Where Do I Start? Algorithmic Strategies to Guide Intelligence Analysts

Hao Wu[1,3], Michael Mampaey[4], Nikolaj Tatti[5],
Jilles Vreeken[5] , M. Shahriar Hossain[2,3], Naren Ramakrishnan[2,3]

[1]Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA
[2]Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA
[3]Discovery Analytics Center, Virginia Tech, Blacksburg, VA 24061, USA
[4]Department of Computer Science, Utrecht University, Netherlands
[5]Department of Mathematics and Computer Science, University of Antwerp, Belgium

ABSTRACT

The "where do I start?" problem is a veritable one in intelligence analysis. We identify several classes of algorithmic strategies that can supply starting points to analysts in their exploration of a document collection. We present nine methods with origins in association analysis, graph metrics, and probabilistic modeling, and systematically evaluate them over multiple document collections. One of these methods, a novel approach to modeling "surprise", is our specific contribution and, further, supports the iterative refinement of suggestions based on user feedback. We demonstrate how these methods guide the analysts to start their investigation on intelligence document collections. Our results reveal selective superiorities of the algorithmic strategies and lead to several design recommendations for creating document exploration capabilities.

Categories and Subject Descriptors

H.2.8 [**Database management**]: Database applications—*Data mining*; H.3.3 [**Information storage and retrieval**]: Information search and retrieval—*Information filtering, Retrieval models*

Keywords

Text Mining, Maximum Entropy, Starting Points, Intelligence Analysis

1. INTRODUCTION

Intelligence analysts today are faced with many challenges, chief among them being the need to fuse disparate streams of data, and rapidly arrive at analytical decisions and quantitative predictions for use by policy makers. Modern communication forms such as social media and microblogs have increased the diversity of sources available but have also created further challenges, especially the need for algorithms and software tools to suitably guide the analyst and complement human analytical skills.

One of the veritable problems for an intelligence analyst is: given a collection of documents to analyze, where do I begin? Which entities should one look at first? What activities appear suspicious to warrant further investigation and followup? This problem is further exacerbated by the imprecise form of threats that intelligence analysts seek to unravel, by the developing nature of events, and by the prevalence of incorrect or misleading data. Nevertheless, once initial leads are identified, analysts can adopt a range of structured exploration strategies that can contribute to hypothesis formation or to conclusively discarding the current line of investigation.

In this paper, we present nine algorithmic strategies that codify aspects of starting points for investigations. These strategies are drawn from diverse backgrounds and work under different assumptions. However, these algorithmic strategies only aim to find starting points in the intelligence dataset, while not the entire solution, which can be found easily by various existing intelligence analysis tools and storytelling algorithms (e.g. [18], [19], [20], [27]). Our contributions are:

1. We present nine methods with origins in association analysis, graph metrics, and probabilistic modeling, and systematically evaluate them over multiple document collections. To the best of our knowledge, our work is the first to conduct a systematic review of algorithms for starting point generation.

2. We introduce MTV, an algorithm adapted from the itemset mining community, as a novel approach to modeling "surprise", and especially suited for starting point generation. Further, MTV supports the iterative refinement of suggestions based on user feedback.

3. We perform a systematic evaluation over five datasets which lead to understanding selective superiorities of the algorithms and subsequent recommendations for future work. We also apply the MTV algorithm on the ISI-KDD 2012 challenge dataset.

2. RELATED WORK

Modern software tools to support intelligence analysis are motivated by different considerations. Specific categories of systems include association rule mining based systems [29, 5], classification based tools (e.g., [37, 36]), model-guided software (e.g., [2, 26, 9]), graph-based frameworks (e.g., [16, 10, 11, 8]), collaborative systems (e.g., [3, 7, 4]), and multi-agent systems (e.g., [30]). These projects however do not explicitly address the issue of identifying starting points for analysis.

The analysis process in a typical intelligence analysis exercise can be viewed through the framework of Pirolli and Card [33],

ISI-KDD'12, August 12, 2012, Beijing, China.

viz. information foraging and sensemaking. Information foraging is aimed at seeking information, searching and filtering it, and reading and extracting information possibly into some schema. Sensemaking involves iterative development of a mental model (a conceptualization) from the schema that best fits the evidence. Some analytic systems, such as *INSPIRE* [34], *Jigsaw* [22], *ThemeRiver* [17], *NetLens* [24] focus on supporting the information foraging loop. However, none of these tools are aimed at discovering a specific start point for the analysts. Other tools, such as *Analyst's Notebook* [21], *Sentinel Visualizer* [15], *Entity Workspace* [4], and *Palantir* [25] focus more on sensemaking, and relegate the task of finding start points to the domain knowledge of intelligence analysts. In our paper, we explicitly aim to provide a start point for analysts as a foraging task that fosters the sensemaking step later.

3. ALGORITHMS

We survey our proposed algorithms in terms of three broad categories: (i) association measures, (ii) graph modeling, and (iii) probabilistic modeling of surprise. Each of these categories is discussed below. But first we define some notations to be used throughout the rest of this paper.

We consider as given a set of documents $D = \{d_1, d_2, \cdots, d_n\}$ and a set of entities $E = \{e_1, e_2, \cdots, e_m\}$ (which could have been extracted from the documents using standard tokenization and NLP software). The occurrence information of entities in documents can be modeled using a relation R between D and E, or as a (weighted) bipartite graph. For ease of description, however, we model a document as a set of entities, i.e., $d_i = \{e_1, e_2, \cdots, e_j\}$. Thus, the given dataset can be denoted as the tuple (D, E).

3.1 Association Measures

Association rule mining is a well studied research problem in data mining, whose aim is to find associations between different items in a large transaction database. Here, a transaction is modeled as a set containing a list of items that appear in the transaction. Notice that since a document can be represented as a set of entities, we can interpret the documents as transactions and entities as items in the transactions. Under this scenario, we can apply association rule mining techniques to find association rules between entities in the documents. Association measures typically capture notions of frequency and co-occurrence that we believe will be useful to the intelligence analyst.

