Part II: Event detection in dynamic graphs

L. Akoglu & C. Faloutsos

Anomaly detection in graph data (ICDM'12)
Part II: Outline

Overview: Events in point sequences
- Change detection in time series
- Learning under concept drift

Events in graph sequences
- Change by graph distance
- Change by graph connectivity
Event detection

- Anomaly detection in time series of multi-dimensional data points
  - Exponentially Weighted Moving Average
  - CUmulative SUM Statistics
  - Regression-based
  - Box-Jenkins models eg. ARMA, ARIMA
  - Wavelets
  - Hidden Markov Models
  - Model-based hypothesis testing
  - …

- This tutorial: time series of graphs
Part II: References (data series)

Part II: Outline

Overview: Events in point sequences
- Change detection in time series
- Learning under concept drift

Events in graph sequences
- Change by graph distance
  - feature-based
  - structure-based
- Change by graph connectivity
Events in time-evolving graphs

Problem: Given a sequence of graphs,

Q1. change detection: find time points at which graph changes significantly

Q2. attribution: find (top k) nodes / edges / regions that change the most
Events in time-evolving graphs

- **Main framework**
  - Compute graph *similarity/distance scores*
  - Find *unusual occurrences* in time series

- *Note: scalability is a desired property*
Taxonomy

Graph Anomaly Detection

Static graphs

- Plain
  - Feature based
    - Structural features
    - Recursive features
  - Community based
- Attributed
  - Structure based
    - Substructures
    - Subgraphs
  - Community based

Dynamic graphs

- Plain
  - Distance based
    - Feature-distance
    - Structure distance
  - Structure based
    - “phase transition”

Graph algorithms

- Learning models
  - RMNs
  - PRMs
  - RDNs
  - MLNs
- Inference
  - Iterative classification
  - Belief propagation
  - Relational netw. classification

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM’12)
Graph distance – 10 metrics

(1) Weight distance

\[
d(G, H) = \left| E_G \cup E_H \right|^{-1} \sum_{u,v \in V} \frac{|w^G_E(u, v) - w^H_E(u, v)|}{\max\{w^G_E(u, v), w^H_E(u, v)\}}
\]

(2) Maximum Common Subgraph (MCS) Weight distance

\[
d(G, H) = \left| E_G \cap E_H \right|^{-1} \sum_{u,v \in V} \frac{|w^G_E(u, v) - w^H_E(u, v)|}{\max\{w^G_E(u, v), w^H_E(u, v)\}}
\]

(3) MCS Edge distance

\[
d(G, H) = 1 - \frac{|\text{mcs}(E_G, E_H)|}{\max\{|E_G|, |E_H|\}}
\]
Graph distance – 10 metrics

- **(4) MCS Node distance**

\[ d(G, H) = 1 - \frac{|\text{mcs}(V_G, V_H)|}{\max\{|V_G|, |V_H|\}} \]

- **(5) Graph Edit distance**  
  Gao et al. ‘10 (survey)

\[ d(G, H) = |V_G| + |V_H| - 2|V_G \cap V_H| + |E_G| + |E_H| - 2|E_G \cap E_H| \]

- Total cost of sequence of edit operations, to make two graphs isomorphic (costs may vary)
- Unique labeling of nodes reduces computation
- otherwise an NP-complete problem
- Alternatives for **weighted** graphs
Graph distance – 10 metrics

(5.5) Weighted Graph Edit distance

Kapsabelis et al. ’07

\[ d_2(G, H) = c \left[ |V_G| + |V_H| - 2|V_G \cap V_H| \right] + \sum_{e \in E_G \cap E_H} |\beta_G(e) - \beta_H(e)| \]

+ \sum_{e \in E_G \setminus (E_G \cap E_H)} \beta_G(e) + \sum_{e \in E_H \setminus (E_G \cap E_H)} \beta_H(e)

