Extracting Statistical Data Frames from Text

Jisheng Liang, Krzysztof Koperski, Thien Nguyen, and Giovanni Marchisio

Insightful Corporation

1700 Westlake Ave N, Suite 500

Seattle, WA 98109

[jliang, krisk, thien, giovanni]@insightful.com

ABSTRACT

We present a framework that bridges the gap between natural language processing (NLP) and text mining. Central to this is a new approach to text parameterization that captures many interesting attributes of text usually ignored by standard indices, like the term-document matrix. By storing NLP tags, the new index supports a higher degree of knowledge discovery and pattern finding from text. The index is relatively compact, enabling dynamic search of arbitrary relationships and events in large document collections. We can export search results in formats and data structures that are transparent to statistical analysis tools like S-PLUS®. In a number of experiments, we demonstrate how this framework can turn mountains of unstructured information into informative statistical graphs.

Keywords

Text mining, natural language processing, NLP, statistical data frames, data analysis, visualization

1. INTRODUCTION

Human language is one of the most complex processes that data miners and statisticians have ever attempted to model. Yet, many commercial applications for text mining, categorization and search today rely on very simple parameterizations of text content. The most common example of a text index is the term document matrix, which captures word frequency distributions in a corpus of documents. This index representation does not do justice to the complexity of human language, but is dictated by the practical difficulty of storing more informative objects, such as constituency or dependency trees, in a compact and searchable form.

In this paper we demonstrate how an indexing framework that includes NLP tags can benefit text mining and greatly increase the power of information discovery from text. The text mining and computational linguistic communities have coexisted for many years with limited interaction [1]. The aim of text mining is similar to data mining in that it attempts to analyze texts to discover interesting patterns such as clusters, associations, deviations, similarities, and differences across large collections of text documents [3]. Data mining researchers come from one of the statistics, database and machine learning communities. They are more familiar with structured data representations such as relational tables or statistical data frames. Often, the starting point for their analyses is a simple and shallow representation of text [11] that does not use NLP tags. For instance, many machine learning models can be trained on simple text attributes, such as word co-occurence, proximity, sequence or order.

The potential to apply NLP or computational linguistics to obtain a richer representation of text has existed for decades [2]. However, NLP tends to focus on one document or piece of text at a time and be rather computationally expensive. It includes techniques like lexical analysis, multiword phrase grouping, sense disambiguation, part-of-speech tagging, anaphora resolution, and role determination. The arguments against NLP are that it is error-prone, and NLP output (i.e. parse trees) contains too much linguistic detail, noise and uncertainty to provide a working knowledge base for data analysis or mining. Failure to account for semantic and syntactic variations across a document collection has led to disappointing results when trying to use fine indexing structures derived from a linguistic parser.

Information Extraction (IE) is a methodology employed as a precursor to text mining especially in bioinformatics [4][5]. IE applies NLP techniques to extract predefined sets of entities, relationships, and patterns of interest from documents. IE systems, like those developed in the MUC [6] and ACE [7] programs, are limited in their power of information discovery. First, they employ pre-determined templates or rule sets. Second, they do not index everything in a corpus, but only what they are preprogrammed to find. The aim of text mining should be to find novel patterns rather than predefined ones. In addition, because of the time involved in reprogramming extraction rules and reprocessing large volumes of text, it is difficult to iteratively apply mining processes to the output of IE tools [12].

We have developed a proof of concept based on a new system for text analysis and search called InFact. The major innovation in InFact is a new indexing process that combines the flexibility and efficiency of keyword indexing with the knowledge of grammatical or syntactic roles. A major difference between our approach and other existing technologies for IE is that, during indexing, we are not restricted by any rigid, pre-determined templates or patterns. InFact indexes every clause of every sentence of every document in a large corpus, whether or not the clause contains certain named entities or trigger words. It follows that we can interactively query a large database of text documents for all kinds of relationships and events. We can also output all search results into formats and data structures that are transparent to statistical analysis tools such as S-PLUS®.

The remainder of this paper is organized as follows. Section 1 presents an overview of our indexing approach. Section 3 describes the data frame export module. Section 4 demonstrates data analyses on a variety of extracted data frames. These include:

- Clustering and visualization of event patterns
- Visualization and analysis of time-series of events with Trellis Graphics®
- Latent Semantic Analysis (LSA) and cross-document clustering of entities, based on their syntactic roles.

