Dealing with Uncertainty in Massive Network Data

Nicholas Arcolano and Benjamin A. Miller
Graph Exploitation Symposium
April 16–17, 2013
Outline

• Introduction

• Inference on massive dynamic graphs

• Dealing with uncertainty
  – Statistical (Bayesian) approaches
  – Optimization (machine learning) approaches

• Summary
Computing and analytics, testbed development, and system prototyping to meet the nation’s defense challenges
The “Three V’s” of Big Data

• **Volume**
  – Need for parallel, high-performance computing and database solutions
  – Burden of massive computation limits the space of algorithmic approaches

• **Velocity**
  – Need rapid ingest, high-performance computing and resource management, and efficient algorithms to meet real-time demands

• **Variety**
  – Data are multi-source, multi-structured, and multi-modal
  – Situational awareness and decision support require more than just “black box” approaches
  – Need advanced analytics that include multi-source fusion, analyst feedback (operator-in-the-loop), and visual analytics
Data Veracity: “The Fourth V”

• We often deal with real-world data characterized by high uncertainty
  – Large amounts missing and corrupted data
  – Noise, clutter, hidden signals, deliberate misinformation

• Examples
  – Social, communication, and text/document networks
  – Geospatial and tracking data
  – Human intelligence, communication and text/document networks
  – Cyber and information networks

• Data veracity:
  – Entities, their characteristics, and connections between them may be incompletely or incorrectly observed
  – Data may represent imperfect observations of the quantities we really care about
  – Data may represent only a subset of a larger universe, about which we wish to infer
Graph Exploitation and Big Data

• Traditionally, big data analytics have been fairly limited in complexity
  – To start, just storing and working with the data was a challenge
  – Simple statistics: counts, averages, histograms
  – Centrality: which entities are the most important?

• Current and future trends (“advanced analytics”)
  – Modeling correlations between entities
  – Clustering / community detection
  – Detection, estimation, prediction
  – Extensions of classical statistical methods and machine learning approaches

• Graphs are a powerful mechanism for representing massive data
  – Big data sets often describe or contain relational networks of interest
  – Connectivity captures dependencies essential for higher-order modeling and analysis
  – Sparse connections are amenable to large-scale computation and storage
Outline

• Introduction

• Inference on massive dynamic graphs

• Dealing with uncertainty
  – Statistical (Bayesian) approaches
  – Optimization (machine learning) approaches

• Summary
Applications of Graph Analytics at MIT LL

**ISR**
- Graphs represent entities and relationships detected through multiple sources
- 1,000s – 1,000,000s tracks and locations
- GOAL: Identify anomalous patterns of life

**Social**
- Graphs represent relationships between individuals or documents
- 10,000s – 10,000,000s individual and interactions
- GOAL: Identify hidden social networks

**Cyber**
- Graphs represent communication patterns of computers on a network
- 1,000,000s - 1,000,000,000s network events
- GOAL: Detect cyber attack or malicious software

**Cross-Mission Challenge:**
Detection of subtle patterns in massive, multi-source, noisy datasets
Graph Analytics and the “Three V’s”

• **Volume**
  – Parallel computing and distributed data storage
  – Sparse representations (sparse adjacency matrices, associate arrays, triple stores)
  – Models that admit low-rank decompositions (e.g. Chung-Lu model)

• **Velocity**
  – More computation—real-time ingest, elastic grid computing, resource management
  – Dynamic graph models representing time-series data

• **Variety**
  – Computing and database capabilities allow us to build graphs out of massive, diverse data sets
  – However, in the end the graph usually only represents one or two types of entities and relationships
  – Still working towards multipartite, attributed graphs, latent-variable models, etc.

• What about **veracity**?
### Example: Anomalous Subgraph Detection

**Input**
- $A(t)$ – A series of adjacency matrices representing a time series of graphs $G(t)$
- No cue

**Output**
- $v_s(t)$ – A potentially dynamic set of vertices representing anomalous subgraphs of $G(t)$

**Graph processing chain** applies a signal processing approach to anomalous subgraph detection.

**Diagram**
- The graph processing chain involves the following steps:
  1. **Temporal Integration**
  2. **Graph Model Construction**
  3. **Residual Decomposition**
  4. **Component Selection**
  5. **Anomaly Detection**
  6. **Identification**

**Text**

- **Input**
  - $A(t)$: A series of adjacency matrices representing a time series of graphs $G(t)$
  - No cue

- **Output**
  - $v_s(t)$: A potentially dynamic set of vertices representing anomalous subgraphs of $G(t)$
Example: Anomalous Subgraph Detection

**Uncertainty Mechanism**

How does uncertainty in the observed data affect detection of anomalous subgraphs?

