Abstract – The explosion of social media and availability of open source data has created an unprecedented opportunity to expose hidden intelligence. In this work we explore the use of a pipeline of novel algorithms executing within a parallel computing framework to improve SNA performance as compared to baseline approaches. These initial findings show improvements in link detection speed and accuracy (F-measure) on community datasets with ground truth information.

Keywords: Social Network Analysis, Machine Learning, Non-obvious Link Extraction, Parallel Machine Learning Frameworks, Noise Reduction and Removal of Irrelevant Intelligence

1 Introduction
The explosion of social media and availability of open source data has created an unprecedented opportunity to expose hidden intelligence. Illicit and disruptive activities are being carried out in plain view including protests and social movements, attack planning and coordination, plotting flash riots and robberies, promoting radical ideology and recruiting new members. Uncovering actionable intelligence or manually ‘connecting the dots’ in overwhelming, heterogeneous and noisy social media data streams has proven extremely difficult for the U.S. and our coalition partners.

The overall goal of this research is to investigate techniques which improve the accuracy and speed of social network analysis (SNA) while using fewer computing resources as compared to baseline approaches. In this paper, we describe ongoing research performed and results obtained to date by employing a combination of the latest machine learning techniques within a parallel computing framework. We have evaluated our research using community datasets with ground truth information and report encouraging results.

2 Related Work
A significant portion of current SNA research is directed towards analyzing data streams to collect intelligence; whether to analyze buying trends for the commercial market space or detecting nefarious activities for government intelligence entities.

Sitaram Asur and Bernardo A. Huberman from the HP Social Computing Lab in Palo Alto, CA have investigated how social media can be used to predict future events. They have used Twitter data to predict movie revenues. Their approach considers simple metrics such as tweet frequency and sentiment to predict outcomes.

Aditya Krishna Menon and Charles Elkan at the University of California investigated link prediction with an approach of supervised matrix factorization. Their approach uses features based on the graphs topological structure combined with other node and edge attributes. Stochastic gradient descent is used for ranking and class imbalance.

Hila Becker and Luis Gravano from Columbia University and Mor Naaman from Rutgers University have researched using Twitter messages to identify real-time events. Their work is focused on distinguishing between messages about events vs. non-events based on topic clusters over large message sets.

Due to the overwhelming amount of data, many current approaches will be CPU intensive. Our approach differs by employing a method of multi-stage algorithms, each narrowing the result by filtering unrelated information, detecting hidden but relevant relations and networks in a very efficient parallel framework for graph processing.

3 Approach
In this paper we explore the use of a pipeline of functions and algorithms executing within a parallel
computing framework. The pipeline consists of a series of capabilities designed to gather, detect and narrow results based on the user's original intelligence requirement (IR). The pipeline contains functional elements for performing data acquisition, noise reduction, network detection, link detection, and node influence.

3.1 Data Acquisition

In this functional element, social media data streams are queried based on the analyst's IR and parsed for key elements of information.

The system narrows the initial search to targeted topics and data sources based on the IR. The IR is parsed for keyword information including synonyms using a thesaurus. These are used to create a matrix of permutations on the original IR in order to achieve maximum coverage over the data sources.

3.2 Noise Reduction

A key function is reducing the ambient noise encountered in social media data streams. We have implemented an algorithm described in a paper by Dr. Carlos Guestrin and Yison Yue entitled “Linear Sub modular Bandits (LSB) and their Application to Diversified Retrieval”. The LSB Greedy algorithm as detailed in this paper takes advantage of sub modular optimization and the upper confidence bounds applied to trees (UCT) in order to describe a noise reduction technique that is both fast and accurate.

The equation $E[r_t^I * A] = W^* T \Delta (A^I | A^{1:t-1})$ represents the optimization function for the linear sub modular bandit’s problem. The $W^* T$ is the unknown weight vector for the users’ intelligence request. Optimizing for this weight threshold allows us to tune down the noise and help return only those results related to the IR. The linear sub modular bandits algorithm follows the exploration vs. exploitation tradeoff scheme and allows for the user to tune the exploration and exploitation values for maximum/desired coverage over the dataset.