2. INFACT SYSTEM OVERVIEW

InFact consists of an indexing and a search module. With reference to Figure 1, indexing pertains to the processing flow on the left of the diagram, while search is on the right. InFact models text as a complex multivariate object using a unique combination of deep parsing, linguistic normalization and efficient storage. The storage schema addresses the fundamental difficulty of reducing information contained in parse trees into generalized data structures that can be queried dynamically. In addition, InFact handles the problem of linguistic variation by mapping complex linguistic structures into semantic and syntactic equivalents. Our long-term objective is to provide a unified knowledge representation for structured and unstructured data sources. This representation should supports dynamic relationship and event search, information extraction and pattern matching from large document collections in real time.

2.1 Indexing

Unlike traditional models that store only word distributions, InFact performs clause level indexing and captures syntactic roles, constructs, relationships, and inter-clause relationships that enable it to understand events. Another strong differentiator of our approach is that we create these indices automatically, without using predefined extraction rules, and we capture all information, not just predefined patterns. Our parser performs a full constituency and dependency analysis, extracting part-of-speech (POS) tags and grammatical roles from every clause.

The raw output of a deep parser usually does not lead to a tractable indexing schema for cross-document analysis. Therefore, we have developed a set of rules for converting an augmented tree representation into a scalable data storage structure. The basic idea involves collapsing selected nodes in the parse tree to reduce the overall complexity of the dependency structures. The result is a set of inter-linked subject-action-object triplets, as shown in Figure 2. The subject-action-object triples can express most types of syntactic relations between various entities within a sentence. With this parameterization in mind, we apply transformational rules (e.g. transformational grammar) that encode specific linguistic variations, and equate structures derived from different surface forms. We apply normalization rules at the syntactic, semantic, or even pragmatic level.

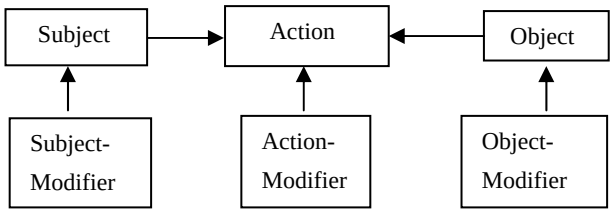

Figure 2: the Subject-Action-Object indexing structure

InFact stores the normalized triplets into a proprietary, patent-pending index that:

- organizes events, relationships and modifiers across documents
- optimizes efficient retrieval of events
- stores metadata and additional ancillary linguistic variables
- (optionally) superimposes taxonomical dependencies from a custom ontology or knowledge base

The InFact index stores “soft events” instead of fitting textual information into a rigid relational schema that may result in information loss. “Soft events” are data structures that can be