There are several existing and well studied association rule measures, including ϕ-coefficient, Odds ration, Yule's Q, Gini Index, Confidence, Interest, and the cosine measure[38]. After support pruning, many of these association measures tend to be similar to each other. This topic is beyond the scope of this paper and the interested reader is referred to [38]. Here, we choose three association measures surveyed in [38]: ϕ-coefficient (ϕ), Gini Index (G), and confidence (c). For a given association rule $X \Rightarrow Y$, these measures can be calculated using the formulas in Equation (1), (2) and (3), respectively:

$$\phi = \frac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)P(Y)(1-P(X))(1-P(Y))}} \quad (1)$$

$$G = \max(P(X)[P(Y|X)^2 + P(\bar{Y}|X)^2] + P(\bar{X})[P(Y|\bar{X})^2 + P(\bar{Y}|\bar{X})^2] - P(Y)^2 - P(\bar{Y})^2, P(Y)[P(X|Y)^2 + P(\bar{X}|Y)^2] + P(\bar{Y})[P(X|\bar{Y})^2 + P(\bar{X}|\bar{Y})^2] - P(X)^2 - P(\bar{X})^2) \quad (2)$$

$$c = \max(P(Y|X), P(X|Y)) \quad (3)$$

where $P(X)$ denotes the probability of entities in X appearing in D, $P(\bar{X})$ denotes the probability of entities in X not appearing in D, $P(X,Y)$ denotes the probability of entities in X and Y appearing in D together, and $P(Y|X)$ denotes the conditional probability of entities in Y appearing in D given the entities in X appearing in D. (The meaning of other symbols are analogous.) Finally, the top, e.g., 10, association rules are supplied to the intelligence analyst based on ranking using specific measures.

3.2 Graph measures

Since the relationship between documents and entities is naturally viewed as graphs, graph measures such as dense subgraphs and central vertices are natural candidates for generating starting points. We will focus on two kinds of graph representations: the entity-entity graph induced based on co-occurrence in documents, and the original document-entity bipartite graph. We will discuss both kinds of representations and algorithms below.

3.2.1 Entity-entity graph

Given a document dataset (D, E), an entity-entity graph is an undirected graph $\mathscr{G}_{entity} = (\mathscr{V}, \mathscr{E})$ where:

$$\mathscr{V} = E$$
$$\mathscr{E} = \{(e_i, e_j) | e_i, e_j \in d_k, \textit{ and } e_i, e_j \in E\}$$

$\mathscr{V}$ is the vertex set, and each entity in E is a vertex in the graph. $\mathscr{E}$ is the edge set, and if entities e_i and e_j appear in a certain document d_k together, then the vertices e_i and e_j are connected.

Algorithm 1 Greedy approximation algorithm for identifying a dense subgraph

```
S ← G = (V, E)
S_dense = S
f_dense = f(S)
while S ≠ (∅, ∅) do
    v_min = findMinDegreeVertex(S)
    S = deleteVertexAndEdges(S, v_min)
    if f(S) > f_dense then
        f_dense = f(S)
        S_dense = S
    end if
end while
return S_dense
```

We implement a greedy approximation algorithm [6] to find the dense subgraph in the undirected entity-entity graph. Before proceeding to describe this greedy algorithm, we must first define the density measure of an undirected graph. Given an undirected graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, the density of graph $\mathscr{G}$, $f(\mathscr{G})$, is given by the formula [6]:

$$f(\mathscr{G}) = \frac{|\mathscr{E}|}{|\mathscr{V}|} \quad (4)$$

where $|\mathscr{E}|$ denotes the number of edges, and $|\mathscr{V}|$ denotes the number vertices in graph $\mathscr{G}$. A clique, therefore, has the highest density among all graphs containing a certain number of vertices.

The greedy approximation algorithm for finding the dense subgraph works in an iterative manner. Given the graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, the algorithm is initialized with $S = \mathscr{G}$. In each iteration, the vertex with minimum degree, denoted by v_{min}, is identified and deleted from S. The edges connected to the vertex v_{min} are also removed from S. Then, a subgraph of $\mathscr{G}$ is generated and its density $f(S)$ is calculated. The algorithm continues until S becomes an empty graph. Among all the subgraphs constructed during this process, the subgraph that maximizes the density measure in Equation (4) is output by the algorithm. The pseudocode in Algorithm 1 gives the details of this greedy approximation algorithm. Figure 1 also

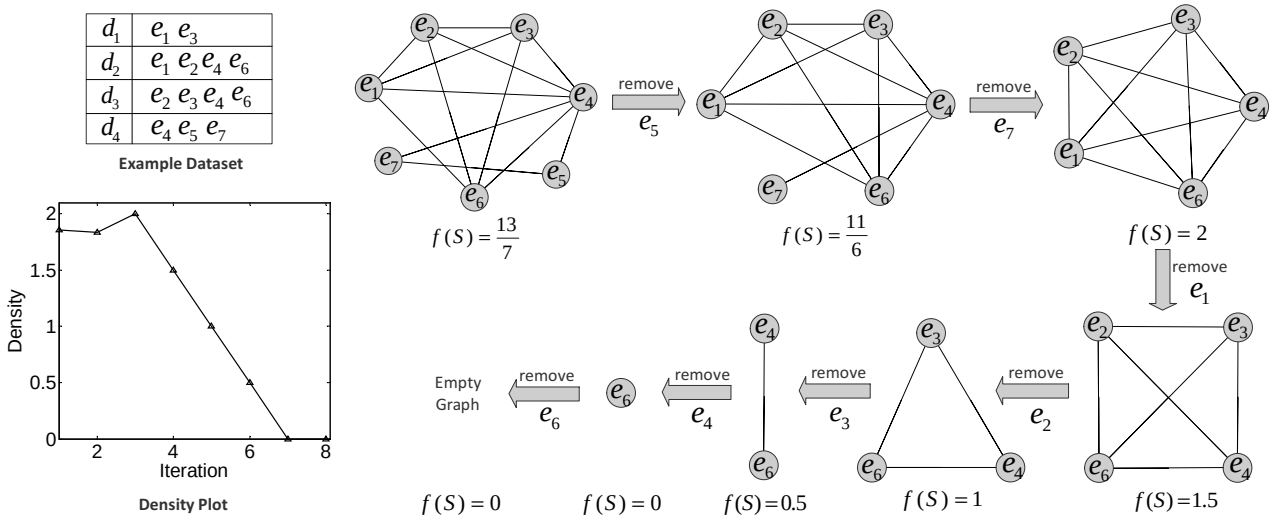

Figure 1: Illustration of the greedy approximation algorithm [6] to identify a dense subgraph.

shows an example dataset comprising 4 documents and 7 entities, and the process of applying the iterative greedy algorithm to the entity-entity graph constructed from this example dataset.

