Charu C. Aggarwal
IBM T J Watson Research Center

Yorktown Heights, NY 10598

# Network Analysis in the Big Data Age: Mining Graphs and Social Streams

Keynote Talk, ECML/PKDD, 2014

# Introduction

- Large networks contain continuously occurring processes which lead to streams of edge interactions and posts.

- These continuous streams of network and content may be mined for useful insights

- Recent research has recognized the importance of generalizing conventional network mining algorithms to the streaming scenario

- The mining of such streams poses a number of unique challenges.

# Problem Settings

- Streaming graphs: Only structure and no content
  - Receive sequence of edges
  - Receive sequence of small graph objects
- Social streams: Content-centered with underlying structure
  - Receive stream of text-tweets etc.
  - Structure is often associated directly or indirectly with the stream

# Applications

- Real time and actionable insights
- Sudden and anomalous events
- Sudden changes in node properties based on dynamic node interactions (e.g., friend and foe estimations in military settings)
- Real-time summarization of dynamic interaction structures
- Real-time labeling of incoming graph objects

# Examples: Graph Stream

- Sequence of communication pairs in a communication network
- Sequence of source-destination pair activities in an email network
- Object Streams: DBLP and IMDB objects $\Rightarrow$ Slow compared to typical stream
  - Useful for qualitative bench-marking

# Scale

- Graph streams are typically defined over a *massive domain of nodes*.

- Node labels are drawn over universe of distinct identifers.
  - URL addresses in a web graph
  - IP-address in a network application
  - User identifier in a social networking application.

# Challenges

- Consider a graph with $10^8$ nodes: Number of possible source-destination pairs: $10^{16}$
- Most of the edges may not be present $\Rightarrow$ Nevertheless the number of distinct possibilities is very large
- May be difficult to store even the basic summary statistics about the graph stream
- The stream scenario makes the problem even more challenging.
  - The speed of the stream.
  - The entire graph is not available at a given time for analysis.

# Social Streams

- The large volume of social streams makes the problem challenging.

- According to the official Twitter blog, the average number of tweets sent per day was 140 million in 2011, which was around 1620 tweets per second.

- There are about 2 million new friend requests and 3 million messages sent every 20 minutes in Facebook.

# Problems with Social Streams

- Text is mixed with structural data
- Structural data is often only implicit and it needs to be extracted through various means
- The text is noisy and short with many non-standard acronyms

# Recent Advances

- In recent years, clustering, classification, and outlier detection methods have been extended to the graph and social stream scenario:
  - Main problem is that we cannot store the graph structure on disk
  - Complex algorithms need to work with *incomplete* knowledge of the graph structure
  - *Key insight is to adapt synopsis structures for conventional streams to graphs e.g., reservoir sampling*
  - More difficult in graph scenario

# Organization of Talk

- Discuss a snapshot of our recent work in the area
  - Outlier detection of nodes and edges (with Y. Zhao, P. Yu, S. Ma, W. Yu, H. Wang)
  - Clustering and classification of graphs (With Y. Zhao and P. Yu)
  - Dense pattern mining in graphs (with Y. Li, P. Yu R. Jin)
  - Social Stream Processing for event detection (with K. Subbian, J. Srivastava)
  - Social Stream Processing for influence analysis (with K. Subbian, J. Srivastava)

# Important Subproblem: Structural Reservoir Sampling

- Design methods for sampling of network streams
  - Dynamic sample (or *reservoir*) should be usable *at any point* in stream computation
  - Should retain specific properties of the network (e.g., node clustering behavior)
  - Dynamic stream sampling is much more challenging, because it functions with incomplete knowledge about underlying network structure
  - **Challenge:** Fast stream scenario in which entire graph is not available at any time for analysis

# Structural Reservoir Sampling: Aims and Goals

- How to maintain a sample (reservoir) from an edge stream, so as to approximately preserve clustering structure?
  - **Answer:** Edge samples preserve clustering structure in terms of connected components.
  - Multiple edge samples can determine 2-way minimum cuts with high probability (Karger et al) $\Rightarrow$ Multiple reservoirs.
- Maintain specific structural constraints:
  - Size of each partition is constrained.
  - Number of partitions are constrained

# Encoding Structural Constraints

- Many natural structural constraints can be encoded with a *monotonic set function* of the underlying edges in the reservoir.

- **Monotonic Set Function:** A monotonically non-decreasing (non-increasing) set function is a function $f(\cdot)$ on edge sets such that:

  – If $S_1$ is a superset (subset) of $S_2$, then $f(S_1) \geq f(S_2)$

# Using Monotonic Set Function

- Some examples of a monotonic set function:
  - $f(S)$ = Number of connected components of edge set $S$ $\Rightarrow$ Monotonically non-increasing
  - $f(S)$ = Number of nodes in largest connected component of edge set $S \Rightarrow$ Monotonically non-decreasing.
- Use thresholds on $f(S)$ to force specific structural properties such as balancing the sizes of different clusters:
  - *Sampling process must dynamically maintain thresholds on set function*.