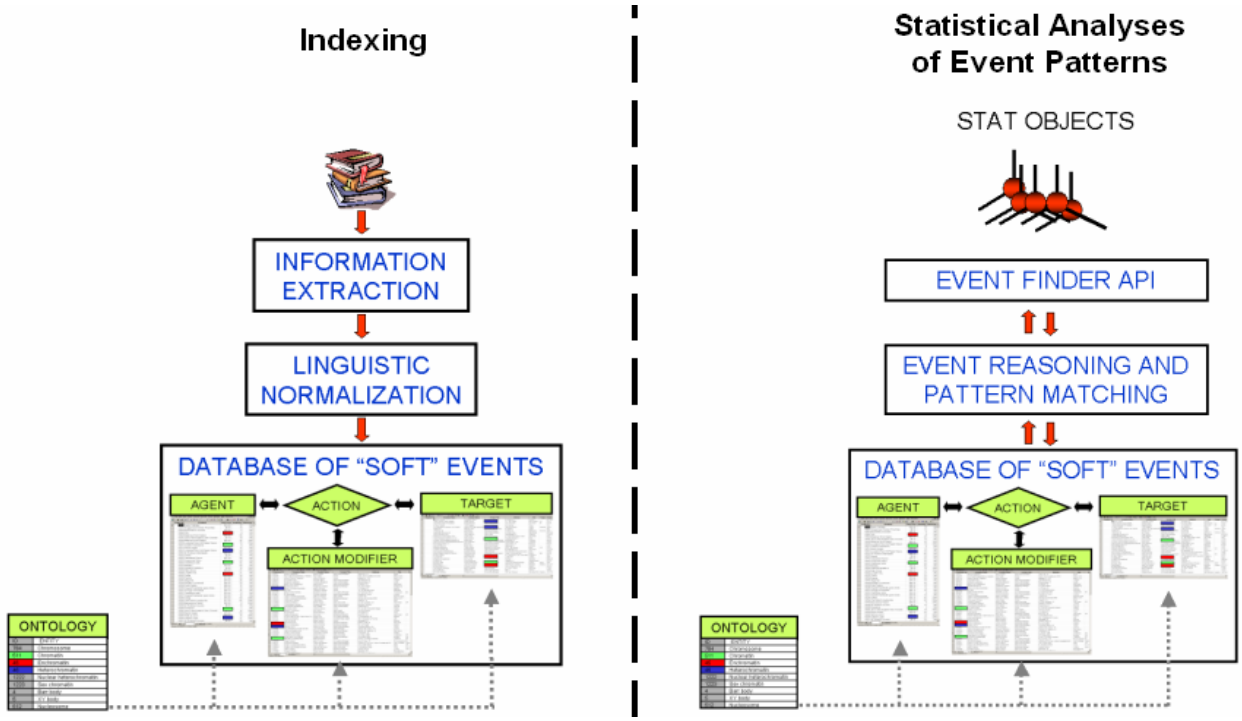

Figure 1: our text-to-data framework

recombined to form events and relationships. "Soft events" are pre-indexed to facilitate thematic retrieval by action, subject, and object type. For instance, a sentence like "The president of France visited the capital of Tunisia" contains evidence of 1) a presidential visit to a country's capital and 2) diplomatic relationships between two countries. Our storage strategy maintains both interpretations. In other words, we allow more than one subject or object to be associated with the governing verb of a sentence. The tuples stored in the database are therefore "soft events", as they may encode alternative patterns and relationships found in each sentence. Typically, only one pattern is chosen at search time, in response to a specific user request (i.e. request #1: gather all instances of "*president > visit > capital*"; request #2: gather all instances of "*[Country] <> any_action <> [Country]*").

2.2 Search

Unlike keyword search engines, InFact employs a highly expressive query language that combines the power of grammatical roles with the flexibility of Boolean operators, and allows users to search for actions, entities, relationships, and events. InFact represents the basic relationship between two entities with an expression of the kind:

Subject Entity > Action > Object Entity,

We can optionally constrain this expression by specifying modifiers or using Boolean logic. The arrows in the query refer to the directionality of the action, which could be either one direction or bi-directional. For example,

Entity 1 <> Action <> Entity 2

will retrieve all relationships involving *Entity 1* and *Entity 2*, regardless of their roles as subject or object of the action.

With the InFact query language, we can search for:

- Any relationships involving an entity of interest

  For example, the query "*George Bush <> * <> **" will retrieve any events involving "*George Bush*" as subject or object

- Relationships between two entities or entity types

  For example, the query "*China <> * <> Afghan**" will retrieve all relationships between the two countries. Note that wildcard is used in "*Afghan**" to handle different spelling variations of Afghanistan. The query "*Bin Laden <>*<> [Organization]*" will retrieve any relationships involving "Bin Laden" and an organization.

- Events involving one or more entities or types

  For example, the query "*Pope > visit > [country]*" will return all instances of the Pope visiting a country. In another example, "*[Organization] > acquire > [Organization]*" will return all events involving one company buying another one.

- Events restricted to a certain action type

  "Action type" is a group of semantically linked actions. For example, query "*[Person] > [Communication] > [Person]*" will retrieve all events involving two people that are in the nature of communication.

Note that in InFact we can represent and organize entity types using taxonomy paths, e.g.:

- *[Entity/Location/Country]*
- *[Entity/Location/City]*

The taxonomic paths can encode "is-a" relation (as in the above examples), or any other relations defined in a particular ontology (e.g. "part-of" relation). When querying, we can use a taxonomy path to specify an entity type, e.g. *[Location/Country]*, *[City]*, *[Location]*, etc., and the entity type will automatically include all subpaths in the taxonomic hierarchy.

InFact's query syntax supports Boolean operators (i.e. AND, OR, NOT). For example, the query:

Clinton NOT Hillary > visit OR travel to > [Location]

is likely to retrieve the travels of *Bill Clinton*, but not *Hillary Clinton*.

We can further constrain actions with modifiers, which can be explicit entities or entity types, e.g. *Paris* or *[location]*. For example, the query

[Organization/Name] > buy > [Organization/Name]^[money]

will only return results where a document mentions a specific monetary amount along with a corporate acquisition. Similarly, in the query

Bush <> meet<> Clinton ^[location]

will return results restricted to actions that occur in an explicit geographical location.