3.2.2 Document-entity bipartite graph

We now turn our attention to a different representation. Given a document dataset (D,E), a document-entity bipartite graph is an undirected graph $\mathcal{G}_{doc} = (\mathcal{V},\mathcal{E})$ where:

$$\begin{aligned} \mathcal{V} &= D \cup E \\ \mathcal{E} &= \{(d_i,e_j) | e_j \in d_i, \text{ and } d_i \in D, e_j \in E\} \end{aligned}$$

Here $\mathcal{V}$ is the vertex set as before, and all documents from D and all entities from E form vertices in the document-entity bipartite graph. $\mathcal{E}$ is the edge set, and if document d_i contains entity e_j, there will be an edge between the vertices d_i and e_j.

In order to find dense subgraphs from the document-entity bipartite graph, we implement a heuristic local search algorithm [28]. Since the vertex set of the document-entity graph contains two kinds of vertices, viz. document vertices and entity vertices, a slight different density measure is adopted here. Given an undirected document-entity bipartite graph $\mathcal{G}_{doc} = (\mathcal{V},\mathcal{E})$, the vertex set $\mathcal{V}$ can be represented as $\mathcal{V} = D \cup E$, and the density measure for the bipartite graph $\mathcal{G}_{doc}$ is defined by:

$$f(\mathcal{G}_{doc}) = \frac{|\mathcal{E}|}{|D||E|} \tag{5}$$

where $|\cdot|$ denotes the number of elements in a particular set.

The heuristic local search algorithm finds the dense subgraph from an undirected bipartite graph $\mathcal{G}_{doc} = (\mathcal{V},\mathcal{E})$ in three steps. First, a size of $m \times n$ bipartite subgraph $\mathcal{G}' = (\mathcal{V}',\mathcal{E}')$ is initialized in a random manner, where $\mathcal{V}' \subset \mathcal{V}$, $\mathcal{E}' \subset \mathcal{E}$, $\mathcal{V}' = D' \cup E'$, $D' \subset D$, $E' \subset E$, $|D'| = m$, and $|E'| = n$. Then, a series of local swap operations (LocalSwap procedure) is applied on the initial bipartite subgraph $\mathcal{G}'$ until $\mathcal{G}'$ does not change any more. The local swap operation tries to replace the document (or entity) vertices in the subgraph with one not in the subgraph so that the number of edges in the subgraph will increase. Notice that the local swap operation will not change the number of vertices in the bipartite subgraph, which implies that the bipartite graph density measure in Equation (5) keeps increasing.

Secondly, a local resize operation (LocalResize procedure) is applied to the bipartite subgraph that is returned by the LocalSwap procedure. The goal of this operation is to allow the size of the bipartite subgraph to change beyond its original size of $m \times n$. In the local resize operation, the document (or entity) vertices not in the bipartite subgraph are added to the subgraph if they are connected to more than $\frac{2}{3}$ of the entities (or documents) in the subgraph. The document (or entity) vertices in the subgraph can also be deleted from subgraph if they are connected to less than $\frac{1}{3}$ of the entity (or document) vertices in the subgraph. In our implementation, we define the bipartite subgraph $\mathcal{G}'$ as changing too much if the set of document vertices or the set of entity vertices changes too much. The document vertex set changes too much if $|D'| > 2m$ or $|D'| < \frac{1}{2}m$, and the entity vertex set changes too much if $|E'| > 2n$ or $|E'| < \frac{1}{2}n$.

Algorithm 2 Local search algorithm for dense bipartite subgraph

1: Initialize a size of $m \times n$ bipartite subgraph $\mathcal{G}' = (\mathcal{V}',\mathcal{E}')$ in a random manner from the bipartite graph $\mathcal{G}_{doc} = (\mathcal{V},\mathcal{E})$.
2: $\mathcal{G}' = \text{LocalSwap}(\mathcal{G}', \mathcal{G}_{doc})$
3: $\mathcal{G}' = \text{LocalResize}(\mathcal{G}', \mathcal{G}_{doc})$
4: $\mathcal{G}' = \text{LocalSwap}(\mathcal{G}', \mathcal{G}_{doc})$
5: **return** $\mathcal{G}'$

Finally, the LocalSwap procedure is applied on the bipartite subgraph returned by the LocalResize procedure again to further improve the density measure of the subgraph. After this iteration of LocalSwap procedure completes, we obtain the dense bipartite subgraph desired. Algorithm 2 summarizes this heuristic local search algorithm. Figure 2 shows the process of applying this heuristic local search algorithm to the document-entity bipartite graph built from the example dataset shown in Figure 1.