Non-linear cost functions

\[ d_3(G, H) = c \left[ |V_G| + |V_H| - 2|V_G \cap V_H| \right] + \sum_{e \in E_G \cap E_H} \frac{|(\beta_G(e) + \epsilon) - (\beta_H(e) + \epsilon)|^2}{(|\min(\beta_G(e), \beta_H(e)) + \epsilon|^2)} \]

+ \sum_{e \in E_G \setminus (E_G \cap E_H)} (\beta_G(e) + \epsilon)^2 + \sum_{e \in E_H \setminus (E_G \cap E_H)} (\beta_H(e) + \epsilon)^2

\[ \epsilon = 1 \]
Graph distance – 10 metrics

(6) Median Graph distance

- Median graph of sequence \((G_{n-L+1}, \ldots, G_n)\)

\[
\tilde{G}_n = \arg\min_{G \in S} \sum_{i=n-L+1}^{n} d(G, G_i)
\]

- \(d(\tilde{G}_n, G_{n+1})\) for each graph \(G_{n+1}\) in sequence
- free to choose any distance function \(d\)

(7) Modality distance

\[
d(G, H) = \| \pi(G) - \pi(H) \|
\]

\[A\pi = \rho\pi, \quad \pi > 0\]

Perron vector

Dickinson et al. ‘04

Kraetzl et al. ‘06
Graph distance – 10 metrics

(8) Diameter distance

\[ d(G, H) = \left| \sum_{v \in V_H} \max d(H, v) - \sum_{v \in V_G} \max d(G, v) \right| \]

(9) Entropy distance

\[ d(G, H) = - \sum_{e \in E_H} (\tilde{w}_e^H - \ln \tilde{w}_e^H) + \sum_{e \in E_G} (\tilde{w}_e^G - \ln \tilde{w}_e^G) \]

\[ \tilde{w}_e^* = \frac{w_e^*}{\sum_{e \in E^*} w_e^*} \]

(10) Spectral distance

\[ d(G, H) = \sqrt{\min \left\{ \sum_{i=1}^{k} (\lambda_i - \mu_i)^2, \sum_{i=1}^{k} \lambda_i^2, \sum_{i=1}^{k} \mu_i^2 \right\}} \]

Gaston et al. ’06

Largest pos. eigenvalues of Laplacian

L. Akoglu & C. Faloutsos

Anomaly detection in graph data (ICDM’12)
# Graph distance – 10 metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Vertices used?</th>
<th>Edges used?</th>
<th>Vertex weights used?</th>
<th>Edge weights used?</th>
<th>Range</th>
<th>Value if graphs identical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>[0,1]</td>
<td>0</td>
</tr>
<tr>
<td>MCS Weight</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>[0,1]</td>
<td>0</td>
</tr>
<tr>
<td>MCS Edge</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>[0,1]</td>
<td>0</td>
</tr>
<tr>
<td>MCS Vertex</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>[0,1]</td>
<td>0</td>
</tr>
<tr>
<td>Graph Edit</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>[0,∞)</td>
<td>0</td>
</tr>
<tr>
<td>Median Edit</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>[0,∞)</td>
<td>0</td>
</tr>
<tr>
<td>Modality</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>[0,1]</td>
<td>0</td>
</tr>
<tr>
<td>Diameter</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>[0,∞)</td>
<td>0</td>
</tr>
<tr>
<td>Entropy</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>(-1,1)</td>
<td>0</td>
</tr>
<tr>
<td>Spectral</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>[0,1]</td>
<td>0</td>
</tr>
</tbody>
</table>
Graph distance to time series

- Each graph as a feature vector
  - graph distance metrics
- Time series of graph distances per feature
- $\text{ARMA}(p, q)$ model for each time series
  \[ X_t = \alpha_1 X_{t-1} + \cdots + \alpha_p X_{t-p} + \epsilon_t + \beta_1 \epsilon_{t-1} + \cdots + \beta_q \epsilon_{t-q} \]
  - assumes stationary series, due to construction
- Anomalous time points: where residuals exceed a threshold

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM'12)
Graph distance to time series