# Sampling with Complete Knowledge

- The edges are sorted in random order, and can be added to $S$ only in *sort order priority*.

- *Sort Sample with Stopping Criterion:* A sort sample $S$ from edge set $\mathcal{D}$ with stopping threshold $\alpha$ is defined as follows:

  - We pick the *smallest* subset $S$ from $\mathcal{D}$ *among all subsets which satisfy the sort-order priority*, such that $f(S)$ is at least (at most) $\alpha$.

- Set function constraint on sample translates to structural constraint

# Incomplete Knowledge Scenario

- The previous algorithm only works with knowledge of the full network.

- In the case of a data stream, a random sample or reservoir needs to be maintained *dynamically along with constraints on the set function*.

- Once edges have been dropped from the sample, how does one compare their sort order to the incoming edges in the stream, and correspondingly update the sample?

# Maintaining Sort order in Stream Case

- Use a *fixed random hash function* on the edges.

- Hash function implicitly creates a sort order among the different edges.

  - Hash function serves to provide *landmarks* for incoming edges in comparison to already discarded edges.

  - Hash function *fixes the sort order among edges throughout the stream computation*.

# Set Function Threshold to Hash Function Threshold

- A sort sample $S$ from a set of edges $\mathcal{D}$ with stopping threshold $\alpha$ is equivalent to the following problem:
  - Apply a uniform random hash function $h(\cdot)$ to each edge $(i, j)$ in $\mathcal{D}$.
  - Determine the smallest threshold $q$, such that the set $S$ of edges with hash value $\leq q$ satisfies $f(S) \geq \alpha$.

# Properties of Hash Function

- **Notation:** *Stopping hash threshold* with respect to set function $f$, hash function $h$, data set $\mathcal{D}$ and stopping threshold criterion $\alpha$ denoted by $H(f, h, \mathcal{D}, \alpha)$.

- **Result:** *The stopping hash threshold exhibits a version of set monotonicity with respect to the underlying data set.*

  - Consider two data sets $\mathcal{D}_1$ and $\mathcal{D}_2$, such that $\mathcal{D}_2 \supseteq \mathcal{D}_1$.

  - The stopping hash threshold $H(f, h, \mathcal{D}_2, \alpha)$ is at most equal to $H(f, h, \mathcal{D}_1, \alpha)$.

# Properties of Hash Threshold

- *The stopping hash threshold is monotonically non-increasing over the life of the data stream.*

- **Critical Result:** *Edges which are not relevant now will never be relevant for sampling over the future life of the data stream.*

- The current sample is the only set we need for any future decisions about reservoir sample maintenance.

# Structural Reservoir Sampling Algorithm

- Simple algorithm in order to maintain the reservoir dynamically.

- Dynamically maintain the *current hash threshold* for making admission-control decisions.

- An incoming edge is added to the reservoir, if its hash function value is less than the current threshold value.

  – May need to remove edges in ensure that current set is the smallest sort-ordered set to satisfy stopping criterion.

  – May need to adjust hash threshold.

# Structural Reservoir Sampling Algorithm

- Process the edges in the reservoir *in decreasing order of the hash function value*.
  - Remove edges till satisfaction of stopping constraint.
- The hash threshold is reduced to the largest hash function value of the *remaining edges in the reservoir* after removal.
- The removal of edges may result in a reduction of the hash threshold in each iteration.
- Continue sampling with modified hash threshold

# Observations

- Multiple reservoirs are maintained for greater robustness
- Union of reservoirs provides a *sparsified* network which preserves the clustering structure of the underlying stream
- Can be used for offline analysis for a variety of network-analysis problems
  - Specific Application: Abnormal relationship discovery in dynamic networks

# Abnormal Relationship Detection in Networks

- Significant deviations from the "normal" structural patterns

- *Unusual relationships correspond to edges between graph regions that rarely occur together*.
  - Unusual Social Activity
  - Unusual Relationship Evolution
  - Spam

# Problem Scenario

- We have an *incoming stream of graph objects*, denoted by $G_1 \ldots G_t \ldots$.

- Each graph $G_i$ is a small subset of a large base domain $N$ of nodes.
  - Graph objects could be information network objects.
  - Entity-relation graphs for a stream of incoming complex objects
  - Local or temporal patterns of activity in a social network

# Algorithm Components

- Using node partitions for the purpose of outlier modeling.
  - Construct *likelihood fit model* for edges across different partitions using the statistics of the edges in the different reservoirs.
- Use of outlier model to make abnormality prediction about incoming object.
  - **Intuition:** Bridging edges across dense regions are outliers
  - Likelihood fit for a particular information network object is a function of that of constituent edges

# Likelihood Fit of Graph Object

- The likelihood fit for a graph object $G$ is defined as a function of the likelihood fits of the constituent edges in $G$.

- Estimate each edge likelihood using the statistical distribution of edges across different partitions in the sample.