3.2.3 Graph centrality measures

In a graph, the centrality of a vertex measures the relative importance of this vertex within the graph. Thus, in the graph constructed from documents and entities, it is straightforward to connect centrality measures of graphs to the importance of the documents and entities. An entity vertex that has high centrality measure may be important to the intelligence analysts. We survey three kinds of centrality measures: degree centrality, betweenness centrality, and closeness centrality [32]. Given an undirected entity-entity graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$, the centrality measures for a vertex $v \in \mathcal{V}$ are defined as follows:

- **Degree centrality**: the degree of vertex v over the number of

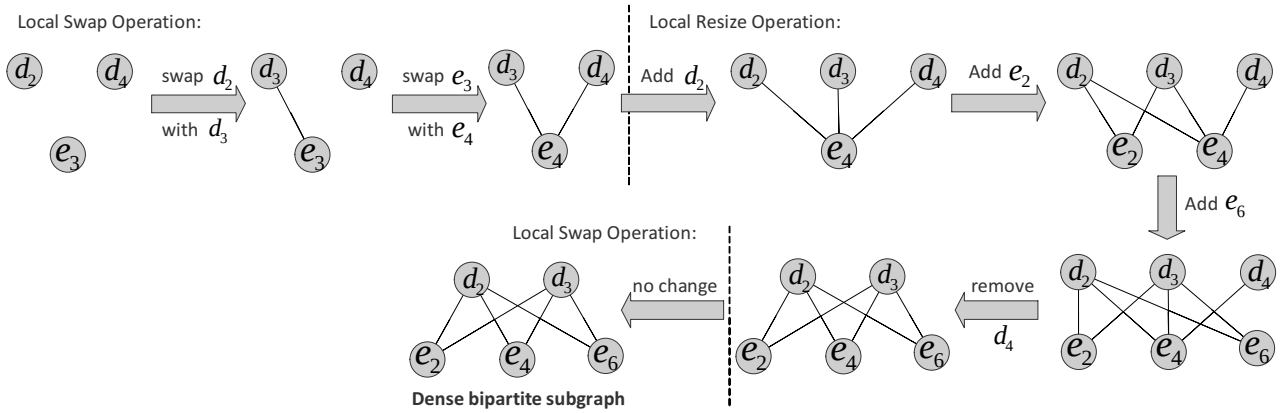

Figure 2: Example of the local search algorithm [28] to find dense subgraphs in a bipartite graph.

vertices in the graph.

$$C_D(v) = \frac{degree(v)}{|\mathscr{V}|}$$

- **Betweenness centrality**: an index for the number of times that vertex *v* appears on the shortest paths between any other vertices.

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}, \quad s,t \in \mathscr{V}$$

where $\sigma_{st}(m)$ denotes the number of shortest paths between vertices *s* and *t* that pass vertex *m*, and σ_{st} denotes the total number of shortest paths from *s* to *t*.

- **Closeness centrality**: the inverse of the mean length of the shortest paths between the vertex *v* and all the other vertices that are reachable from *v*.

$$C_c(v) = \frac{|V|-1}{\sum\limits_{i \in V \setminus \{v\}} d_{vi}}$$

where *V* is the vertex set of the connected component that is reachable from vertex *v* in graph $\mathscr{G}$, and d_{vi} denotes the length of the shortest path from vertex *v* to *i*.

In this work, we apply these three centrality measures on the undirected entity-entity graph constructed from the document dataset. The centrality measures for each vertex in the entity-entity graph are calculated using the R package igraph [12], and vertices with large values for each centrality measure are provided to the intelligence analyst for further investigation.

4. MODELING "SURPRISE"

The idea in our newly proposed MTV algorithm is to acknowledge the intrinsically exploratory and iterative nature of analysis, whether by visual methods, machine learning, or traditional statistical methods. That is, what we already know about the data greatly determines our expectations, and therefore, which results we would find interesting and/or surprising. Early on in the process of analyzing a dataset, for instance, we are happy to learn about the generalities underlying the data (e.g., that the bulk of the dataset has to do with financial transactions), while later on we will be more interested in the specifics that build upon these concepts (e.g., that there is suspicious activity happening around mortgage refinancing).

Essentially, this process comes down to summarization: we would like to know what is interesting in the data, and we want this to be reported as succinctly as possible, without redundancy. In a nutshell, that is the approach of MTV algorithm [31]: it incrementally adjusts its model of the data as it iteratively discovers the most informative patterns, in order to obtain a non-redundant summary of the data.

4.1 Maximum Entropy Models

To model the data, MTV uses the powerful and versatile class of maximum entropy models. We construct a maximum entropy distribution which allows us to directly calculate the expected frequencies of entities and combinations thereof. Then, every iteration, we employ a convex heuristic to efficiently find that entity combination that provides the most new information, i.e., for which our frequency estimate was most off. We update our model with this new knowledge, and continue the process.

By the Bayesian Information Criterion (BIC) we can automatically identify the most informative model, and prevent overfitting. The non-redundant model that contains the most important information is thus automatically identified. As such, MTV can tell you what you need to know about the data.

4.2 Formalisms

The Maximum Entropy principle [13, 23] is a general technique for forming a distribution from statistics. The approach has many virtues: the resulting distribution captures all the information hidden in statistics and also all the implications. That is, it uses the provided information *optimally*.

For example, consider that we know that whenever *A* occurs, *B* occurs as well, and whenever *B* occurs, then *C* occurs. If we apply the maximum entropy approach to this knowledge, not only we capture the dependencies between *A* and *B*, and *B* and *C*, but we also capture automatically the fact that by transitivity whenever *A* occurs, *C* occurs. On the other hand, if the knowledge at hand allows us some freedom in selecting the distribution, we always try to be as unbiased as possible. For example, if we have no knowledge, then the resulting maximum entropy distribution will be the uniform distribution. If, on the other hand, we know individual means of the attributes, but not their correlations, then the resulting distribution will be the independence distribution.

More formally, a statistic is a function $S(d)$ that maps a single document to a real number. Given a statistic *S* and a document dataset *D* we compute the average of *S* over *D*, $f = \frac{1}{|D|}\sum_{d \in D} S(d)$. Consider that we have *k* different statistics $S_1, \ldots, S_k$ and that we have computed f_i, an average for each S_i. We want to build a distribution that would capture all the information represented by (S_i, f_i) in the most unbiased way. In order to do so, let us consider all possible distributions defined over the sample space.

In order to capture the information in the statistics, let us constrain ourselves to those distributions that produce the same statistics. That is, a distribution p must satisfy $\mathrm{E}_p[S_i] = f_i$ for $i = 1, \ldots, k$. Such a restriction rarely yields a single distribution, so from this collection we have to choose one. Here we employ the Maximum Entropy principle, namely, we select a distribution maximizing entropy $-\sum_d p(d) \log p(d)$, where the sum is taken over the whole sample space.