- Minimum mean squared error $S = X_1, X_2, \ldots, X_M$

$$\text{MSE}(m) = \sum_{i=1}^{m} (X_i - \bar{X}_L)^2 + \sum_{i=m+1}^{M} (X_i - \bar{X}_R)^2$$

- change point: $m$ with minimum MSE($m$)
- randomized bootstrapping for confidence

- **CUmulative SUMmation**

$$C = (s_0, s_1, \ldots, s_M)$$

- $s_0 = 0$
- $s_k = s_{k-1} + X_k - \bar{X}$

- bootstrap $\Delta C' = \max_{i=1,\ldots,M} C' - \min_{i=1,\ldots,M} C'$

**Note:** single feature to represent whole graph
Scan statistics on graphs

- For each "scan region"
  - compute locality statistic
- (11) Scan statistic = max of locality statistics

For graph data
- k-th order neighborhood
- scan region: induced k-th order subgraph
- locality stat.: e.g., #edges, density, domination #, …
- scale (k)-specific scan stat.

$$M_k(D) = \max_{v \in V(D)} \Psi_k(v)$$

Priebe et al. ’05

Anomaly detection in graph data (ICDM’12)
Scan statistics on graphs

- Vertex-dependent normalized locality statistic

\[ \widetilde{\Psi}_{k,t}(v) = \left( \Psi_{k,t}(v) - \widehat{\mu}_{k,t}(v) \right) / \max(\widehat{\sigma}_{k,t}(v), 1) \]

mean and std in (t–tau) window

\[ \widetilde{M}_{k,t} = \max_v \widetilde{\Psi}_{k,t}(v) \]
Graph similarity

*Note: sensitivity is a desired property
  
  - e.g. “high/low-quality” pages in Web graph
  - quality/importance: e.g., pagerank

(12) Vertex ranking

\[
sim_{VR}(G, G') = 1 - \frac{2 \sum_{v \in V \cup V'} w_v \times (\pi_v - \pi'_v)^2}{D}
\]

- Rank: v in both G and G’ → average
  - v in only V → \( \pi'_v = |V'| + 1 \)
  - v in only V’ → \( \pi_v = |V| + 1 \)
Graph similarity

(13) Sequence similarity
- Depth-first-like sequencing with high-quality first
  Repeat
  - pick unvisited node with highest quality
  - visit highest quality unvisited neighbor, if any

- Apply shingling
  - all $k$-length subsequences, i.e. shingles $S(T)$
  - $\text{sim}_{SS}(G, G') = \frac{S(T) \cap S(T')}{{S(T)} \cup S(T')}$
  - $T = \langle F, E, D, A, C, H, B, G \rangle$

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM'12)
Graph similarity

(14) Vector similarity

- Compare **weighted** edge vectors
- relative importance of an edge:

\[
\gamma(u, v) = \frac{q_u \times \#\text{outlinks}(u, v)}{\sum_{\{v' : (u, v') \in E\}} \#\text{outlinks}(u, v')}
\]

- Similarity over union of edges in G and G’

\[
sim_{VS}(G, G') = 1 - \frac{\sum_{(u,v) \in E \cup E'} |\gamma(u,v) - \gamma'(u,v)|}{\max(\gamma(u,v), \gamma'(u,v)) |E \cup E'|}
\]

**note:** for edges not in G’ \( \gamma'(u, v) = 0 \), and vice versa
Graph similarity

(15) Signature similarity

- Transfer graph G to a set L of weighted features

\[ L = \left\{ (t_i, w_i) \right\} \]

e.g. \( L(G) = \left\{ (C, 0.51), (CF, 0.51), (F, 1.29), (FC, 1.29 \times 0.5), (FH, 1.29 \times 0.5), (H, 0.51), (HF, 0.51) \right\} \).