Due to the overwhelming amount of social media data, approaches such as clustering could take an inordinate completion time on standard computer. Dr. Guestrin’s algorithm takes minutes to execute. We are continuing to collaborate with Dr. Guestrin to optimize and tune the algorithm implementation. Implementation of the Dr. Guestrin’s noise reduction algorithm decreased relative noise by 67% as measured by one user study at CMU which compared the LSB Greedy with competing algorithms.

3.3 Network Detection and Significance

The concept of a network is to represent a graph structure across varying data sources. A network is any structure which is repeated in a statistically significant manner. Within these structures exists well established structures such as the “feed-forward loop” or FFL that represents an AND or an OR gate inside of the network. These structures can be used in a machine learning algorithm as features to determine a network’s significant values such as clustering coefficient, network density, and network connectedness. We have examined three possible network finding algorithms in order to assist with the task of finding both structural and content networks. The algorithm chosen for this process is the G-trie algorithm. The G-trie algorithm works by placing random walkers on the network and finding networks of user defined sizes to create networks. These networks will then be used in the statistical calculations in order to determine event classification type.

G-Tries

G-Tries is a very high compression algorithm based on the famous trie data structure. This structure allows for maximum compression of the underlying data and easier computability of the sub-graphs. A consideration when using tries is that starting from different points on a graph will give widely different data structures; to solve this, g-trie uses a common canonical representation where larger vertices appear first. The algorithm is split up into three major sections. The first is the algorithm to insert a sub-graph into a g-trie, the second is to find if the inserted graph is significant over all possible graphs, and the third is designed to stop recompilation of networks that have already been examined.

---


Algorithm 1 Inserting a graph G in a g-trie T

1: procedure INSERT(G, T)
2: \hspace{1em} M := canonicalAdjMatrix(G)
3: \hspace{1em} insertRecursive(M, T, 0)
4: procedure insertRecursive(M, T, k)
5: \hspace{2em} if k < numberOfNodes(M) then
6: \hspace{3em} for all children c of T do
7: \hspace{4em} if c.value = first k + 1 values of M[k] then
8: \hspace{5em} insertRecursive(M, c, k + 1)
9: \hspace{2em} return
10: \hspace{1em} n := new g-trie node
11: \hspace{1em} n.value := first k + 1 values of M[k];
12: \hspace{1em} T.insertChild(n)
13: \hspace{1em} insertRecursive(M, n, k + 1);

Figure 1: G-trie inserts a sub-graph

Algorithm 2 Census of subgraphs of T in graph G

1: procedure CENSUS(G, t)
2: \hspace{1em} for all children c of T.root do
3: \hspace{2em} match(c, G, 0, 0)
4: procedure MATCH(T, G, k, V_used)
5: \hspace{2em} if V_used = \emptyset then
6: \hspace{3em} V_cand := V := G
7: \hspace{2em} else
8: \hspace{3em} V_conn := \{V_used[i] : T.value[i] = '1'\}
9: \hspace{2em} m := \min_{V \in V_conn : \forall v \in V_conn, |N(m)| \leq |N(v)|} V
10: \hspace{2em} V_cand := \{v \in N(m) : v \not\in V_used\}
11: \hspace{2em} for v \in V_cand do
12: \hspace{3em} add v to end of V_used
13: \hspace{2em} if T.isLeaf() then
14: \hspace{3em} reportGraph()
15: \hspace{2em} else
16: \hspace{3em} for all children c of T do
17: \hspace{4em} match(c, G, k + 1, V_used)
18: \hspace{2em} remove v from V_used

Figure 2: Finding sub-graph significance

Algorithm 3 Symmetry breaking conditions for graph G

1: function FINDCONDITIONS(G)
2: \hspace{1em} Conditions := \emptyset
3: \hspace{1em} Aut := setAutomorphisms(G)
4: \hspace{1em} while |Aut| > 1 do
5: \hspace{2em} m := minimum v : \exists map \in Aut, map[v] \neq v
6: \hspace{2em} for all v \neq m : \exists map \in Aut, map[m] = v do
7: \hspace{3em} add m < v to Conditions
8: \hspace{1em} Aut := \{map \in Aut : map[m] = m\}
9: \hspace{1em} return Conditions

Figure 3: Avoiding redundant calculations

The ability for g-tires to avoid the redundant calculations induced in the census method of random graphs leads to a much faster algorithm. The ability of the algorithm to include both a computationally efficient data structure and the ability to break symmetries leads to the fastest available algorithm for discovering the random networks.