- **Graph Object Likelihood Fit:** The likelihood fit $\mathcal{GF}(G, \mathcal{C}_1 \ldots \mathcal{C}_r)$ for a graph object $G$ is the product of the (composite) likelihood fits of the edges in $G$.

$$\mathcal{GF}(G, \mathcal{C}_1 \ldots \mathcal{C}_r) = \left[ \prod_{(i,j) \in G} \mathcal{MF}(i, j, \mathcal{C}_1 \ldots \mathcal{C}_r) \right]$$

# Experimental Results

- Tested on two real and one synthetic data set.
  - IMDB and DBLP data sets
- For real data sets, case studies are presented for effectiveness
- For synthetic data sets, precision and recall can be presented.

# Examples of Anomalous Bibliographic Objects (DBLP)

- Yihong Gong, Guido Proietti, Christos Faloutsos, *Image Indexing and Retrieval Based on Human Perceptual Color Clustering*, CVPR 1998: 578-585.
  - Computer Vision and Multimedia Dominated : *Yihong Gong*
  - Database and Data Mining Dominated : *Christos Faloutsos*
- The two kinds of nodes were assigned to different partitions by the structural reservoir sampling algorithm

# Examples of Anomalous Bibliographic Objects (DBLP)

- Natasha Alechina, Mehdi Dastani, Brian Logan, John-Jules Ch Meyer, *A Logic of Agent Programs*, AAAI 2007: 795-800.

- The co-authorship behavior of these cohorts was defined by geographical proximity.

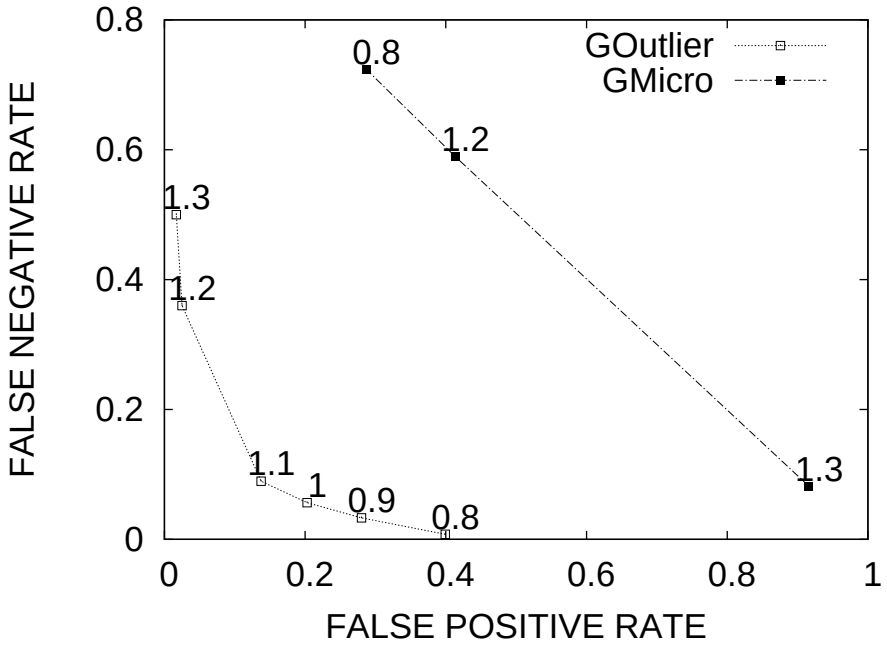
- The first partition includes a group of researchers in the United Kingdom, while the second partition is composed of researchers in the Netherlands.

- The different groups were naturally assigned to different clusters.

# Examples of Anomalous Movie Objects

- *Movie Title:* Cradle 2 the Grave (2003)
- This movie was directed by Andrzej Bartkowiak, and the actors include Jet Li, DMX (I), etc.
- Non-chinese director which contains an international cast along with many chinese actors.
- *Movie Title:* Memoirs of a Geisha, 2005: Contains participants from Chinese, Japanese and American backgrounds

# Effectiveness Results




- Synthetic data set

# Anomalous Hotpot Discovery

- The previous approach discovers relationship outliers in graph streams
- A second approach is to discover node hotspots
- Node hotspots are anomalous points in the network which show one of the following two types of changes:
  - Sudden change in magnitude of activity
  - Sudden change in patterns of activity

# Broad Approach

- Use the eigenvectors of the adjacency matrix in a node's locality to determine significant directions:
  - Capture changes in magnitude of eigenvectors (eigenvalues)
  - Capture changes in angle between eigenvectors

# Applications

- Discovering sudden intrusion attacks at a specific node
- Sudden changes in collaboration patterns of a faculty member (e.g., increased output, different collaborators)
- Sudden changes in friendship patterns at a social networking site.
- Refer to ICDM 2011 paper (with Y. Wu, S, Ma, H. Wang) for detailed results and experimental examples.