- Construct b-bit signature for G
  - For each \( t_i \)
    - randomly choose b entries from \( \{-w_i, +w_i\} \)
  - Sum all b-dimensional vectors into h
  - Set ‘+’ entries to 1 and ‘-’s to 0

\[ \text{sim}(L, L') = 1 - \frac{\text{Hamming}(h, h')}{b} \]
**Graph similarity**

- **Missing connected**
- **Missing random**
- **Connectivity change**

**Vertex ranking**
- Very good
- Bad
- Bad

**Sequence similarity**
- Good
- Bad
- Very good

**Vector similarity**
- Very good
- Good
- Good

**Signature similarity**
- Very good
- Very good
- Very good
Part II: Outline

- Overview: Events in **point sequences**
  - Change detection in time series
  - Learning under concept drift

- Events in **graph sequences**
  - Change by graph **distance**
    - feature-based
    - structure-based
  - Change by graph **connectivity**
    - phase transition

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM'12)
Eigen-space-based events

- **Given** a time-evolving graph
- **Identify** faulty vertices

**Challenges**
- Large number of nodes, impractical to monitor each
- Edge weights are highly dynamic
- Anomaly defined collectively (different than “others”)

**Event:** a “phase transition” of the graph (in overall relation between the edge weights)
“Summary feature” extraction

- Definition of “activity” vector

\[ u(t) \equiv \arg \max_{\tilde{u}} \left\{ \tilde{u}^T D(t) \tilde{u} \right\} \text{ subject to } \tilde{u}^T \tilde{u} = 1 \]

activity vector at \( t \)  
adjacency matrix at \( t \)  
(symmetric, non-negative)

- The above equation can be reduced to

\[ D(t) \tilde{u} = \lambda \tilde{u} \text{ subject to } \tilde{u}^T \tilde{u} = 1 \]

→ The principal eigenvector gives the summary of node “activity”
Activity feature

Why “activity”? (intuition)

- If \( D_{12} \) is large, then \( u_1 \) and \( u_2 \) should be large because of argmax (note: \( D \) is a positive matrix).
- So, if \( s_1 \) actively links to other nodes at \( t \), then the “activity” of \( s_1 \) should be large.

- Also interpreted as “stationary state”: probability that a node is holding the “control token”
Anomaly detection

- Problem reduced from a sequence of graphs to a sequence of (activity) vectors

adjacency matrix

activity vector

principal eigenvector

summary vector

$z(t) \equiv 1 - r(t-1)^T u(t)$

L. Akoglu & C. Faloutsos

Anomaly detection in graph data (ICDM'12)
Experiment

- Time evolution of activity scores effectively visualizes malfunction

- Anomaly measure and online thresholding dynamically capture activity change

- Nodes changing most can be attributed
Feature/Eigen-space-based events

Nodes

Features (egonet)

Time

Features: inweight

Past pattern

Eigen-behavior at t

Eigen-behaviors
Change point detection

F: out-degree

F: reciprocal degree

Christian New Year

“back to work”

Hindi New Year

Event score $Z$ over time
Change attribution

Time series of top 5 nodes with highest ratio index

26 DEC

Time (days)

#SMS received

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM’12)
Community-based events

- **Main idea:** monitor community structure and alert event when it changes.
Community-based events

- Many graph clustering/partitioning algorithms
  - METIS, Karypis et al. ’95
  - Spectral Clustering, Shi & Malik ’00, Ng et al.’02
  - Girvan-Newman ’03
  - Co-clustering, Dhillon et al. ’03, Chakrabarti ’04
  - ...

- Challenge
  - distance measure between clusterings
Community detection

- Clustering problem as compression problem

1. Similar nodes are grouped together
2. As few groups as necessary

A few, homogeneous blocks

Good Clustering

implies

low code cost (blocks)

+ low description cost (blocks’ model)

Good Compression

L. Akoglu & C. Faloutsos Anomaly detection in graph data (ICDM’12)
Community-based events

- **Goal:** partition the graph sequence into segments $G^{(1)}$ and $G^{(2)}$, where each segment exhibits a (different) clustering.

- **Q:** when does a new segment (=event) emerge?
Change detection

Guiding principle: encoding cost benefit

1: append

G(S) G(t)

union G(S'), with cost c(S')

c(S') - c(S) < c(t)

event?

