by Li et al. [9] discusses an approach to multi-view clustering based on a Markov Chain Monte Carlo sampling of relevant subspaces. A subspace is a subset of input features, and is considered to be a state of a Markov chain. The neighbors of a given state in the chain are the immediate subsets (one feature removed) and supersets (one feature added). The search in the chain is driven by the assessed quality of the clustering structure in the corresponding subspaces. Furthermore, in order to facilitate the discovery of diverse views of the data, the search is biased in favor of those subspaces that are dissimilar from the previously detected ones.

Clusters in subspaces are detected using the Mean Shift algorithm, which is based on a non-parametric kernel density estimation approach. The quality of a subspace is measured in terms of the density of the clusters discovered therein. A weighting term, measuring the similarity with previously detected subspaces, is added to the density function, with the effect of favoring the sampling of subspaces dissimilar from one another. Two sampling processes are investigated: simulated annealing and greedy local search.
The preliminary results measuring clustering quality are encouraging. Scaling the proposed method to a large dimensionality, and the automatic identification of the number of views, are interesting open challenges for future directions the authors plan to pursue.

Babagholami-Mohamadabadi et al. [4] focus on the problem of distance-metric learning in a semi-supervised context. The problem consists in learning an appropriate metric distance for an input set of points based on a number of must-link and/or cannot-link constraints that are defined over the input points. The main novelty of the approach by Babagholami-Mohamadabadi et al. is that, unlike most existing methods, it can profitably take advantage of the data points that are not involved into any constraints. Based on this intuition, the authors develop a novel linear metric-learning method, which they also kernelize so to develop a non-linear version of the same method. The optimization strategy relies on the Deterministic Annealing EM (DAEM) algorithm, which allows for finding a local maximum of the proposed objective function.

Shiga and Mamitsuka [12] introduce a probabilistic generative approach to co-clustering that enables the embedding of auxiliary information. External information associated to both the rows and columns of the data matrix can be added and incorporated in the inference process. The parameters over the row and column clusters are learned via variational inference using an Expectation Maximization-style algorithm. The authors test the effectiveness of the proposed method using a gene expression dataset. They represent the auxiliary information as graphs that connect genes (or samples) known to be in the same cluster, according to the ground truth. Comparisons against unsupervised Bayesian co-clustering are in favor to the proposed technique, showing the positive effect of embedding the external information. Semi-supervised co-clustering is a relevant approach in a variety of applications, including text mining and recommender systems, where information regarding the users and products allows us to perform prediction for new users.

Kamishima and Akaho [8] distinguish "Absolute and Relative Clustering". This difference is intended to relate to the relationship between the data set and the clustering result. In absolute clustering, the decision to cluster two objects in the same cluster is independent of other objects. In relative clustering, the decision to cluster two objects in the same cluster is depending on other objects, i.e., the clustering task as a whole. The authors present several examples for their intuition. In the discussion, Shai Ben-David questioned the idea by assuming that the class of absolute clustering is probably empty. It would seem, however, that the authors' distinction can be an original approach to think about semi-supervised clustering, where pairwise instance-level constraints indeed specify desired decisions for pairs of objects independently of the remainder of that data set. How a (semi-supervised) clustering approach addresses such constraints would be a different question.

Spectral graph partitioning is the topic addressed in the short paper by Zheng and Wu [15]. The basic motivation for this study is to try overcome an accuracy issue in spectral modularity optimization — the repeated bisection process performed by a traditional spectral modularity optimization algorithm can fail in reaching global optimality due to its greedy nature. In order to take into account the global structure information in a graph, the spectral algorithm proposed by Zheng and Wu aims to find better, multisection divisions of the graph by extending the modularity matrix to a higher order and making use of orthogonal vectors of the Hadamard matrix for the representation of group assignments of the vertices in the graph divisions. The modularity matrix is randomly "inflated" to higher orders through the Kronecker product, as to coordinate with the orthogonal vectors. As a result, the graph can be cut into multiple sections directly. In sparse graphs, the time complexity is $O(K^4 n^2)$ for a graph of n vertices, where K is the estimated number of communities.

3. CONCLUSIONS AND OUTLOOK

Clustering is a very traditional data mining task but at the same time provides many new challenges. The MultiClust workshop brings together researchers working at different aspects of the clustering problem with a particular focus on making use of multiple clustering solutions, envisioning a unified framework reconciling and integrating the different aspects of the clustering problem.
A continuation of the MultiClust workshop series is planned as a Mini-Symposium at SIAM Data Mining (SDM) 2014: `http://uweb.dimes.unical.it/multiclust2014/`.

Acknowledgements

We would like to thank all the authors. Their creativity made the workshop a success. Also all participants shared their thoughts in our discussions. In particular we thank the invited speakers for sharing their insights and experience in our topic. Last but not least we acknowledge the good work and effort of all members of the program committee. They provided high quality reviews for our submissions in a tight schedule. The members of the program committee in alphabetical order are:

- James Bailey, University of Melbourne, Australia
- Ricardo J. G. B. Campello, University of São Paulo, Brazil
- Xuan-Hong Dang, Aarhus University, Denmark
- Ines Färber, RWTH Aachen University, Germany
- Wei Fan, IBM T. J. Watson Research Center and IBM CRL, USA
- Ana Fred, Technical University of Lisbon, Portugal
- Stephan Günnemann, CMU, USA
- Dimitrios Gunopulos, University of Athens, Greece

- Michael E. Houle, NII, Japan
- Emmanuel Müller, KIT, Germany
- Erich Schubert, LMU Munich, Germany
- Thomas Seidl, RWTH Aachen University, Germany
- Grigorios Tsoumakas, Aristotle University of Thessaloniki (AUTh), Greece
- Giorgio Valentini, University of Milan, Italy
- Jilles Vreeken, University of Antwerp, Belgium

4. REFERENCES

[1] M. Ackerman and S. Ben-David. Clusterability: A theoretical study. *Journal of Machine Learning Research - Proceedings Track*, 5:1–8, 2009.