A celebrated theorem [13] states that the maximum entropy distribution has a particular form, namely that it can be written as $\log p(d) = r_0 + \sum_{i=1}^{k} r_i S_i(d)$. Such model is sometimes referred as exponential family or log-linear model. Discovering the parameters is typically done by Iterative Scaling [13, 14], where a single r_i is updated in turn, while the rest of the parameters are kept constant.

4.3 Maximum Entropy Model for Entity Sets

The MTV algorithm [31] uses sets of entities, or entity-sets, and their supports to describe and summarize a collection of documents. That is, the summary consists of a collection of k informative combinations of entities $\mathscr{C} = \{X_1, X_2, \ldots, X_k\}$ and their respective frequencies $\{f_1, f_2, \ldots, f_k\}$, where the frequency f_i of an entity-set X_i is defined as the relative number of documents in D that contain X_i. Using the Maximum Entropy principle, a probabilistic model p of the dataset is constructed, satisfying the frequency constraints in $\mathscr{C}$. Out of all possible distributions satisfying these frequencies, p uses the available information in an unbiased way, and furthermore maximizes the likelihood. Using the terminology above, the statistics used in this context are quite straightforward: $S_i(d) = 1$ if $X_i \subset d$ and $S_i(d) = 0$ otherwise. Hence, the log probability $\log p(d)$ of a single document d can simply be computed as the sum of an appropriate subset of the parameters r_i.

Ideally, a summary $\mathscr{C}$ should contain only patterns that are truly informative with respect to the document dataset, not contain any redundant information, and at the same time not contain too many patterns, since the analyst can only examine a limited number of them. The quality of any given summary is evaluated using the Bayesian Information Criterion (BIC) [35], defined as

$$\mathrm{BIC}(p, D) = -\log p(D) + \frac{k}{2} \log |D|, \tag{6}$$

where p is the maximum entropy distribution and k the number of parameters of the model, which equals the number of entity-sets we have added to the model. The first term rewards summaries that achieve high likelihood, i.e., that contain informative patterns that model the data well. The second term penalizes summaries that are overly complex, i.e., that contain too many (redundant) entity-sets. Hence, BIC rewards summaries that model the data as well as possible, using as few patterns as possible. Due to this, redundant patterns are automatically avoided. Consider the following example, with $\mathscr{C} = \{A, B, AB\}$, where the frequencies of A and B are 20% and 30% respectively. If the frequency of AB in the data is roughly 6%, this entity-set is redundant with regard to what we already know, and hence the BIC score will be high. On the other hand, if AB's frequency equals, say, 18%, then A and B co-occur more often than expected, and hence AB can be considered surprising and interesting.

4.4 Searching for models

To discover a good summary of the data D, the MTV algorithm employs an incremental, greedy approach. The model is initialized using the frequencies of the individual entities; under MaxEnt, p then simply corresponds the independence model. The algorithm starts from the empty set, and iteratively adds an entity-set to minimize the BIC score of model, i.e., $\arg\min_X BIC(\mathscr{C} \cup \{X\})$. To find this entity-set, a heuristic is used:

$$\arg\max_X h(fr(X), p(X)) \tag{7}$$

where $h(a, b) = a \log \frac{a}{b} + (1 - a) \log \frac{1-a}{1-b}$, $fr(X)$ is the frequency of X in the dataset, and $p(X)$ is the frequency of X as predicted by the model. Intuitively, h measures the surprisingness or interestingness of the frequency of X with respect to its expected value according to the current model. If $fr(X) = p(X)$ then it equals 0; the more $fr(X)$ deviates from $p(X)$, the higher it becomes. Since this heuristic is convex, mining the combination of entities that minimizes the score can be done efficiently. Next, the pattern maximizing h is added to $\mathscr{C}$, and the distribution p is updated accordingly to reflect the newly gained information, using the Iterative Scaling procedure. This process is repeated as long as the BIC score improves.

Algorithm 3 MTV algorithm

```
initialize p with the independence model
𝒞 ← ∅
while BIC(p, D) decreases do
    X ← FindMostSurprisingEntityset(p, D)
    𝒞 ← 𝒞 ∪ {X}
    p ← IterativeScaling(𝒞)
    compute BIC(p, D)
end while
return 𝒞
```

4.5 Incorporating Prior Knowledge

As stated, it is possible to make the algorithm take background knowledge into account. This can be done in the form of entity-sets and their frequencies. In its most basic form, the algorithm always includes background knowledge in the form of individual entity frequencies. However, it is possible to initialize the model with additional information, for instance, if the analyst already knows that "new" and "york" occur together frequently, the algorithm won't report this as surprising when it is informed of this fact.

4.6 Analyst-Guided Exploration

Furthermore, it is possible to use the MTV algorithm in an interactive fashion. Rather than having it provide the analyst with a fixed set of patterns, the analyst can guide the search process. This can be done by, in each step, letting the user inspect a list of potentially interesting entity-sets from which one is then chosen and added. Making the discovery process more interactive in this way, allows the analyst to guide the search for patterns in a direction that is more interesting to him.

5. EXPERIMENTS

In this section, we present some results of the proposed algorithms on five intelligence analysis datasets.

5.1 Datasets

The five intelligence datasets we used to evaluate the algorithms will be referred to as Crescent, Manpad, AltanticStorm, VAST10, and VAST11. The scenarios and tasks of these intelligence datasets are to investigate and reveal the plots of any imminent threat, arms dealing, or possible terrorist attacks. The sizes of these five intelligence datasets range from small (less than 50 documents) to large (more than 4,000 documents), which are summarized in Table 1. The number of entities and documents participating in the correct solution are also tabulated here.

To apply most of the algorithms described earlier on these intelligence analysis datasets, some preprocessing steps are undertaken.

Table 1: Dataset Statistics.