We can also filter search results by specifying document-level constraints, including:

- Document metadata tags – lists of returned actions, relationships or events are restricted to documents that contain the specified metadata values.
- Boolean keyword expressions – lists of returned actions, relationships or events are restricted to documents that contain the specified Boolean keyword expressions.

InFact can also support synonyms and query expansion via custom ontologies. In this case, InFact will automatically recognize the equivalence of entities or actions that belong to the same ontology node.

3. DATA FRAME EXPORT

We provide a Java API to export events and relationships extracted by the Search and Retrieval module into data mining tools like S-PLUS ® and I-Miner®. The data frame export facility converts information from free text to data frames and matrices, on which we can perform statistical data analyses. We support two basic data types:

- Data Frame – a set of observations (rows) on a number of variables (columns)
- Frequency Matrix - each cell (r, c) contains the total number of times that the pair (r, c) co-occurs in the given corpus, and the rows and column are customizable

The data frame export module is flexible and configurable, and allows any combination of entities, actions, types, and document metadata. It is also interactive, and hyperlinks observations to the original source documents for quick validation.

3.1 Data Frames

A Data Frame is a data table that, for each observation (rows), stores a list of variables or attributes (columns). In our case, observations are relationship tuples. That is, each observation is an InFact relationship or event, and the variables could be any properties associated with the particular relationship or event, like date. There are two steps in exporting relationship-based data frames:

1. Perform an InFact search to select a set of relationships to export. This is done through the InFact search operators at either the API or UI level.
2. Select the set of properties to export for each relationship tuple.

We can add a variable by specifying:

- Role constraints - the element must have one of the following roles: Subject, Object, Action, Subject Modifier or Object Modifier or Action modifier
- Type constraints – the element must have the specified entity type or action type
- Regular Expression constraints – the element must satisfy a regular expression

We can also select any subset of document metadata tags to export as document attributes.

For example, we may search *"[Organization/name] > purchase > [Organization/name] ^[Money] OR [Date]"* and select the following 5 variables to export for each event InFact finds:

	Variable Name	Role Constraint	Type Constraint	Format
V1	Buyer	Subject	[organization/name]	String
V2	Target	Object	[organization/name]	String
V3	Money-amount	Action-modifier	[money]	Number
V4	Date	Action-modifier	[date]	Date
V5	Creation-Country	Document Creation-Country Metadata		String

In another example, we may want to find all interactions between Al Qaeda and any particular person:

Query: " Al Qaeda <> * <> [Person/Name]"

and select variables as listed below:

	Variable-name	Role Constraint	Type Constraint	Format
V1	Person	Subject OR Object	[person/name]	String
V2	Action-type	Action		Category
V3	Document Date	Document Creation-Date Metadata		Date
V4	Doc URL	Document URL Metadata		String

3.2 Frequency Matrices

A frequency matrix is a two-dimensional array of data of the same mode. The simplest example is a document-term matrix, where the rows are documents, the columns are terms, and the cells are frequency counts of document-term pairs. InFact goes far beyond conventional term-document matrices. For example, a user of InFact can produce a person-action matrix, where the rows are all persons in a corpus that are the subject of an action, the columns are all actions, and the cell values are the frequency counts of a person performing a particular action. In other words, we extend the classic term-document indexing framework of information retrieval to index matrices that have a notion of entities and syntactic roles, such as entity-entity, and action-entity matrices. We generate frequency matrix, where each cell (r, c) contains the total number of times that the pair (r, c) co-occurs in the corpus. Given a frequency matrix, it is straightforward to apply any weighting scheme, such as the standard tf.idf function that is commonly used in information retrieval and text categorization.

To create a frequency matrix, we:

1. Choose two variables as rows and columns. This step is the same as specifying the Data Frame variables to export, except the number of variables is now 2. For example, we can set Row = Subject and Column = Actions, or Row = (Subject OR Object) and Column = Document ID.
2. Extract data frames that satisfy the conditions set in Step 1.
3. Iterate through the extracted data frames. For each pair of variables, update the count in the corresponding matrix cell.