[2] M. Ackerman, S. Ben-David, and D. Loker. Towards property-based classification of clustering paradigms. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *NIPS*, pages 10–18. Curran Associates, Inc., 2010.

[3] I. Assent, C. Domeniconi, F. Gullo, A. Tagarelli, and A. Zimek, editors. *4th MultiClust Workshop on Multiple Clusterings, Multi-view Data, and Multi-source Knowledge-driven Clustering, in conjunction with the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA*, 2013.

[4] B. Babagholami-Mohamadabadi, A. Zarghami, H. A. Pourhaghighi, and M. T. Manzuri-Shalmani. Probabilistic non-linear distance metric learning for constrained clustering. In Assent et al. [3], pages 4:1–4:8.

[5] M. Berthold, B. Wiswedel, and D. Patterson. Interactive exploration of fuzzy clusters using neighborgrams. *Fuzzy Sets and Systems*, 149(1):21–37, 2005.

[6] C. Domeniconi, F. Gullo, and A. Tagarelli, editors. *The First International Workshop on Multi-view data, High Dimensionality, and External Knowledge: Striving for a Unified Approach to Clustering, in conjunction with PAKDD 2012, Kuala Lumpur, Malaysia*, 2012.

[7] X. Z. Fern, I. Davidson, and J. G. Dy. MultiClust 2010: discovering, summarizing and using multiple clusterings. *SIGKDD Explorations*, 12(2):47–49, 2010.

[8] T. Kamishima and S. Akaho. Absolute and relative clustering. In Assent et al. [3], pages 6:1–6:6.

[9] G. Li, S. Günnemann, and M. J. Zaki. Stochastic subspace search for top-k multi-view clustering. In Assent et al. [3], pages 3:1–3:6.

[10] E. Müller, S. Günnemann, I. Assent, and T. Seidl, editors. *2nd MultiClustWorkshop: Discovering, Summarizing and Using Multiple Clusterings, in conjunction with ECML PKDD 2011, Athens, Greece*, 2011.

[11] E. Müller, T. Seidl, S. Venkatasubramanian, and A. Zimek, editors. *3rd MultiClust Workshop: Discovering, Summarizing and Using Multiple Clusterings, in conjunction with SIAM Data Mining 2012, Anaheim, CA*, 2012.

[12] M. Shiga and H. Mamitsuka. Variational Bayes coclustering with auxiliary information. In Assent et al. [3], pages 5:1–5:4.

[13] B. Wiswedel and M. R. Berthold. Fuzzy clustering in parallel universes. *International Journal of Approximate Reasoning*, 45(3):439–454, 2007.

[14] R. Zadeh and S. Ben-David. A uniqueness theorem for clustering. In J. Bilmes and A. Y. Ng, editors, *UAI*, pages 639–646. AUAI Press, 2009.

[15] H. Zheng and J. Wu. Spectral graph multisection through orthogonality. In Assent et al. [3], pages 7:1–7:6.

Current and Future Challenges in Mining Large Networks: Report on the Second SDM Workshop on Mining Networks and Graphs

Lawrence B. Holder
Washington State University

Maleq Khan
Virginia Tech

Christine Klymko
Lawrence Livermore

Rajmonda Caceres
MIT Lincoln Lab

Nitesh V. Chawla
University of Notre Dame

Tina Eliassi-Rad
Rutgers University

David F. Gleich
Purdue University

Ravi Kumar
Google, Inc.

Aditya Prakash
Virginia Tech

Jason Riedy
Georgia Tech

Yinghui Wu
Washington State University

ABSTRACT

We report on the Second Workshop on Mining Networks and Graphs held at the 2015 SIAM International Conference on Data Mining. This half-day workshop consisted of a keynote talk, four technical paper presentations, one demonstration, and a panel on future challenges in mining large networks. We summarize the main highlights of the workshop, including expanded written summaries of the future challenges provided by the panelists. The current and future challenges discussed at the workshop and elaborated here provide valuable guidance for future research in the field.

Keywords

Network mining, graph mining, big data, challenges.

1. INTRODUCTION

Real-world applications give rise to networks that are unstructured and often comprised of several components. Furthermore, they can support multiple dynamical processes that shape the network over time. Network science refers to the broad discipline that seeks to understand the underlying principles that govern the synthesis, analysis and co-evolution of networks. In some cases, the data relevant for mining patterns and making decisions comes from multiple heterogeneous sources and streams in over time. Graphs are a popular representation for such data because of their ability to represent different entity and relationship types, including the temporal relationships necessary to represent the dynamics of a data stream. However, fusing such heterogeneous data into a single graph or multiple related graphs and mining them are challenging tasks. Emerging massive data has made such tasks even more challenging.