Dataset	Number of Documents	Number of Entities	Number of Documents in Solution	Number of Entities in Solution
Crescent	41	284	24	20
Manpad	47	143	16	16
AtlanticStorm	111	716	55	43
VAST10	103	621	50	20
VAST11	4470	55106	15	10

First, co-reference techniques are applied to the intelligence documents in each dataset to eliminate the duplicate entities that refer to the same object. Then, we extract all the entities from each dataset using the tool of AlchemyAPI. Typically, the entities extracted are nouns that refer to the names of persons, organizations, cities, etc., numbers and dates. Finally, each entity and each document is assigned a unique ID within the dataset, respectively. After these preprocessing steps, each document in a dataset can be represented as a collection of entities it contains, and the whole dataset can be represented as the collection of documents. Table 1 shows some important statistics for each dataset.

5.2 Results Summary

We first outline the steps we have undertaken for each of the classes of methods. For the association measure-based methods, we sort the discovered association rules in decreasing order according to the selected association measures for each dataset. We then collect the entities from the association rules that are ranked on top until the number of the entities we have collected is equal to the size of the entity set in the solution of the dataset (this size threshold was put in place to ascertain whether the methods are capable of identifying all important players). Finally, a collection of documents that contain these collected entities is presented to the user.

For the graph-based methods, the final result is constructed in different ways. With the entity-entity undirected graph, we collect all the entity vertices in the dense subgraph output by the greedy approximation algorithm described in Algorithm 1 as the entity set, and all the documents that contain these entity vertices as the document set of the result. With the document-entity bipartite graph, since the heuristic local search algorithm we implemented allows us to specify the size of the initial bipartite subgraph, we set the initial size of the bipartite subgraph the same as the size of the solution of each dataset. Naturally, the entity vertex set and document vertex set in the bipartite subgraph output by the local search algorithm will constitute the result. In the graph centrality based methods, the top N entity vertices consist of the entity set in the final result based on the chosen centrality measures, where N is the size of the entity set in the solution of each intelligence analysis dataset. The document set is just the collection of documents that contain any of these top N entities.

For the MTV algorithm, we formulate the entity set in the result by collecting the entities from the top summary itemsets output by the MTV algorithm until the size of the entity set is equal to that in the solution of each intelligence analysis dataset. The document set in the result is again the collection of documents that contain any of the collected entities in the entity set.

We presented the experiment results of these methods to an intelligence analyst who is familiar with the five datasets we used, and requested an evaluation of the quality of these results based on the solution and the scenarios of each intelligence analysis dataset. Table 2 depicts the scores (range from 0 to 10, where higher score indicates better quality of the results that either have more entity overlap with the solution or contain very important key entities of the solution), given by the intelligence analyst, of each method on the intelligence analysis dataset Crescent, Manpad, AtlanticStorm and VAST10. c, G and ϕ in the table represent the association measures of Confidence, Gini index and ϕ-coefficient, respectively.

Table 3: Entity sets extracted from AtlanticStorm by MTV algorithm.

{Derwish, Windsor, Ontario, Canada}
{Omar Hanif, Bahamas, Nassau}
{Los Andes Hotel, Mexico, Bogota, Tampico, in, Columbia}
{Cuba, Havana}
{Casablanca, Holland Orange, Holland Orange Shipping Lines}
{Central Russia, Moscow, Central Russia Airlines}

From Table 2, we can see that the association measure based method works best on the AtlanticStorm dataset and fairly for the other datasets. The dense entity-entity and document-entity subgraph methods have a good performance on the Crescent dataset, but do not work well for the rest of the three datasets. Moreover, for the dense entity-entity subgraph method, one drawback of this method is that we cannot control the size of the dense subgraph. For example, although the result of this method contains nearly 50% of the important entities in the solution and has a high intelligence analyst assigned score, it also includes 181 irrelevant entities, which makes it difficult to identify the important entities in the plot. The graph centrality measure based method performs fairly on the Crescent, Manpad and AtlanticStorm datasets, but a little poorly on the VAST10 dataset. The MTV algorithm performs best on all datasets except the Manpad dataset.

Here, we use the MTV algorithm as an example to illustrate how these algorithmic strategies guide the analysis of intelligence analysts. Table 3 shows the entity sets extracted by MTV from the AtlanticStorm dataset. The intelligence analyst reads the documents that contain these entities, and finds that the entity sets {*Omar Hanif, Bahamas, Nassau*}, {*Casablanca, Holland Orange, Holland Orange Shipping Lines*} and {*Central Russia, Moscow, Central Russia Airlines*} in the result appear to be interesting. The intelligence analyst may also identify some other entities that have connections with these entity sets that could potentially be interesting. These interesting and uninteresting entities can be supplied back to the MTV algorithm as feedback. With the interesting entities in hand, the intelligence analyst can use them as starting points, and apply any existing tools discussed in section 2 to investigate the entire dataset and unravel the plot.

We did not list the scores of the algorithms on the VAST11 dataset in Table 2 because none of these methods captured the structure of the plot in the VAST11 dataset, and thus have trouble in generating starting points for intelligence analysts. Recall that one basic way to approach the information discovery process is to categorize the given documents into different topics and use established document classification techniques to obtain a first cut understanding of the collection. So, in addition to the algorithmic strategies we have surveyed, we also applied document classification techniques to the VAST11 dataset. The tool of AlchemyAPI is adopted to achieve the document classification objective. The AlchemyAPI implements statistical NLP algorithms and machine learning algorithms to analyze the content of the documents and extract useful semantic information from the documents provided [1]. We first invoke the text categorization API to identify the most likely topical distribution of the documents and then classify the documents in the dataset into different classes according to their identified topics. Among all the document categories generated by AlchemyAPI on the VAST11 dataset, there are categories *crimes and law* and *unknown*, which contain 122 documents altogether. The category of *crimes and law* conveys some information about the VAST11

Table 2: Experimental evaluation results.