Dynamic graph with uncertain observations

Detection Performance with Observation Uncertainty

Arcolano and Miller -- GraphEx 2013
Dealing with Veracity

• For graph analytics to be truly useful for our applications, we need to be able to deal with uncertainty

• Two basic approaches
  – Model the uncertainty mechanism in the data (statistical / Bayesian approach)
  – Pursue methods that are robust to uncertainty in the data (machine learning approach)

• Some challenges
  – Tractable models can be too simple to represent real data
  – More sophisticated models can quickly become impractical for massive graphs
  – Uncertainty mechanisms often aren’t well understood for real data
Outline

• Introduction

• Inference on massive dynamic graphs

• Dealing with uncertainty
  – Statistical (Bayesian) approaches
  – Optimization (machine learning) approaches

• Summary
Latent Graph Estimation ("Graph Denoising")

- To illustrate these different approaches, consider the following general framework

- Given a simple graph $G = (V, E)$ with $n$ nodes and $m$ edges
  - May be random with prior model $P(G)$, or nonrandom and unknown
  - Assume $n$ is known

- We obtain $c$ noisy observations of this network $\mathcal{Y} \equiv \{G_1, \ldots, G_c\}$
  - Independent, but not necessarily identically distributed—observations may have varying levels of fidelity
  - May choose to assume and/or estimate an observation model $P(G_k | G)$
  - Example: multiple views of a social network via different mechanisms (email, telephone, online social network)

- Goal: recover an estimate $\hat{G} = f(\mathcal{Y})$
Statistical Approaches for Graph Denoising

- Standard approach is to choose an optimal estimate to minimize some loss function (e.g. edit distance)

\[
L(\hat{G}(\mathcal{Y}), G) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \mathbb{I}(\hat{a}_{ij} \neq a_{ij}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} (\hat{a}_{ij} - a_{ij})^2
\]

- In a statistical (Bayesian) framework, we assume \( G \) and \( G_1, \ldots, G_c \) are random and minimize expected loss (Bayesian risk)

\[
\hat{G}^*(\mathcal{Y}) = \arg \min_{\hat{G}(\mathcal{Y})} \mathbb{E}_G L(\hat{G}(\mathcal{Y}), G) \\
= \arg \min_{\hat{G}(\mathcal{Y})} \mathbb{E}_{G|\mathcal{Y}} L(\hat{G}(\mathcal{Y}), G) \\
= \arg \min_{\hat{G}(\mathcal{Y})} \sum_{G \in \Omega_n} L(\hat{G}(\mathcal{Y}), G) \left[ \prod_{k=1}^{c} P(G_k | G) \right] P(G)
\]
Statistical Approaches for Graph Denoising

- Within this framework, we need to assume (and usually estimate parameters for) \( P(G_k | G) \) and \( P(G) \).

- Example: if \( P(G_k | G) \) and \( P(G) \) are separable, i.e.

  \[
P(G_k | G) = \prod_{i=1}^{n} \prod_{j=i+1}^{n} P\left(a_{ij}^{(k)} | a_{ij}\right) \quad \quad P(G) = \prod_{i=1}^{n} \prod_{j=i+1}^{n} P(a_{ij})
\]

  then we can decompose the risk function and estimate each edge individually.

- For edit distance loss, we recover MAP estimation; corresponding decision rule says to set \( a_{ij} = 1 \) if

  \[
  \prod_{k=1}^{c} \frac{P\left(a_{ij}^{(k)} | a_{ij} = 1\right)}{P\left(a_{ij}^{(k)} | a_{ij} = 0\right)} > \frac{Pr\left(a_{ij} = 0\right)}{Pr\left(a_{ij} = 1\right)} \equiv \Gamma_{ij}
  \]
Example: Constant Error Rate with Erdős-Rényi Prior

- Prior: Erdős-Rényi with parameter $p$
  \[ P(a_{ij}) = p^{a_{ij}} (1 - p)^{1-a_{ij}} \]

- Noise model: constant probability of error $p_k$
  \[ \Pr(a^{(k)}_{ij} \neq a_{ij}) = p_k \]

- Decision rule: set edge $a_{ij}$ to 1 if
  \[ \prod_{k=1}^{c} \left( \frac{p_k}{1 - p_k} \right)^{1-2a^{(k)}_{ij}} > \frac{1 - p}{p} \]

- Tractable, but not terribly applicable to real network data—each edge is an independent entity

Example ROC Curves for Estimation of Erdős-Rényi Graph ($n = 100$)
Challenges of Statistical Approaches

• What if we don’t have the parameters?
  – Priors: can use decision-theoretic methods (EER, CFAR), though we may need a different threshold for each potential edge
  – Estimating the parameters is difficult, especially if we only get one observation with each level of fidelity
  – Performing ML estimation (uniform prior) is equivalent to assuming an Erdős-Rényi model with $p = 0.5$