The 2015 SDM Workshop on Mining Networks and Graphs [21] brought together researchers and practitioners in the field to deal with the emerging challenges in processing and mining large-scale networks. Such networks can be directed as well as undirected, they can be labeled or unlabeled, weighted or unweighted, and static or dynamic. Networks of networks are also of interest. Specific scientific topics of interest for this meeting include mining for patterns of interest in networks, efficient algorithms (sequential/parallel, exact/approximation) for analyzing network properties, methods for processing large networks (i.e., Map-Reduce and Giraph based frameworks), use of linear algebra and numerical analysis for mining complex networks, database techniques for processing networks, and fusion of heterogeneous data sources into graphs. Another particular topic of interest is to couple structural properties of networks to the dynamics over networks, e.g., contagions.

The workshop consisted of a keynote talk by Ravi Kumar from Google, four technical paper presentations, a demonstration of the CINET Cyberinfrastructure for Network Science by Maleq Khan from Virginia Tech, and a panel on Future Challenges in Mining Large Networks. The panelists included Rajmonda Caceres from MIT Lincoln Lab, Nitesh Chawla from Notre Dame, Tina Eliassi-Rad from Rutgers, David Gleich from Purdue, Christine Klymko from Lawrence Livermore, Ravi Kumar from Google, Jason Riedy from Georgia Tech, Aditya Prakash from Virginia Tech, and Yinghui Wu[1] from Washington State. The workshop was co-chaired by Lawrence Holder from Washington State, Maleq Khan from Virginia Tech, and Christine Klymko.

In the following sections we summarize the presentations and discussions at the workshop. Each panelist has also provided a written summary elaborating on their future challenge.

2. CURRENT DIRECTIONS FOR MINING NETWORKS AND GRAPHS

Ravi Kumar gave a keynote talk entitled "Estimating Network Parameters." Estimating the parameters such as the size and average degree of a large network, which cannot be accessed in its entirety, is a basic data mining question. Recently, the problems of estimating the size of the web, the size of a web index, the size and other parameters of online social networks, etc. have been actively considered in the context of World Wide Web [12]. In this talk, Ravi Kumar addressed several questions with the main focus on estimating the network size and the average degree. The main motivation of estimating these parameters is to understand the network in general. In the case of social network, it can help in gaining business insight and competitive advantage [12]. These

[1] Yinghui Wu was unable to attend the workshop due to last minute visa issues, but we have included the written summary of his challenge in this report.

problems become challenging and interesting with the following realistic assumptions: i) the network is not available to us in its entirety – we can only query a node and obtain all its neighbors, ii) these queries are expensive, and thus an algorithm has to make a small number of queries, and iii) it may not be possible to access a uniformly random node in the network. The speaker discussed some traditional methods and then showed some recently developed advanced techniques that reduce the number of queries significantly.

Four contributed papers [2, 10, 16, 38] were presented in the workshop. These papers have also been published in the workshop proceedings. In [2], the authors addressed the problem of mining coevolving patterns in dynamic networks. They present an algorithm to analyze all relational changes between entities (nodes) and find all frequent coevolving induced relational motifs. Their results show that these motifs capture network characteristics that can be useful for modeling the underlying dynamic network. A recent trend and important problem in graph mining is to mine social, financial, or other relevant networks for detecting intrusion and suspicious activities. Another paper [16] presents a method of detecting intrusion using frequent subgraphs. Community detection in a network is another important problem and recently received significant attention of the researchers. Large-scale networks (networks with billions of nodes and edges) require very efficient algorithms. Some efficient methods for detecting communities in large-scale networks are presented in [38] and [10].

Maleq Khan gave a demonstration of an open–access web-based network analysis tool called CINET [1, 15], a Cyber Infrastructure for NETwork Science[2]. CINET has been developed at Virginia Tech and partially funded by NSF. It provides a large set of networks and modules (such as computing diameter, clustering coefficient and shortest path) to analyze them. Users can also add their own networks to be analyzed by the provided algorithms. The web-based interface has been designed to simplify analysis of complex networks for users who are not necessarily computer scientists.

3. FUTURE CHALLENGES ON MINING LARGE NETWORKS

While the panelists had only three minutes each to present their challenge at the workshop, they have also provided written descriptions after the workshop, which are included here.

3.1 Graph Representation Learning[3]

Rajmonda Caceres, MIT Lincoln Laboratory

The process of going from raw data to the right graph representation is a critical building block for a successful data-to-decisions analytical framework. When properly done, the graph representation captures the essential aspects of the data and abstracts away the noisy, irrelevant parts. Many inference algorithms make two fundamental assumptions: 1) the graph is already constructed 2) the constructed graph has the qualitative properties necessary for their analysis to work, i.e., the patterns that we are looking for are present and recoverable. In reality, what we have available is raw data that is often noisy and collected from different modalities. Furthermore, no clear methodology exists in place for converting these data into a useful graph representation. Current practices often aggregate different graph sources ad-hoc, making it difficult to compare algorithms across different domains or even within the same domain using different data sources. The immediacy for rigorous approaches on representation learning of graphs is even more apparent in the big data regime, where challenges connected to variety and veracity exacerbate the challenges of volume and velocity.

Constructing quality graph representations from raw data is a challenging task. Often the data we collect represent indirect measurement of the true relationships we want to analyze, for example, we want to analyze social relationships, but we collect proximity information. Data collections systems often introduce a lot of noise in the form of missing or irrelevant connections. Finally, it is not clear how to integrate different, potentially complementary data sources into one unified representation.

An orthogonal challenge has to do with our mathematical understanding (or lack of) of what makes a graph representation qualitative. If we did have a good understanding of this, we could then hope to design algorithms to drive the data-to-graph mapping in the right direction. In reality, we do not have ground truth, nor do we have notions of quality that we agree upon. More importantly, we often observe that the quality of graph representation depends on the objective of the learning task, and for the same learning task, multiple graph representations might be useful.