Dataset	Association measures			Dense entity-entity subgraph	Dense document-entity subgraph	Graph centrality measures			MTV
	c	G	ϕ			Degree	Betweenness	Closeness	
Crescent	7	7	7	8	8	7	6	6	9
Manpad	8	7	7	6	0	6	7	9	0
AtlanticStorm	9	9	9	2	7	7	7	7	9
VAST10	6	6	6	5	2	2	5	5	8

Table 4: Entities extracted from IslamicAwakening.

{Afghanistan, Taliban}
{Gaza Strip, Hamas}
{Osama bin Laden, Afghanistan}
⋮

dataset (recall that the task of this dataset is to find any imminent threat) and the *unknown* category contains more vague documents that need to be further investigated. More important, document classification greatly reduces the number of documents the intelligence analyst needs to inspect initially, and these 122 documents also contain the part of the documents in the solution of VAST11. If we extract the key entities in the solution documents that are classified into *crimes and law* and *unknown* categories, they are very close (2 or 3 hops) to the rest of the important entities in the solution of VAST11 in the entity-entity graph. These results imply that for large datasets, such as VAST11, incorporating some preprocessing steps, such as document classification, may facilitate the process of generating starting points for intelligence analysts.

5.3 Discussion

Our experimental results on five intelligence analysis datasets indicate that all of the algorithmic strategies presented in this paper have the ability to generate some meaningful guidance to intelligence analysts. In this section, we will discuss the possible reasons why some strategies perform well on some datasets, but not others.

One common idea behind the association measures, graph measures, and MTV algorithm is that all of them aim to identify entities that are most frequent (important) or central to the dataset. When applied over the five datasets, however, there are clear selective superiorities that emerge. In Crescent, Manpad, AtlanticStorm and VAST10 datasets, around 50% documents contain the interesting entities in the solution. Thus, some of the entities in the solution can be considered globally frequent (important) or central to the dataset. This explains why these methods can retrieve many of the interesting entities with only a few exceptions, such as the dense document-entity subgraph approach or the MTV approach on the Manpad dataset. However, in the VAST11 dataset, the interesting entities in the solution are focused on a very small portion of documents, viz. 15 documents compared to a total of 4470 documents in the VAST11 dataset, and in this case, the entities in the solution can not be considered as globally frequent or central to the dataset. Thus, without any pre-processing, these methods do not fare well on the VAST11 dataset.

With the document classification approach, the AlchemyAPI classifies the documents using sophisticated statistical algorithms, whose distributions are inferred from massive troves of general documents. As a result, it can only classify documents into very general topics, thus its use is limited when using alone. However, as a preprocessing step, document classification approach can reduce the number of documents that needs to be investigated initially by intelligence analysts, and when incorporating with other methods, it may facilitate the process of generating starting points of intelligence analysis for analysts.

Table 5: Entity sets extracted after document clustering.

{Bismillah Walhamdulillah, Al-Qaida, Rasulillah Assalamu, WaAlaikum Salam, Dr. Ayman Zawahiri, Pakistan}
{Beitullah Masud, Mullah Fazl., Mangal Bagh, Taliban}
{Al-Qaida, CAIRO, Osama bin Laden, Egypt, Ayman al-Zawahri}
⋮

Thus, one lesson from our studies has to do with how "broadly" the hidden plot is woven across the document collection, or whether it is concentrated over a minority of instances. The MTV algorithm fares well when the entities to be discovered have frequencies anomalous to a maximum entropy estimate, but can get sidetracked e.g., in the case of the Manpad dataset, where it emits one entity *FBI* as the output. Deleting such entities will generally improve the results of MTV.

5.4 Results on ISI-KDD'12 Challenge Dataset

We also applied the MTV algorithm to the ISI-KDD 2012 challenge dataset *IslamicAwakening*, which comes from the Dark Web Portal of several complete multi-year extremist forums (Notice that we did not aim to solve the challenge tasks with these algorithmic strategies, which is not the goal of this paper). Each posting in this dataset is treated as a separate document in our analysis. Table 4 lists some of the entity sets in the *IslamicAwakening* dataset provided by the MTV algorithm. From Table 4, we can see that some of these entities are the names of well known terrorism organizations and the regions involved. Besides these well known associated entities, we also tried to further investigate the dataset and find other interesting entities that can serve as starting points for intelligence analysis.

We clustered the *IslamicAwakening* dataset into several groups, such that within each group, the documents have at least one common entity. Table 5 shows part of the entity sets generated by the MTV algorithm on these groups of documents, which reveal connections between people and organizations. These entities might be interesting to the intelligence analyst. Starting from these entities, it is easy to identify the postings that contain these entities and the related threads or members in the extremist forums. The forum members that talked a lot about these terrorism organizations or the persons that have connections with the terrorism organizations might be radical persons, and intelligence analysts may be interested in further investigating these postings and forum members.

6. CONCLUSION AND FUTURE WORK

In this paper, we have explored several existing categories of algorithmic strategies, including association measures, graph metrics, and probabilistic modeling using the maximum entropy principle, to guide intelligence analysis. Our results applying these strategies to five datasets indicate selective superiorities based on small vs. large datasets and broad vs. focused plots, and the MTV algorithm performs generally better than other methods on the intelligence datasets across which the hidden plots is broadly woven. But pre-processing steps such as document classification are necessary

to address larger document collections. As part of our future work, we will continue to explore new mechanisms and frameworks to support intelligence analysts in unarticulated aspects of investigation.

7. ACKNOWLEDGEMENT

This work is supported in part by the US National Science Foundation through grant CCF-0937133.

8. REFERENCES

[1] AlchemyAPI: Tranforming Text into Knowledge, 2012. http://www.alchemyapi.com/.

[2] R. Alonso and H. Li. Model-guided Information Discovery for Intelligence Analysis. In *CIKM '05*, pages 269–270, 2005.

[3] E. Bier, S. Card, and J. Bodnar. Principles and Tools for Collaborative Entity-Based Intelligence Analysis. *TVCG*, 16(2):178–191, 2010.

[4] E. Bier, E. Ishak, and E. Chi. Entity Workspace: An Evidence File That Aids Memory, Inference, and Reading. In *ISI '06*, pages 466–472, 2006.