# Graph Clustering Scenarios

- Two main cases for graph stream clustering
  - Node clustering in the context of an edge stream
  - Clustering a stream of small graphs drawn on a massive base domain
- We focus on the second case in which the graphs are clustered as objects

# Problem Definition

- Denote node set by $\mathcal{N}$ (very large)
- The individual graphs in the stream are denoted by $G_1 \ldots G_n \ldots$.
- The stream comprises a *sequence* of graph objects:
  - Incoming stream of entity-relation graphs
  - Web click graphs
  - Networking applications for tracking patterns of activity

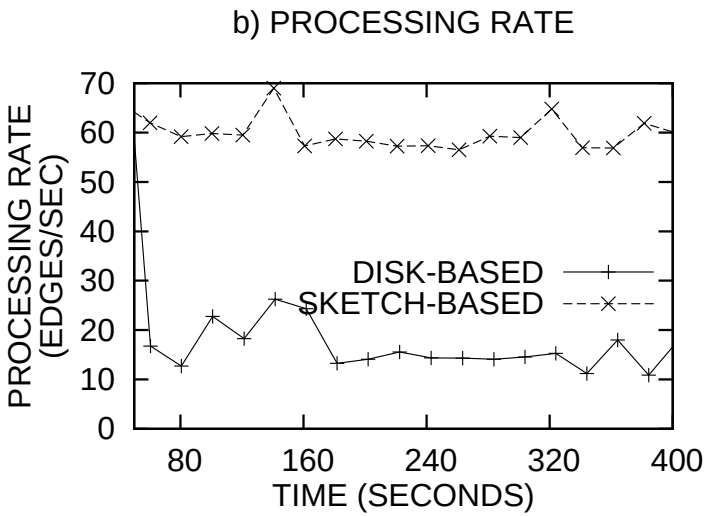
# Related Challenges

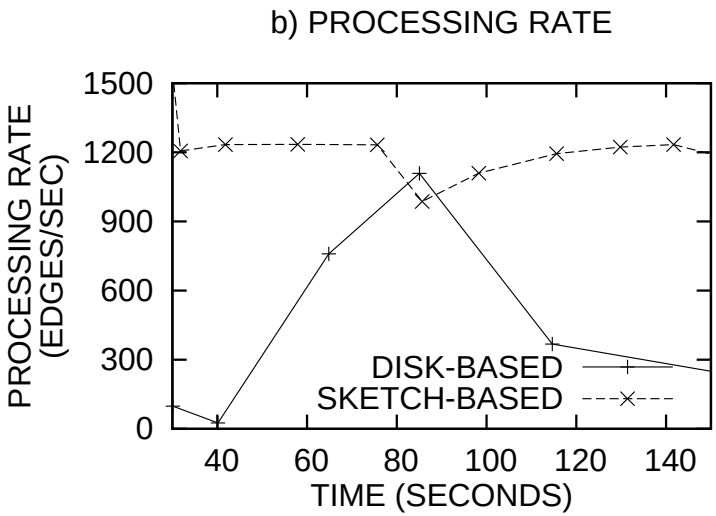
- Structural analysis is difficult in streaming scenario
- The number of possible edges scales up quadratically with the number of nodes.
- The individual graphs may contain only a small fraction of the nodes $\Rightarrow$ Difficult to compare between graphs
- The number of edges which correspond to different clusters grows over time $\Rightarrow$ Difficulty in statistics maintenance

# Broad Approach

- Extend micro-clustering approach to graph data
- Define a micro-cluster representation which can be used in order to track the summary statistics of the graphs in a given micro-cluster
- Key challenge $\Rightarrow$ Micro-cluster statistics can become unwieldy because of the large number of distinct edges
- **Solution:** Use sketch-compressed micro-clusters
- Provide theoretical bounds on accuracy (SDM 2010).

# Effectiveness Results




- Effect of sketches on efficiency in DBLP and IBM sensor data sets

# Dense Pattern Mining

- Determining highly co-occurring sets of nodes, which are also highly interconnected
- Use of well known min-hash summarization technique.
- Size of summary is independent of the length of the data stream.
- Provide theoretical bounds on the accuracy of this approach.
- Related work on min-hash technique uses the approach to find dense patterns in a single large graph (Gibson et al, 2005)

# Assumptions

- The stream $\mathcal{S}$ is defined as the sequence $G_1 \ldots G_r \ldots$, where each graph $G_i$ is a set of edges.

- An important assumption here is that the graph is drawn over a *massive domain of nodes*.

- The individual edge set $G_i$ contains only a small fraction of the underlying nodes.

- The sparsity property is important in enabling a meaningful problem formulation.

# Desired Properties of Mined Patterns

- **Node Co-occurrence:** We would like to determine nodes which co-occur frequently in the network.
  - Node co-occurrence is defined in terms of *relative presence*, so that irrelevant patterns are pruned automatically.
- **Edge Density:** Within a given node set, we would like these interactions to be as dense as possible.
  - Want to determine node sets in which a *large fraction* of the possible edges are populated.

# Node Affinity

- The node co-occurrence over a set of nodes $P$ is defined by a parameter called *node affinity*.