A much-needed capability in this problem setting is one that takes multi-source, incomplete, noisy data and constructs quality networks together with estimations of uncertainty/confidence of the network components (edges, subgraphs, etc.). There are additional related open research questions and potential areas of impact, from developing methods for validating the quality of graph representation in the absence of ground truth, to identifying scenarios when fusion of different sources helps, to deriving performance guarantees for different graph construction or graph recovery techniques.

3.2 Representing Higher-Order Dependencies in Networks

Nitesh V. Chawla, University of Notre Dame

How to construct the network representation from data, such that the underlying phenomena in data are correctly captured and represented?

The conventional way of constructing a network from raw data typically assumes the Markov property (first order dependency) by considering only the pairwise connections in data. That is, in such a network, a movement simulation (such as trajectories of vehicles, retweets, clickstream traffic, etc.) is only able to follow the probability distribution of the first order, and cannot reflect higher order dependencies that may exist in the data. This can lead to inaccuracies when applying a wide range of network analyses tools that are based on the simulations of movements in the network, such as clustering, PageRank, various link prediction methods based on random walking, and so on.

[2] http://www.vbi.vt.edu/ndssl/cinet.

[3] This challenge is part of work sponsored by the Department of the Air Force under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Specifically, the challenge that we posit is as follows. Construct a network that accurately captures the variable and higher order of dependencies such that it is:

a) representative of the underlying phenomena in the data to more accurately represent simulation of movement

b) compact in size allowing for variable order of dependencies versus using a fixed high order

c) compatible with existing network analysis tools such that the analysis toolkit does not have to change to respond to the network representation

3.3 Provably Increasing Network Awareness

Tina Eliassi-Rad, Rutgers University

The underlying processes generating network data are often partially observed. Thus, regardless of how big the data is, it is incomplete and noisy. For example, current maps of the Internet are known to be incomplete and significantly biased [18]. The challenge is to provably increase network awareness. Specifically, given an incomplete, noisy, and possibly biased network, can we infer network properties (at micro, mezzo, and macro levels) with provable accuracy? Then, given these inferences can we design active graph probing/learning algorithms for graph mining tasks (such as community detection, role extraction, etc.)? Approaches from computer science theory such as property testing [14, 32] and sublinear algorithms [33] and from machine learning such as active learning [6, 30] are possible solutions to this solution.

This challenge is joint work with Sucheta Soundarajan (Rutgers University), Brian Gallagher (Lawrence Livermore National Laboratory), Ali Pinar (Sandia National Laboratories), C. Seshadhri (University of California Santa Cruz), and Bradley Huffaker (CAIDA).

3.4 A Turing Test for Synthetic Network Models

David F. Gleich, Purdue University

We propose establishing a Turing-like test to assess the current state of synthetic network models. Synthetic network models are important for two problems: (i) assessing the statistical significance of results in networks [28] and (ii) measuring the performance of new algorithms on extremely large graphs [7]. But there is widespread disagreement about the relevance of the current state of synthetic generators. New models are constantly being proposed to fit the latest observed feature of real-world networks (see, for instance [24]). The ones that see widespread use are often due to reasons that are distinct from their accuracy as far as modeling real networks [29, 36]. The basis for our proposal is to quantify the current state of synthetic network models and address the question: can we distinguish the distribution of graphs generated by synthetic methods from the distribution of graphs that are from the real-world?

A hypothetical model for such a test is as follows. At the start of each month, there is a new collection of networks released. These networks are either generated by a synthetic generator or a piece of a real-world dataset. At the end of the month, challengers would submit their results on if they believe that the network was the result of a generator or a real-world network. If the current state of synthetic network generation is sufficient, then the two distributions should be indistinguishable. If there are distinguishing features, this suggests how we need to improve current synthetic generators.

Justification. One of the irksome questions in graph mining is trying to determine if a finding is significant or if should have been expected given the known properties of social networks. An approach to answer this question for many subgraph and subset queries involves studying synthetic network models of networks and evaluating the likelihood of finding that subgraph or subset in the synthetic model (or one with similar properties). But this methodology is only useful if the synthetic model *has* the properties that are known be associated with the original class of networks and also has some variance over the distribution of graphs [27]. It is unclear if the current class of synthetic models meets these requirements and the Turing test proposed above would help us answer that question and would also suggest important properties to distinguish real-world networks from their synthetic approximations.

Additionally, extremely large graphs are difficult to find outside of a small number of select institutions such as Google and the NSA. The largest publically available network is 126B edges and 6B vertices (http://webdatacommons.org). There are many problems with this graph that can be solved on a modern laptop computer [26]. One of the approaches to overcome the lack of data is to evaluate synthetic networks that can be generated at arbitrary size-scales. But the relevance of these networks to algorithmic performance is questionable if the underlying networks are not a reasonable approximation. This is especially important for things like partitioning problems where many synthetic networks have relatively simple optimal partitioning strategies.

3.5 Noisy Data and Fuzzy Subgraph Detection

Christine Klymko, Lawrence Livermore National Laboratory

One important issue in dealing with network data is how to account for noise. Noisy data can result from a variety of processes, including: collection error (missing edges, false edges, etc.), mutations (such as those occurring in certain biological networks), actual but unimportant/meaningless interactions (i.e., wrong number phone calls), and nodes attempting to hide their interactions in a network (such as might occur in various social or cybersecurity applications). The presence of noise complicates many data mining problems: see [17, 37], among others.