• What models should we use?
  – Separable priors (e.g. Erdős-Rényi, Chung-Lu) tend not to represent real data very well
  – More sophisticated models (e.g. latent-variable models) often lead to an estimation problem that is intractable for massive data
  – Separability is a bit easier to tolerate in the observation model, but still may not be adequate
  – Can we model noise and corruption in real data well enough for it to even matter?
Outline

• Introduction

• Inference on massive dynamic graphs

• Dealing with uncertainty
  – Statistical (Bayesian) approaches
  – Optimization (machine learning) approaches

• Summary
Optimization Approaches for Graph Denoising

• Alternatively, let’s assume that we don’t have access to models for $P(G_k | G)$ and $P(G)$

• Loss function is no longer random, but remains a function of the observed data and depends on the unknown true network $G$

• What can we do? Some options:
  – Empirical risk minimization: treat observed data as true network
    $$\hat{G}^*(\mathcal{Y}) = \underset{\hat{G}(\mathcal{Y})}{\arg \min} \sum_{k=1}^{c} L(\hat{G}(\mathcal{Y}), G_k)$$
  – Constrained / regularized optimization: favor estimates with desirable properties (degree distribution, clustering, path lengths)
Optimization Approaches for Graph Denoising

• What if we assume we know a little more?

• Example: though we don’t know \( P(G_k \mid G) \), assume we do have some weight \( w_k \in (0,1) \) that measures each data source’s “reliability”
  – Weighted empirical risk

\[
\hat{G}^* (\mathcal{Y}) = \arg \min_{\hat{G} (\mathcal{Y})} \sum_{k=1}^{c} h(w_k) L(\hat{G} (\mathcal{Y}), G_k)
\]

  – For appropriate choice of mapping \( h \), we recover the decision rule from the Erdős-Rényi example

• Another example: what if we knew (or had a good estimate) of \( m \)?
  – Constrained weighted empirical risk

\[
\hat{G}^* (\mathcal{Y}) = \arg \min_{\hat{G} (\mathcal{Y}) \in \Omega_{n,m}} \sum_{k=1}^{c} h(w_k) L(\hat{G} (\mathcal{Y}), G_k)
\]

where \( \Omega_{n,m} \) is the set of all \( n \)-node simple graphs with \( m \) edges
Example: Constant Error Rate for Web of Science Network

• If the objective function is separable, we can estimate each edge individually
  – Edit distance loss without constraints
  – Similar to Bayesian approach

• Example: estimation of graph from Web of Science corpus
  – Directed graph containing citation network for 1920-1920
  – 53,786 vertices, 81,318 edges, 1.45 billion possible connections
  – We generate three observed networks where edges/non-edges are flipped with probabilities $p_1$, $p_2$, and $p_3$
  – Probabilities chosen to generate average of 1, 10, and 25 errors per vertex (respectively)
  – Also include results for an Erdos-Rényi graph for comparison

Average Number of Errors for Graph Denoising Approaches

<table>
<thead>
<tr>
<th>Method</th>
<th>Average Errors Per Node (Web of Science)</th>
<th>Average Errors Per Node (Erdos-Rényi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empirical risk</td>
<td>3.0335</td>
<td>3.0285</td>
</tr>
<tr>
<td>Weighted empirical risk</td>
<td>0.9996</td>
<td>1.0001</td>
</tr>
<tr>
<td>Constrained weighted empirical risk</td>
<td>0.00052</td>
<td>0.00047</td>
</tr>
<tr>
<td>Bayesian (Erdős-Rényi prior)</td>
<td>0.00052</td>
<td>0.00046</td>
</tr>
</tbody>
</table>
Challenges of Optimization Approaches

• Computational complexity
  – Separable objective functions are computationally tractable, but edge-wise optimization essentially reduces to voting methods
  – Constrained approaches quickly become intractable

• Regularization and constrained optimization
  – To move beyond the most basic methods, we need to incorporate limited side information (confidence/reliability, favorable network structures)
  – Where do we get this information, and how robust are the resulting algorithms?

• As with statistical approaches, lack of ground truth makes validation on real data difficult
Outline

• Introduction

• Inference on massive dynamic graphs

• Dealing with uncertainty
  – Statistical (Bayesian) approaches
  – Optimization (machine learning) approaches

• Summary
Summary

• Big data means volume, velocity, and variety, but we also need to worry about veracity

• We need solutions that can work in noisy, multi-source settings, while remaining scalable to massive data sets

• There are different schools of thought for dealing with noisy network data
  – Traditional statistical approaches provide solid theoretical results, but scaling sophisticated models to massive graphs can be prohibitive
  – Optimization / machine learning approaches rely on fewer assumptions, but face similar computational challenges
  – Sometimes they give us the same answers

• Lots of open theoretical and practical questions
  – When are simple models “good enough”?
  – How big is the effect of prior assumptions about the underlying network?
  – What can we do if we only observe a single graph?