3.4 Feature Discovery

An important aspect of this research is this identification of connections between two nodes which are important to the analyst. We are investigating the combination of both noise reduction algorithms and multivariate Bayesian scan statistics in order to determine which connections are important and which ones have significant meaning for the analyst. These statistics are common to each network and will be averaged over all networks and categorized for event classification. These features include but are not limited to:

- Average Sentiment: Twitter POS tagger + Opinion Lexicon (English, Spanish)
- Average Tweets per second: \( A = \text{Tweets" } / \text{(Event Begin-Event End )} \)
- Event time is measured in seconds from the year 1970
- Average Clustering Coefficient
- Event time is measured in seconds from the year 1970
- Average Time active
- Average Influence of Network
- Average number of high influencers
- Topics tweeted via Latent Dirichlet Allocation
- Average network retweets
- Location distance of members (e.g. proximity of one member to another)
- Friends/Followers associates
- Geolocation of members
- Location in relation to event

3.5 Link Detection

A focus of this research is on the ability to find implicit links or relationships that a user might not see due to the overwhelming volume of disparate data. We have researched two detection methods for their efficacy and scalability. Both methods improved precision and recall over the current state of the art, but utilize online learning algorithms that improve performance by four fold over the traditional offline methods without significant loss of precision or recall. Online learning states that all data is not available at processing time and updates the link model incrementally as new data comes into the system. Offline learning processes requires that all data is present and processed at one time. When new data is added, all information must be reprocessed. Online processing is well suited to streaming data sources from large data sources such as Twitter or Facebook.

We have tested performance of the implicit relation detection algorithm on Twitter and Facebook (anonymized) datasets. Using these data sources we
have discovered very interesting characteristics of implicit relations in social networks. First, the “small world” phenomenon is vastly overstated in most of the literature. This phenomena is explained by the statement “No person is more than six degrees away from another”. In processing the data we found that people are usually more than the standard six degrees apart if group aggregation is taken into account. We discovered many niche communities connected by separate networks, this contradicts the small world assumption as the lengths of these single chains was found to be around 10 nodes.

Preliminary results of the inferred entity relation algorithm using 10 folded crossover validation shows an 85% accuracy for inferred links without side channel information. Adding this information boosts the algorithm’s accuracy to around 91.2% using the same data sets.

The biggest challenge in discovering relationships within the data was not the absence of the small world metric but the degrees of separation within groups. If all the friends/followers from a Twitter or Facebook dataset are graphed, a fully connected graph state is quickly reached. Most of these friends are erroneous and just placed in the list because they are either celebrities, corporate entities, or have a very loose association with the entity over the internet. With this data, we learned that going with a full duplex communications solution solves this issue and provides increased accuracy over more lenient models.

As depicted in the figure below, our link detection approach uses a multi-dimensional feature analysis with values plotted for Singular Value Decomposition, Modified Katz Index and Side Info associated with each node. The red circles represent nodes within the social network which are not connected. Blue circles represent nodes which are related to one another.

As shown in the next diagram, the multi-dimensional plane is processed using the Vowpal Wabbit to determine best fit of each node within a given hyper plane. The significance of this result is that nodes on same hyper plane share a relationship. Vowpal Wabbit is an extremely fast online learning algorithm out of Yahoo and Microsoft Research.

One of the most significant aspects of this approach is that as nodes change (e.g. new tweets or Facebook updates) or new social media users are added, the graph is quickly updated (i.e. online processing). Additionally, the implementation within GraphLab makes this efficient to add new nodes and allows
scalability from the single computer up to a cloud based solution.

**Figure 6.** Response times are significantly reduced as new information becomes available within the social network. New points or nodes are quickly classified and placed on their appropriate hyper plane according to relations with other nodes on the same plane.

The final outcome of the pipeline is that groups of related nodes can be shown as well as relations between networks of networks as shown below. Network neighborhoods allow for discovery of who is more likely to discuss certain topics and allow the user to quickly discover new and interesting connections present in the data.