An example of the difficulties of data mining tasks in the presence of noisy data is the question of subgraph/network motif detection, which becomes especially complicated when noise is taken into account. Subgraph detection is important in a number of areas [19, 25]. However, given the presence of noise, it does not make sense to search for exact subgraphs. Instead, a search for "fuzzy" subgraphs (allowing the addition or deletion of a small number of nodes and edges to the original search query) will often produce more meaningful results. However, there are still few methodologies to effectively perform fuzzy subgraph detection. The development of noise robust methodologies (for subgraph detection and other data mining questions) is an important area of research.

3.6 Scalable Graph Algorithms in Emerging Computational Models

Ravi Kumar, Google

The challenge is to develop and study computational models that are best suited for large data, especially, large graphs. Modern computing paradigms such as streaming and map-reduce have been very useful in developing algorithms that can scale to large

data; these paradigms are reasonably well-established by now and their limitations are well understood. Emerging models such as the asynchronous computational model and the parameter-server model (popular in the machine learning community) seem promising for many new classes of problems; their power and limitations are yet to be understood both from theoretical and applied points of view. It becomes important to study these models and see their applicability to large-scale graph problems – the topic is nascent and rich.

3.7 Error and Sensitivity Analysis for Graphs

Jason Riedy, Georgia Institute of Technology

Most current graph analysis methods assume correct data and knowledge. However, this rarely occurs. We have little knowledge about and fewer models of the sensitivity of analysis results to errors. Graphs imperfectly represent some real phenomenon. "Friendships" in online social networks do not always reflect personal relationships, or the data is obscured for privacy reasons as in health data. Computation imperfectly analyzes the graph. Many problems are only approximated to fit within time or energy limitations. Many codes have subtle bugs. If some problem occurs once in a billion edges, massive graphs will uncover it. Other scientific computing areas have established frameworks for analyzing and addressing sensitivity to perturbations. We need mental and formal methods for addressing error and sensitivity in graph analysis results, and we need to condense those into rules of thumb for practitioners.

The wide range of graph analysis tasks will need a variety of approaches. Globally averaged properties like a graph's clustering coefficient often are not very sensitive to perturbations. Local properties, however, can be affected drastically. Experiments in Zakrzewska and Bader [40] imply that for a variety of graphs and edge dropping heuristics, nearly a quarter of the edges could be ignored while affecting the global clustering coefficient by at most 10%. The vector of local clustering coefficients changes in one-norm relative difference by 20% to 80% in the same range. Consider measuring or modeling error in connected components. The interpretation of error will change depending on the source of the graph data. If the graph is derived from thresholds, say from significance of protein-protein interaction measurements [4], the single threshold may provide leverage in defining a model for the overall graph. Discrete interaction networks as occur in criminal network analysis [8] will require other prediction methods, although meaningfully predicting interactions between disconnected components is (to this author's knowledge) an open problem.

Understanding graph analysis algorithms' sensitivity to error and perturbation is a step towards making graph analysis a solid scientific computing approach. Other scientific computing disciplines have error analysis frameworks that are distilled into basic rules of thumb for practitioners. We need to provide analysts and scientists with the same level of support for confidence in the graph analysis results.

3.8 Propagation over Networks

Aditya Prakash, Virginia Tech

How do contagions like Ebola and Influenza spread in population networks? How do malware propagate? How can blackouts spread on a nationwide scale? How do rumors spread on Twitter/Facebook? Which group should we market to for maximizing product penetration? Answering all these big-data questions involves the study of aggregated dynamics over complex connectivity patterns [5, 20, 23, 31]. Dynamic processes over networks can give rise to fascinating macroscopic behavior, leading to fundamental research problems which recur in multiple domains. Understanding such propagation processes will eventually enable us to manipulate them for our benefit, e.g., understanding dynamics of epidemic spreading over graphs helps design more robust policies for immunization.

These problems are typically very challenging, as they involve high-impact real-world applications as well as deep technical issues like the need for scalability and handling of heterogeneous noisy data in a principled manner. Data for these problems will typically come from domains like epidemiology and public health (both simulation and real data), social media (tweets, blog posts, movie ratings), cyber security (malware databases), historical (newspapers) and so on. Moreover, promising approaches seem to be very inter-disciplinary – drawing concepts and techniques ranging from theory and algorithms (combinatorial and stochastic optimization), systems (asynchronous computation) to machine learning/statistics (minimum description length, graphical models) and non-linear dynamics. Clearly, progress in this sphere holds great scientific as well as commercial value.

3.9 Resource-bounded Graph Mining

Yinghui Wu, Washington State University

An emerging challenge is to develop scalable mining techniques over massive network data with limited resource. Graph mining tasks such as subgraph pattern discovery are inherently expensive, and it is often hard to theoretically reduce the complexity. On the other hand, emerging applications require mining with limited computing resource, such as response time, space cost, energy constraints, etc. For example, applications in cyber network monitoring typically require the anomaly communication patterns be discovered in real-time [11]. The need for big graph mining with bounded resource and (guaranteed) high accuracy is evident in resource-intensive applications.

Recent study on resource bounded and budgeted graph search suggests to explore bounded fraction of graph data to generate approximate answers [13]. Data sketch, summary and compression techniques are applied to generate and query small synopsis from original graphs [3]. The effectiveness and possible performance guarantees of these approaches may rely on specific query classes, domain knowledge and data properties. A possible future direction is to leverage learning techniques and design resource-accuracy trade-off mining algorithms upon specific application need. This may also lead to adaptive mining tools that support large-scale graph analytics in cloud services.