- **Definition:** *Let $f_{\cap}(P)$ be the fraction of graphs in $G_1 \ldots G_n$ in which* **all** *nodes of $P$ occur. Let $f_{\cup}(P)$ be the fraction of graphs in which* **at least one** *of the nodes of $P$ occur. Then, the node affinity $A(P)$ of pattern $P$ is denoted by $f_{\cap}(P)/f_{\cup}(P)$.*

- The definition of node-affinity is focussed on relative presence of nodes rather than the raw frequency.

- Represents Jaccard Coefficient defined on the node set.

# Edge Density

- We wish to determine sets of nodes between which the edges are densely populated.

- We define the edge density $D(P)$ of the node set $P$ as follows:

- **Definition:** *Let $G_i$ be a graph which contains all nodes of $P$. Let $h(P, G_i)$ denote the fraction of the $|P| \cdot (|P|-1)/2$ possible edges defined on $P$ which are included in the edge set $E_i$ of $G_i$. Then, the value of $D(P)$ is defined as the average of $h(P, G_i)$* **only over** *those graphs which contain* **all** *nodes in $P$.*

# Significant Pattern Mining Problem

- We define the significant pattern mining problem with the use of two threshold parameters $(\theta, \gamma)$ on node correlation and edge density:

- **Definition:** *A set of nodes $P$ is said to be be $(\theta, \gamma)$-significant, if it satisfies the following two* **node-affinity** *and* **edge-density** *constraints:*

  - *(a) The node-affinity $A(P)$ is at least $\theta$.*

  - (b) *The edge-density $D(P)$ is at least $\gamma$.*

# Observations

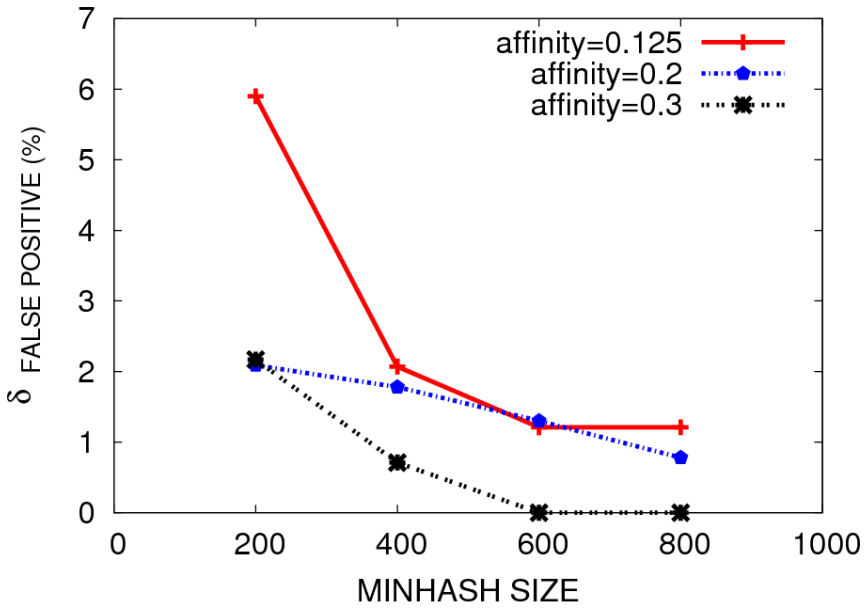
- Computational efficiency is an important concern for the stream scenario.

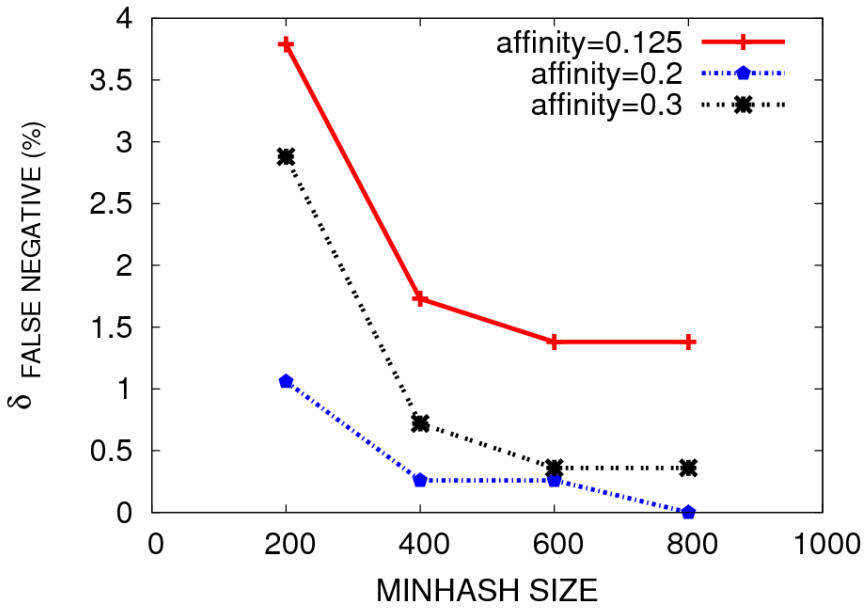
- We use a probabilistic min-hash approach which exploits the sparsity property of the underlying graphs.