[5] A. L. Buczak and C. M. Gifford. Fuzzy Association Rule Mining for Community Crime Pattern Discovery. In *ACM ISI-KDD '10*, pages 2:1–2:10, 2010.

[6] M. Charikar. Greedy Approximation Algorithms for Finding Dense Components in a Graph. In *APPROX '00*, pages 84–95, 2000.

[7] M. Chau, J. J. Xu, and H. Chen. Extracting Meaningful Entities from Police Narrative Reports. In *Annual National Conference on Digital Government Research*, pages 1–5, 2002.

[8] G. Chin, O. A. Kuchar, P. D. Whitney, M. Powers, and K. E. Johnson. Graph-based Comparisons of Scenarios in Intelligence Analysis. In *SMC '04*, volume 4, pages 3175–3180, 2004.

[9] K. Chopra and C. Haimson. Information Fusion for Intelligence Analysis. In *HICSS '05*, page 111.1, 2005.

[10] T. Coffman, S. Greenblatt, and S. Marcus. Graph-based Technologies for Intelligence Analysis. *Commun. ACM*, 47:45–47, March 2004.

[11] P. Crossno, B. Wylie, A. Wilson, J. Greenfield, E. Stanton, T. Shead, L. Ice, K. Moreland, J. Baumes, and B. Geveci. Intelligence Analysis Using Titan. In *IEEE VAST*, pages 241–242, 2007.

[12] G. Csardi. igraph: Network Analysis and Visualization, Reference Manual, December 2011. http://cran.r-project.org/web/packages/igraph/index.html.

[13] I. Csiszár. I-Divergence Geometry of Probability Distributions and Minimization Problems. *The Annals of Probability*, 3(1):146–158, Feb. 1975.

[14] J. Darroch and D. Ratcliff. Generalized Iterative Scaling for Log-Linear Models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[15] FMS Advanced Systems Group, FMS Inc. Sentinel Visualizer. Last accessed: May 26, 2011, http://www.fmsasg.com/.

[16] J. Gersh, B. Lewis, J. Montemayor, C. Piatko, and R. Turner. Supporting Insight-based Information Exploration in Intelligence Analysis. *Commun. ACM*, 49, April 2006.

[17] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: Visualizing Thematic Changes in Large Document Collections. *TVCG*, 8(1):9–20, 2002.

[18] M. S. Hossain, C. Andrews, N. Ramakrishnan, and C. North. Helping Intelligence Analysts Make Connections. In *AAAI'11, Worskshop on Scalable Integration of Analytics and Visualization*, pages 22–31, 2011.

[19] M. S. Hossain, P. Butler, A. P. Boedihardjo, and N. Ramakrishnan. Storytelling in Entity Networks to Support Intelligence Analysts. In *KDD '12*, To appear, 2012.

[20] M. S. Hossain, J. Gresock, Y. Edmonds, R. Helm, M. Potts, and N. Ramakrishnan. Connecting the Dots between PubMed Abstracts. *PLoS ONE*, 7(1):e29509, 2012.

[21] i2group. The Analyst's Notebook. Last accessed: May 26, 2011, http://www.i2group.com/us.

[22] Information Interfaces Research Lab., Georgia Tech. Jigsaw: Visual Analytics for Exploring and Understanding Document Collections. Last accessed: February 9, 2012, http://www.cc.gatech.edu/gvu/ii/jigsaw/.

[23] E. Jaynes. On the Rationale of Maximum-Entropy Methods. *PIEEE*, 70(9):939–952, 1982.

[24] H. Kang, C. Plaisant, B. Lee, and B. B. Bederson. NetLens: Iterative Exploration of Content-actor Network Data. *Info. Vis.*, 6(1):18–31, 2007.

[25] H. Khurana, J. Basney, M. Bakht, M. Freemon, V. Welch, and R. Butler. Palantir: A Framework for Collaborative Incident Response and Investigation. In *IDtrust*, 2009.

[26] A. Koltuksuz and S. Tekir. Intelligence Analysis Modeling. In *ICHIT '06*, 2006.

[27] D. Kumar, N. Ramakrishnan, R. F. Helm, and M. Potts. Algorithms for Storytelling. *IEEE Trans. on Knowl. and Data Eng.*, 20(6):736–751, June 2008.

[28] R. Kumar, U. Mahadevan, and D. Sivakumar. A Graph-Theoretic Approach to Extract Storylines from Search Results. In *ACM KDD '04*, pages 216–225, 2004.

[29] S. Lin and D. E. Brown. Criminal Incident Data Association Using the OLAP Technology. In *ISI'03*, pages 13–26, 2003.

[30] E. Lindahl, S. O'Hara, and Q. Zhu. A Multi-agent System of Evidential Reasoning for Intelligence Analyses. In *AAMAS '07*, pages 279:1–279:6, 2007.

[31] M. Mampaey, N. Tatti, and J. Vreeken. Tell Me What I Need to Know: Succinctly Summarizing Data with Itemsets. In *ACM KDD '11*, pages 573–581, 2011.

[32] A. Noori. On the Relation between Centrality Measures and Consensus Algorithms. In *HPCS, 2011*, pages 225 –232, july 2011.

[33] P. Pirolli and S. Card. The Sensemaking Process and Leverage Points for Analyst Technology as Identified through Cognitive Task Analysis. In *ICIA '05*, 2005.

[34] PNNL. Pacific Northwest National Laboratory, INSPIRE Visual Document Analysis. Last accessed: May 26, 2011, http://in-spire.pnl.gov.

[35] G. Schwarz. Estimating the Dimension of a Model. *Annals of Statistics*, 6(2):461–464, 1978.

[36] D. B. Skillicorn. Applying Interestingness Measures to Ansar forum Texts. In *ACM ISI-KDD '10*, pages 7:1–7:9, 2010.

[37] A. Sun, M.-M. Naing, E.-P. Lim, and W. Lam. Using Support Vector Machines for Terrorism Information Extraction. In *ISI'03*, pages 1–12, 2003.

[38] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. In *ACM KDD '02*, pages 32–41, 2002.