3.10 Panel Discussion

In summary, the presented challenges focused on how best to represent data as a graph, especially noisy data with higher-order dependencies, and how to evaluate the quality of the resulting graph. Since any constructed graph necessarily represents a sample of the real world, how can we assess the quality of the sample and the certainty of the conclusions drawn from the data (e.g., error, sensitivity, and p-value for graphs)? Addressing these issues will help with other challenges related to the design and testing of scalable graph mining algorithms that take maximum advantage of limited resources. After the panelists presented their challenges, a lively discussion ensued among the panelists as they responded to questions from the audience and amongst themselves. Here, we summarize this discussion.

An interesting comment by one of the panelists related the experience of *how seemingly deterministic graph algorithms may yield different results simply by relabeling the nodes* in the graph. A question from the audience asked for an elaboration of the reasons behind such behavior, and the main reasons were the arbitrary ranking among nodes with equivalent values and the precision errors when computing these values, which may be extremely small or large. One panelist asked if this was really a problem, given that we do not always need exact answers to graph problems, e.g., when merely ranking nodes. Others pointed out that if these error-tolerant tasks are repeated or are part of a larger workflow, then errors may propagate, which brings us back to one of the focuses of the challenges: how to assess error in the networks and in the results of graph algorithms. In general, graph analysis is often interested in the solution and not necessarily in optimizing a specific metric. Approaching such a highly non-linear and bumpy problem from different directions/permutations will likely result in different locally-optimal solutions. This is a challenge as it expands the space of viable solutions and complicates the evaluation of algorithms for mining large networks.

Next, one of the panelists proposed a straw man argument of *whether truly big real-world graphs exist, or at least graphs whose size requires more memory and computational power than a modern laptop*. More realistically, are there large graphs that exceed readily available computational resources that cost less than $10,000? Specifically, while Facebook purports to have a real-world graph on the order of one trillion edges [34], and the National Security Agency purports to have a real-world graph with 70 trillion edges requiring one petabyte of storage [7], the largest publicly-available graph has around 128 billion edges [39]. The panelist argued that for most graph mining tasks, a laptop is sufficient for processing a graph on the order of 100 billion edges. Other panelists pointed out that even larger graphs can be constructed by combining multi-typed data from different sources (e.g., all of the web), or incorporating time as in clickstream and network traffic flow data. While such graphs are typically sampled from, filtered, or abstracted in order to fit within memory requirements, simply loading these graphs into memory can take hours. And computationally complex algorithms, such as finding high-order motifs, or simply rerunning algorithms under different experimental conditions, require considerable computational resources. Such graphs and graph algorithms can easily exceed the power of a laptop and/or the patience of the experimenter, but do such graphs exist?

And if we had such large real-world graphs, what would we do with them? What questions would we ask about them? One panelist pragmatically pointed out that the right questions are the ones that have a clear broader impact as defined by the National Science Foundation, the source of much of the funding for graph mining research. Obviously, large graphs allow us to test the scalability of our algorithms, but do we really need trillion-edge graphs to test scalability? Benchmark datasets exist, such as the Graph 500 [29], but the overhead of handling such large graphs becomes an obstacle to the very testing that the benchmarks are designed to support. Also, at some point we must consider the amount of energy necessary to answer the questions we wish to pose. As the area of sustainable computing has been contemplating energy consumption for computation, we as graph miners must also consider the limitations of what is practically computable. Finally, recent efforts in the area of graph stream mining may offer some hope for answering questions once thought intractable on one large graph by streaming the graph in over time.

In the absence of real-world, publicly–available graphs on the order of one trillion in size, one solution is to develop more advanced graph generators that better mimic real-world graph properties. In fact, one of the audience members asked the panelists to comment on the *challenges of generating such synthetic graphs while constraining multiple interdependent graph properties*. Even before we can address this challenge, we need a good model of the distribution of such graphs in the real world, and these models are difficult to obtain [22]. Clearly, no model can represent everything, but which properties are the critical ones to model? It seems that the only way to model real-world networks is to allow them to be built in a realistic way. For example, if you want a model of Wikipedia, then start your on online encyclopedia and monitor its growth. If you want to model email communication, then find a group of people willing to let you monitor their email communication (good luck with that). Currently there are very few robust graph generators, with the exceptions being RMAT [9] and BTER [35]. But RMAT is focused on realistically modeling only the degree distribution. BTER is focused on modeling both degree distribution and triangle distribution, but does a poor job of maintaining a realistic ratio between the two. And neither generator supports the recovery of ground truth, e.g., the true communities for validating community detection algorithms. Furthermore, some panelists pointed out that many algorithms that perform well on these synthetic graphs do not perform well on real-world graphs. The subject of anomalies also came up, and how they can be realistically generated. Manually-constructed anomalies can be inserted into synthetic graphs, but many real anomalies are as yet unimagined. All of this suggests that the proper modeling of real-world graphs, i.e., identifying the salient properties that control the behavior of real-world graphs, and efficiently generating these graphs, remains an important challenge for the field.

4. CONCLUSIONS

The 2015 SDM Workshop on Mining Networks and Graphs provides a valuable snapshot and look ahead for the field. Clearly, the challenge of dealing with large and dynamic graphs is of particular focus, especially choosing proper representations, handing noise, dealing with limited resources, summarization and statistical significance of network mining results. We hope that the workshop proceedings, as well as the summaries of the technical presentations and panel included in this report, will motivate future directions in the field.