# Two Phase Approach

- The dense pattern mining algorithm requires two phases.
- The first phase determines the correlated node patterns with min-hash summarization.
  - These are defined as patterns for which the affinity is greater than the user-specified threshold $\theta$.
- The second phase determines the subset of those node patterns which satisfy the edge-density constraint with user-specified threshold $\gamma$.
- Two passes can be consolidated into a single pass (Consolidation method discussed in VLDB 2010 paper).

# Effectiveness Results (Sensor Data Set)




- Sensor data stream, $\delta$-error variation with Min-Hash Sample Size (different affinities, density= 0.05, $\delta = 0.05$)

# Streaming Graph Classification

- Denote node set by $N$ (very large)
- The individual graphs in the stream are denoted by $G_1 \ldots G_n \ldots$.
- Each graph $G_i$ is associated with the class label $C_i$ which is drawn from $\{1 \ldots m\}$.
- The edges of each graph $G_i$ may not be neatly received at a given moment in time $\Rightarrow$ May appear *out of order* in the data stream.
  - The edges are received as $< EdgeId, GraphId >$

# Classification Modeling Approach

- Design a rule-based classifier which relates subgraph patterns to classes
  - Left hand side contains the subgraph and right hand side contains the class-label
- Rules are maintained *indirectly* in the form of a continuously updatable and stream-friendly data structure.
- Use two criteria to mine subgraphs for rule-generation:
  - **Relative Presence:** Determine subgraphs for which relative presence of co-occurring edges (as a group) is high.
  - **Class Distribution:** Determine subgraphs which are discriminative towards a particular class.

# Modeling Relative Presence of Subgraphs

- Determine subgraphs which have significant presence in terms of the *relative frequency* of its constituent edges.

- $f_{\cap}(P) \Rightarrow$ Fraction of graphs in $G_1 \ldots G_n$ in which **all** edges of subgraph $P$ are present.

- $f_{\cup}(P) \Rightarrow$ Fraction of graphs in which **at least one or more** of the edges of subgraph $P$ are present.

- The **edge coherence** $C(P)$ of the subgraph $P$ is denoted by $f_{\cap}(P)/f_{\cup}(P)$.

# Observations

- The definition of edge coherence is focussed on *relative presence* of subgraph patterns rather than the absolute presence.
  - This ensures that only significant patterns are found.
  - Ensures that large numbers of irrelevant patterns with high frequency but low significance are not considered.
- Computationally more challenging than direct support-based computation.

# Class Confidence

- Among all graphs containing subgraph $P$, determine the fraction belonging to class label $r$
  - Also referred to as **confidence** of pattern $P$ with respect to the class $r$.
- The dominant class confidence $DI(P)$ or subgraph $P$ is defined as the maximum class confidence across all the different classes $\{1 \ldots m\}$.
- A significantly large value of $DI(P)$ for a particular test instance indicates that the pattern $P$ is very relevant to classification.

# Formal Definition (Significant Patterns)

- A subgraph $P$ is said to be $(\alpha, \theta)$-significant, if it satisfies the following two *edge-coherence* and *class discrimination* constraints:
  - The edge-coherence $C(P)$ of subgraph $P$ is at least $\alpha$.

$$C(P) \geq \alpha \tag{1}$$

  - The dominant class confidence $DI(P)$ is at least $\theta$.