2: split (time)

segment G(S), with cost c(S)

new graph G(t), with cost c(t)
Community changes in Enron

- 34K email addresses
- 165 weeks
- ~2M emails

Key change-points correspond to key events

Bit-cost can quantify event “intensity”
Reconstruction-based events

General Framework

- Network forensics
  - Sparsification ➔ load shedding
  - Matrix decomposition ➔ summarization
  - Error Measure ➔ anomaly detection

L. Akoglu & C. Faloutsos
Anomaly detection in graph data (ICDM'12)
Sun+ICDM'07
modified with permission
Matrix decomposition

- **Goal:** summarize a given graph
  - decompose adjacency matrix into smaller components
  
  1. Singular Value Decomposition (SVD)
  - PCA, LSI, ...
  
  2. CUR decomposition
  - Drineas et al. ‘05
  
  3. Compact Matrix Decomposition (CMD)
  - Sun et al. ‘07
1. Singular Value Decomposition

\[ A = U \Sigma V^T \]

- **Optimal low-rank approximation**
- **Lack of Sparsity**

\[ A \approx U_k \Sigma_k V_k^T \]
2. CUR decomposition

C, U, R for small $\|A - \text{CUR}\|$

+ Provably good approximation to SVD
+ Sparse basis (A is sparse)
- Space overhead (duplicate bases)

basis vectors actual cols and rows of A
3. Compact Matrix Decomposition

C, U, R for small $||A-CUR||$, and No duplicates in C and R

+ Sparse basis ($A$ is sparse)
+ Efficiency in space and computation time
Reconstruction-based events

Network forensics
- Sparsification → load shedding
- Matrix decomposition → summarization
- Error Measure → anomaly detection
Error measure: reconstruction

- **accuracy** = 1 - Relative Sum-Square-Error
  \[
  RSSE = \frac{\sum_{i,j}(A(i,j) - \tilde{A}(i,j))^2}{\sum_{i,j}(A(i,j)^2)}
  \]

- Monitor accuracy over time

- Also, high reconstruction error of rows/cols for static snapshot anomalies

Volume monitoring cannot detect anomaly of link patterns
Practical issue 1: non-linear scaling

- **Issue:** skewed entries in A matrix
  - few “heavy” rows/cols dominate (CUR/CMD) decomposition
  - poor anomaly discovery

- **Solution:** rescale entries $x$ by $\log(x+1)$
Practical issue 2: fast approx. error

**Issue:** Direct computation of SSE is costly; norm of two big matrices, $A$ and $A - \tilde{A}$, are needed.

**Solution:** approximated error

$$\tilde{e} = \frac{m \cdot n}{|S|} \sum_{(i,j) \in S} \left( A(i,j) - C(i)UR(j) \right)^2$$

$$
\begin{pmatrix}
\vdots \\
A \\
\vdots
\end{pmatrix} \approx 
\begin{pmatrix}
\vdots \\
C \\
\vdots
\end{pmatrix} \cdot
\begin{pmatrix}
U \\
R
\end{pmatrix}
$$
Part II: References (graph series)


Part II: References (graph series) (2)

- Sun, Jimeng and Xie, Yinglian and Zhang, Hui and Faloutsos, Christos. Less is more: Compact matrix representation of large sparse graphs. ICDM 2007.
- Sun, Jimeng and Tao, Dacheng and Faloutsos, Christos. Beyond streams and graphs: dynamic tensor analysis. KDD 2006: 374-383
Tutorial Outline

- Motivation, applications, challenges
- **Part I:** Anomaly detection in *static* data
  - Overview: Outliers in *clouds of points*
  - Anomaly detection in *graph data*
- **Part II:** Event detection in *dynamic* data
  - Overview: Change detection in *time series*
  - Event detection in *graph sequences*
- **Part III:** Graph-based *algorithms and apps*
  - Algorithms: *relational learning*
  - Applications: *fraud and spam* detection