5. ACKNOWLEDGMENTS

We would like to thank the workshop program committee for their help selecting a set of quality papers. We would also like to thank the SDM organizers, especially the workshop co-chairs Xiaoli Fern and Xifeng Yan, for their support and guidance in bringing this workshop together. Finally, we would like to thank the authors, panelists and attendees for their contributions and participation.

6. REFERENCES

[1] Abdelhamid, S.H.E.M. et al. 2014. {CINET} 2.0: {A} CyberInfrastructure for Network Science. *10th {IEEE} International Conference on e-Science, eScience 2014, Sao Paulo, Brazil, October 20-24, 2014* (2014), 324–331.

[2] Ahmed, R. and Karypis, G. 2015. Mining Coevolving Induced Relational Motifs in Dynamic Networks. *Proceedings of the 2nd SDM Workshop on Mining Networks and Graphs: A Big Data Analytic Challenge* (2015).

[3] Ahn, K.J., Guha, S. and McGregor, A. 2012. Graph Sketches: Sparsification, Spanners, and Subgraphs. *Proceedings of the 31st Symposium on Principles of Database Systems* (New York, NY, USA, 2012), 5–14.

[4] Bader, J.S., Chaudhuri, A., Rothberg, J.M. and Chant, J. 2004. Gaining confidence in high-throughput protein interaction networks. *Nat Biotech*. 22, 1 (Jan. 2004), 78–85.

[5] Bakshy, E., Rosenn, I., Marlow, C. and Adamic, L. 2012. The Role of Social Networks in Information Diffusion. *Proceedings of the 21st International Conference on World Wide Web* (New York, NY, USA, 2012), 519–528.

[6] Bilgic, M., Mihalkova, L. and Getoor, L. 2010. Active Learning for Networked Data. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (2010).

[7] Burkhardt, P. and Waring, C. 2013. An NSA Big Graph experiment. *Report NSA-RD-2013-056002v1* (2013).

[8] Calderoni, F. 2014. Identifying Mafia Bosses from Meeting Attendance. *Networks and Network Analysis for Defence and Security*. A.J. Masys, ed. Springer International Publishing. 27–48.

[9] Chakrabarti, D., Zhan, Y. and Faloutsos, C. 2004. R-MAT: A Recursive Model for Graph Mining. *SIAM International Conference on Data Mining* (2004).

[10] Cheng, Y. and Wang, N. 2015. Graph clustering by recursive membership identification in neighborhood. *Proceedings of the 2nd SDM Workshop on Mining Networks and Graphs: A Big Data Analytic Challenge* (2015).

[11] Choudhury, S., Holder, L.B., Chin, G., Agarwal, K. and Feo, J. 2015. A Selectivity based approach to Continuous Pattern Detection in Streaming Graphs. *Proceedings of the 18th International Conference on Extending Database Technology (EDBT)* (2015), 157–168.

[12] Dasgupta, A., Kumar, R. and Sarlos, T. 2014. On Estimating the Average Degree. *Proceedings of the 23rd International Conference on World Wide Web* (New York, NY, USA, 2014), 795–806.

[13] Fan, W., Wang, X. and Wu, Y. 2014. Querying Big Graphs Within Bounded Resources. *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2014), 301–312.

[14] Goldreich, O., Goldwasser, S. and Ron, D. 1998. Property Testing and Its Connection to Learning and Approximation. *J. ACM*. 45, 4 (Jul. 1998), 653–750.

[15] Hasan, S.M.S. et al. 2012. CINET: A Cyberinfrastructure for Network Science. *Proceedings of the 2012 IEEE 8th International Conference on E-Science (e-Science)* (Washington, DC, USA, 2012), 1–8.

[16] Herrera-Semenets, V., Acosta-Mendoze, M. and Gago-Alonso, A. 2015. A Framework for Intrusion Detection based on Frequent Subgraph Mining. *Proceedings of the 2nd SDM Workshop on Mining Networks and Graphs: A Big Data Analytic Challenge* (2015).

[17] Holstein, D., Goltsev, A. V and Mendes, J.F.F. 2013. Impact of noise and damage on collective dynamics of scale-free neuronal networks. *Phys. Rev. E*. 87, 3 (Mar. 2013), 32717.

[18] Huffaker, B., Fomenkov, M. and Claffy, K. 2012. Internet Topology Data Comparison. *CAIDA Technical Report* (2012).

[19] Kashtan, N., Itzkovitz, S., Milo, R. and Alon, U. 2004. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*. 20, 11 (2004), 1746–1758.

[20] Kempe, D., Kleinberg, J. and Tardos, É. 2003. Maximizing the Spread of Influence Through a Social Network. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2003), 137–146.

[21] Khan, M., Klymko, C. and Holder, L.B. eds. 2015. Proceedings of the Second Workshop on Mining Networks and Graphs. *SIAM International Conference on Data Mining* (2015).

[22] Kim, M. and Leskovec, J. 2012. Multiplicative Attribute Graph Model of Real-World Networks. *Internet Math.* 8, 1-2 (2012), 113–160.

[23] Koff, R.S. 1992. Infectious diseases of humans: Dynamics and control. By R.M. Anderson and R.M. May, 757 pp. Oxford: Oxford University Press, 1991. 95.00. *Hepatology*. 15, 1 (1992), 169.

[24] Leskovec, J., Kleinberg, J. and Faloutsos, C. 2007. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*. 1, 1 (Mar. 2007), 1–41.

[25] Li, X., Wu, M., Kwoh, C.-K. and Ng, S.-K. 2010. Computational approaches for detecting protein complexes from protein interaction networks: a survey. *BMC Genomics*. 11, Suppl 1 (2010).