$$DI(P) \geq \theta \tag{2}$$

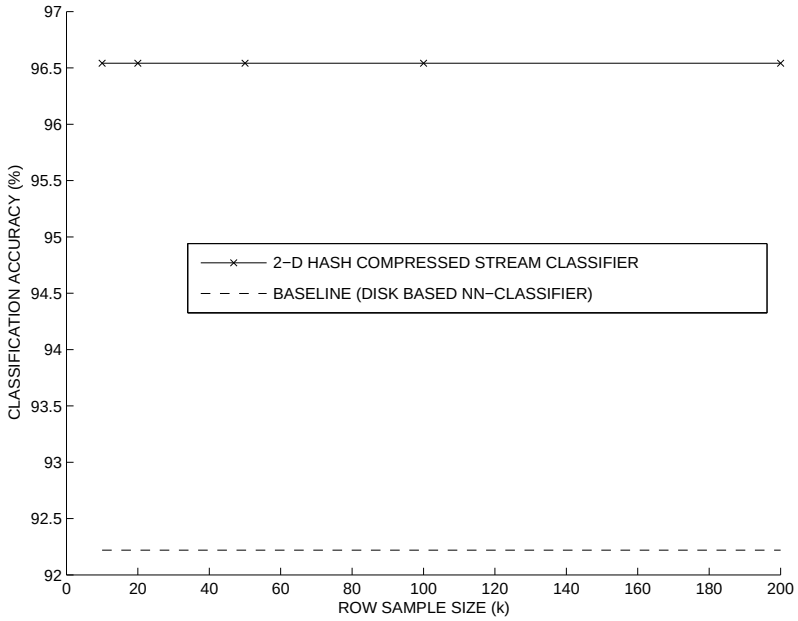
# Broad Approach

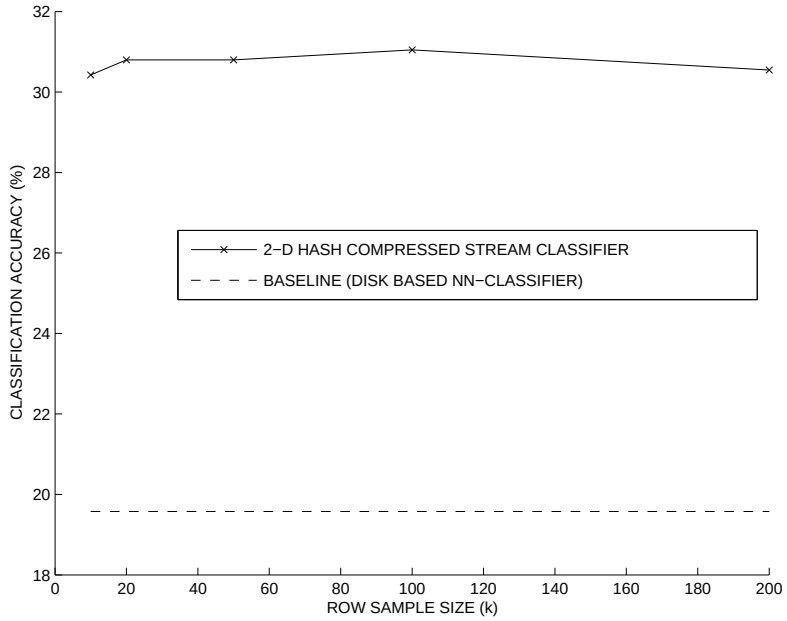
- **Aim:** Design a continuously updatable synopsis data structure, which can be efficiently mined for the most discriminative subgraphs.

- Small size synopsis with min-hash approach:
  - Can be dynamically maintained and applied in online fashion at any point during stream progression.
  - The *structural synopsis* maintains sufficient information which is necessary for classification purposes.

- Use min-hash synopsis to perform real-time classification.

# Classification Accuracy Results




- Classification Accuracy with increasing min-hash size for (a) DBLP data set (b) Sensor data set

# Social Streams

- Massive amounts of user activity in online social networks
- Can be crowd-sourced to learn useful insights such as relevant events
- A significant challenge is that the data is often not reliable

# Event Detection in Social Streams

- Social media often reports significant events earlier than traditional channels.
- Bin Laden's raid in Abbottabad was reported live by a Twitter user
- Hard to detect such events because they are mixed with noise and rumors $\Rightarrow$ Did not start a cascade of tweets
- Story broke on Twitter anyway before official announcement $\Rightarrow$ Unofficial Keith Urbahn tweet
- When events lead to sudden *aggregate* changes, they are easier to detect.

# Clustering Approach

- Since clustering can accurately characterize the aggregate trends in the stream, it can discover key events by monitoring the changes in aggregate patterns

- Supervised and unsupervised:
  - Unsupervised methods discover a sudden change
  - Supervised methods discover a sudden change with a similar signature as a supplied sample of training events

- Both scenarios can be captured by variations of clustering methods

# Clustering Challenges

- Fine-grained clustering is required to meaningfully detect events.
- More challenging than streaming clustering, because structure is also used in the clustering process
- Need to incorporate synopsis structures into the clustering process.

# Overview of Approach

- Continuously maintain micro-clusters as the stream arrives
- Micro-clustering concept is modified to store a combination of content and structure
- Combine micro-clustering with sketch concept to store the clusters in a more compact way

# Unsupervised Evolution Events

- An evolution event over horizon $H$ at current time $t_c$ is said to have occurred at threshold $\alpha$ for cluster $\mathcal{C}_i$, if the ratio of the relative presence of points in cluster $\mathcal{C}_i$ over the horizon $(t_c - H, t_c)$ to that before time $t_c - H$ is greater than the threshold $\alpha$. In other words, we have:

$$\frac{F(t_c - H, t_c, \mathcal{C}_i)}{F(t(\mathcal{C}_i), t_c - H, \mathcal{C}_i)} \geq \alpha \tag{3}$$

- The notation $t(\mathcal{C}_i)$ denotes the time of the creation of cluster $\mathcal{C}_i$.

# Supervised Evolution Events (Training Signatures)

- The *relative distribution of event-specific stream objects* to clusters is used as a signature which is specific to the event.

- The event signature of a social stream is a $k$-dimensional vector $V(\mathcal{E})$ containing the (average) relative distribution of event-specific stream objects to clusters.

- The $i$th component of $V(\mathcal{E})$ is the fraction of event-specific (training) stream objects which are assigned cluster $i$.