**Network neighborhood**

Network neighborhoods are particularly important for many analytical reasons. Displaying the various groups and their connections provides explicit data of how different cells are related. These relations can depict a multitude of relation types such as communication, transaction, familial, and contagion. Typically when analysts search, they focus on very specific cliques that are usually homogenous. However, the network neighborhood feature allows the analysts to gain both local and global intelligence across a plethora of node and edge types.

The network neighborhood feature is derived from the Layered Network Analysis model that was defined by LaMonica and Waskiewicz (2011) and refers to the fusion of previously disconnected data in a common operating picture allowing the user to explore and exploit interconnected data in a unified state. Layered Network Analysis can be used to discover patterns within and across layers, discover previously unknown nodes and edges, and allow users to focus on multiple data types within a specific time and space.

The intent is to understand the structure and dynamics of network data as this data is of vital importance to decision makers. Current analysis of network data occurs primarily within one or two particular domains and relies on the user to infer and derive links between data sources. However, to fully understand the network environment, decision makers must be able to investigate interconnected relationships of many diverse network types simultaneously as they evolve spatially and temporally. No single data network exists in a vacuum as networks are interconnected through space and time. Each layer represents a diverse network data type and these layers can be connected through nodal and edge similarities providing both local and global situation awareness.

### 3.6 Node Influence

We use a Modified Decreasing Cascade Model within PageRank statistics to determine which node has the most influence within a given network (e.g. # of retweets, # of connections, time until topic is tweeted, etc.). This influence score may allow the analyst to decide who should be monitored for further study or followed more closely. Influence scores are updateable and amendable by the analyst; the user can change the weights of the scores and recalculate influence based on this prior information. As shown in the graphic below, color intensity could indicate relative influence of individuals as compared to other members of the same neighborhood.
Figure 8. Identify the most influential nodes within a social network of users with similar interests. The strength of influence can be indicated by color intensity.

4 Results

Thus far, we have tested the link detection capability on two datasets:

- The co-author network of astro-physicists. SNAP provided 18,772 nodes with 396,160 edges.
- Co-author network of NIPS. Over 10,000 nodes and 600,000 edges. Each node has a 14035 word index and bag-of-words feature vector.

The output was compared against other approaches to classify nodes within a graph:

- Majority Classifier – Every point in the classifier is assigned to whichever the majority class is in the dataset (baseline).
- Random Classifier – Random flip if the node is connected or not (baseline).
- Perceptron Learner – Simple gradient descent learning algorithm with 2 layers of hidden nodes and trained with back propagation.
- Bayesian Network – Network of probabilities trained from the input data using EM algorithm to discover unknown values.
- SVM (kernel gradient optimization) – Using sub gradients to find the proper kernel metric and applying a SVM to the problem to generalize.

The link detection speed and accuracy exceed those of current baseline approaches. As compared to a Support Vector Machine, Securboration’s approach is 322% faster with 5% better accuracy; compared to the Bayesian approaches testing showed 222% speed and 15% accuracy improvements. This particular test was based on a Twitter dataset with 81,306 nodes and 1,768,149 edges.

Figure 9: Securboration SNA Classifier Time to Complete

Figure 10: Securboration SNA Classifier Accuracy

Figure 11 below shows a social network graph output from LAKE. The purple nodes in the graph are topics extracted from social media streams based on IR inputs. The red nodes are people with links to the topics and other individuals as detected by LAKE. The numbers sequences in black are unique ids for actual social media messages posted by the user and related to the IR. Relation strength is indicated by the edge thickness. The user can right click on graph nodes to display more information about the user and relevant messages as indicated by the green arrow.
5 Conclusion and Future Work

This research represents a step forward in the development of tools which automate the analysis of social media data streams. The research uses the latest advances in the field of parallel processing, machine learning and data mining to increase speed, accuracy and precision of intelligence processing. This initial research proved that select algorithms could be configured in a pipeline and executed on standard computers to produce results which are superior to the current baseline. One objective of this research is to build a capability which can be deployed on standard hardware environments to increase adoption of the system. Future work will focus on extending this research and the development of a service based suite of analytical tools to support the defense, intelligence, and law enforcement communities.

Acknowledgement – Work described in this paper was funded by the Air Force Research Laboratory (AFRL)’s Information Directorate, Rome, New York.

6 References


Menon, A. K., Elkan, C., Link Prediction via Matrix Factorization, University of California