[26] McSherry, F., Isard, M. and Murray, D.G. 2015. Scalability! But at what cost? *15th Workshop on Hot Topics in Operating Systems (HotOS XV)* (Kartause Ittingen, Switzerland, May 2015).

[27] Moreno, S., Kirshner, S., Neville, J. and Vishwanathan, S. 2010. Tied kronecker product graph models to capture variance in network populations. *Allerton'10* (2010), 17–61.

[28] Moreno, S. and Neville, J. 2013. Network Hypothesis Testing Using Mixed Kronecker Product Graph Models. *IEEE 13th International Conference on Data Mining (ICDM)* (Dec. 2013), 1163–1168.

[29] Murphy, R.C., Wheeler, K.B., Barrett, B.W. and Ang, J.A. 2010. Introducing the Graph 500. *Cray User's Group* (May 2010).

[30] Pfeiffer III, J.J., Neville, J. and Bennett, P.N. 2014. Active Exploration in Networks: Using Probabilistic Relationships for Learning and Inference. *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management* (New York, NY, USA, 2014), 639–648.

[31] Prakash, B.A., Chakrabarti, D., Faloutsos, M., Valler, N. and Faloutsos, C. 2011. Threshold Conditions for Arbitrary Cascade Models on Arbitrary Networks. *Proceedings of the 2011 IEEE 11th International Conference on Data Mining* (Washington, DC, USA, 2011), 537–546.

[32] Ron, D. 2010. *Algorithmic and Analysis Techniques in Property Testing*. Now Publishers Inc.

[33] Rubinfield, R. 2006. Sublinear Time Algorithms. *Proceedings of the International Conference of Mathematicians* (2006).

[34] Scaling Apache Giraph to a Trillion Edges: 2013. *https://www.facebook.com/notes/facebook-engineering/scaling-apache-giraph-to-a-trillion-edges/10151617006153920*.

[35] Seshadhri, C., Kolda, T.G. and Pinar, A. 2012. Community structure and scale-free collections of Erdős-Rényi graphs. *Phys. Rev. E*. 85, 5 (May 2012).

[36] Seshadhri, C., Pinar, A. and Kolda, T. 2011. An In-Depth Study of Stochastic Kronecker Graphs. *Proceedings of IEEE International Conference on Data Mining* (2011).

[37] Subramaniyam, N.P. and Hyttinen, J. 2014. Characterization of dynamical systems under noise using recurrence networks: Application to simulated and {EEG} data. *Physics Letters A*. 378, 46 (2014), 3464–3474.

[38] Wang, H., Zheng, D., Burns, R. and Priebe, C. 2015. Active Community Detection in Massive Graphs. *Proceedings of the 2nd SDM Workshop on Mining Networks and Graphs: A Big Data Analytic Challenge* (2015).

[39] Web Data Commons - Hyperlink Graphs: *http://webdatacommons.org/hyperlinkgraph/*.

[40] Zakrzewska, A. and Bader, D.A. 2013. Measuring the Sensitivity of Graph Metrics to Missing Data. *PPAM Workshop on Power and Energy Aspects of Computation* (Sep. 2013).

About the authors:

Lawrence B. Holder is a Professor in the School of Electrical Engineering and Computer Science at Washington State University. He received his Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign in 1991. Email: holder@wsu.edu.

Maleq Khan is a Research Scientist in the Network Dynamics and Simulation Science Laboratory at Virginia Bioinformatics Institute at Virginia Tech. He received his Ph.D. in Computer Science from Purdue University in 2007. Email: maleq@vbi.vt.edu.

Christine Klymko is a Postdoctoral Researcher in the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory. She received her Ph.D. in Computational Mathematics from Emory University in 2013. Email: klymko1@llnl.gov.

Rajmonda Caceres is a Research Staff Member in the Computing and Analytics Group at the MIT Lincoln Laboratory. She received her Ph.D. in Mathematics and Computer Science from University of Illinois at Chicago in 2012. Email: rcaceres@ll.mit.edu.

Nitesh V. Chawla is a Professor of Computer Science and Engineering at the University of Notre Dame. He received his Ph.D. in Computer Science and Engineering from the University of South Florida in 2002. Email: nchawla@nd.edu.

Tina Eliassi-Rad is an Associate Professor of Computer Science at Rutgers University. She received her Ph.D. in Computer Sciences with a minor in Mathematical Statistics from University of Wisconsin-Madison in 2001. Email: eliassi@cs.rutgers.edu.

David F. Gleich is an Assistant Professor in the Department of Computer Science at Purdue University. He received his Ph.D. in Computational and Mathematical Engineering from Stanford University in 2009. Email: dgleich@purdue.edu.

Ravi Kumar is a Senior Staff Research Scientist at Google, Inc. in Mountain View, CA. He received his Ph.D. in Computer Science from Cornell University in 1998. Email: ravi.k53@gmail.com.

B. Aditya Prakash is an Assistant Professor in the Department of Computer Science at Virginia Tech. He received his Ph.D. in Computer Science from Carnegie Mellon University in 2012. Email: badityap@cs.vt.edu.

Jason Riedy is a Senior Research Scientist in the School of Computational Science and Engineering at the Georgia Institute of Technology. He received his Ph.D. in Computer Science from the University of California Berkeley in 2010. Email: jason.riedy@cc.gatech.edu.

Yinghui Wu is an Assistant Professor in the School of Electrical Engineering and Computer Science at Washington State University. He received his Ph.D. in Computer Science from the University of Edinburgh in 2011. Email: yinghui@eecs.wsu.edu.