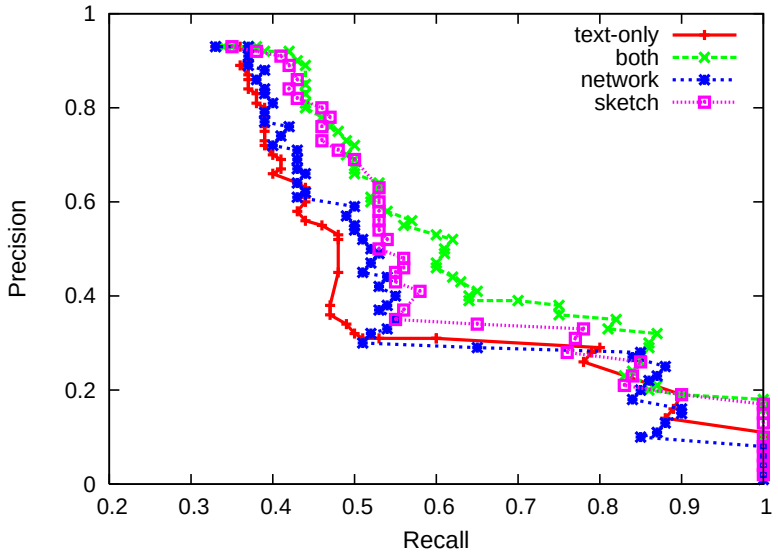
# Testing Process

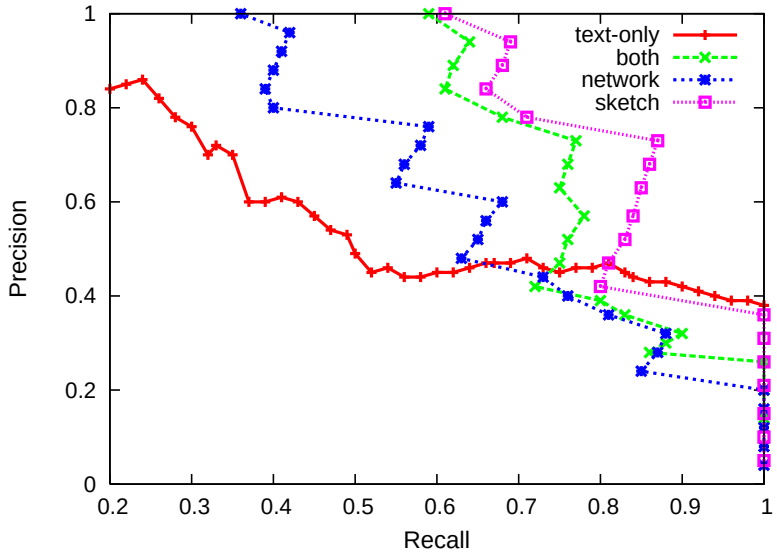
- *Horizon-specific Signature:* The horizon signature over the last time period $(t_c - H, t_c)$ is a $k$-dimensional vector containing the relative distribution of social stream objects to clusters which have arrived in the period $(t_c - H, t_c)$.

- Need to match with training signatures

- In order to perform the supervised event detection, we simply compute the dot product of the horizon signature with the known event signature, and output an alarm level.

# Sample Event Detection Results




- (a) Japan nuclear crisis (b) Uganda protests

- See Aggarwal and Subbian (SDM 2012) for more details

# Influence Analysis in Social Streams

- Most influence analysis models are static in which edge interaction probabilities are modeled.
- However, the influence probabilities are dynamic and change with time.
- Influence probabilities may also be sensitive to the specific keyword or topic being analyzed.
- The edge interaction probabilities are never given: $\Rightarrow$ The only visible data is the social stream

# Streaming approach for influence analysis

- Discover cascades of actor-patterns that propagate the same keyword
- Use real-time sequential pattern mining to discover the relevant patterns
- Flow patterns can be sensitive to keyword content and topics
- Propose topic-sensitive influencers in real time from flow patterns
- See Subbian, Aggarwal and Srivastava (CIKM 2013) for details and experiments

# Relevant Survey Papers and Books

- M. Spiliopoulou. Evolution in Social Networks: A Survey, *Social Network Data Analytics*, Springer, 2011.

- C. Aggarwal and K. Subbian. Evolutionary Network Analysis: A Survey, *ACM Computing Surveys*, May 2014.

- C. Aggarwal. Mining Text and Social Streams: A Review, *ACM SIGKDD Explorations*, December 2013.

- M. Gupta, J. Gao, C. Aggarwal, and J. Han. Outlier Detection for Temporal Data: A Survey, *IEEE TKDE Journal*, 2014. Expanded as a 100-page book (synthesis lecture) by *Morgan and Claypool*, 2014.

- Temporal and Graph Chapters in the book: C. Aggarwal, Outlier Analysis, *Springer*, 2013.

# Conclusions

- Graph and social streams are an important and emerging area of research.
- Challenging because of the difficulty in storing structural aspects in a summarized way
- Current research shows how to use specialized synopses for specific methods
- Most sampling methods are tailored to specific problems
- Significant scope for improving the breadth of applicability of the ideas