Example 1- Document-Entity Matrix:

In a document-entity matrix, the row must be documents (i.e. Document IDs), and the column could be one of the following:

- Entity Terms. We set the terms to be certain entity types, e.g. ([Location] OR [Organization] OR [Person]), satisfying certain grammatical roles (e.g. subject, object)
- Action Terms. We set the actions to be certain action types, e.g. [Motion] OR [Communication].
- Entity Types. We could simply compute frequency counts of Entity Types in each document
- Action Types. We could simply compute frequency counts of Action Types in each document

To create a document-entity matrix, we:

1. Select a set of documents by executing an InFact query, like a document keyword search with metadata constraints (e.g. "*Iraq after 1992"*). The default is to select all documents in the given corpus.
2. Specify the entity/action terms or types (column type)
3. For each document in the set, find all relationships that contain the entity/action terms or types. Produce a frequency count vector for each document. Append the vector to the document-entity matrix.
4. Output the matrix, the list of rows (document IDs), and the list of columns (entity/action terms or types)

In this mode, a single document becomes effectively a bag of structural relationships or even events that can be modeled and linked across a collection.

Example 2 - Entity-Entity Matrix:

To create an entity-entity matrix, we:

1. Enter an InFact query to retrieve a set of relationships.
2. Specify any two variables as rows and columns.
3. Iterate through the extracted data frames. For each pair of variables, update the count in the corresponding matrix cell.

The table below shows a person-action frequency matrix, where each row is a person entity and columns are actions performed by each given person, such as

- Row: Role= Subject, and Entity Type=Person
- Column: Role=Action

Elements in the resulting person-action matrix represent co-occurrences of persons and actions.

	Action 1	Action 2	Action 3
Person 1	2	1	
Person 2		1	1
Person 3	2		
Person 4		1	
Person 5			2

4. DATA ANALYSES

We describe a few data analysis experiments based on a variety of exported data frame objects.

4.1 Statistical Analyses of Event Patterns

In these experiments, we perform analyses on event patterns extracted from 810,000 news stories in the Reuters Corpus (August 1996- August 1997) [14]

In the first experiment, we look for the activity patterns of presidents, ministers and chairmen as a function of day of the week. In a few seconds, InFact returns a 35,557 row travel/non-travel data frame from which S-PLUS® produces the (normalized) comparative dot chart of Figure 3.

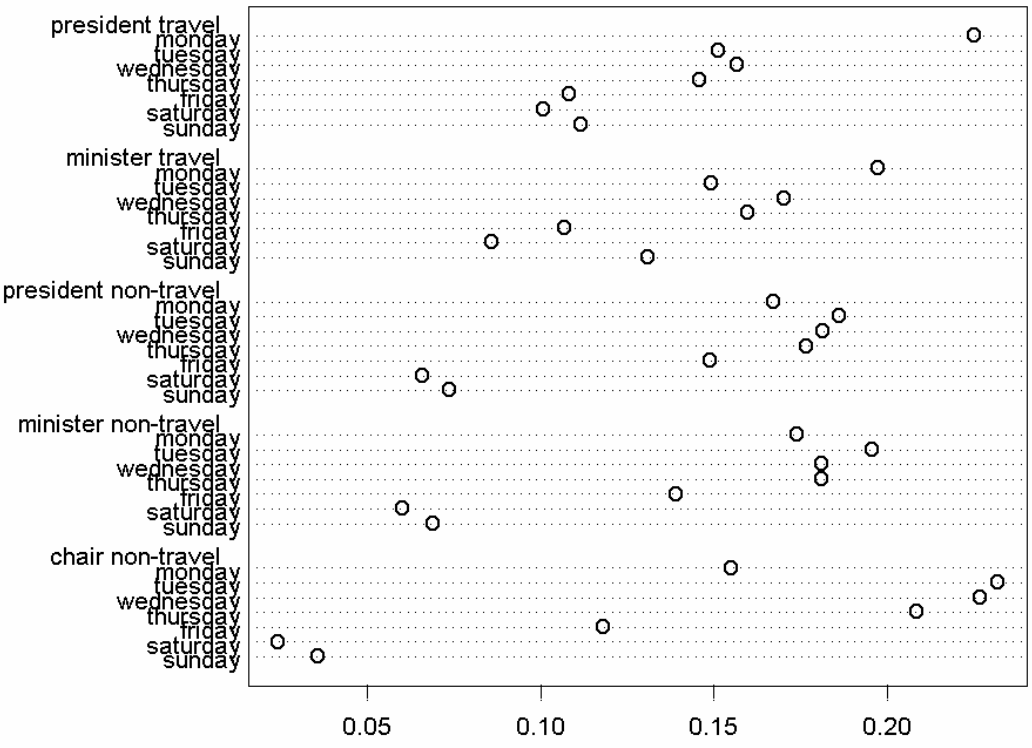

Figure 3: normalized comparative dot chart of travel and non-travel actions for presidents, ministers and chairmen generated directly from text

Next, we execute the query

[organization/Name]>acquire>[organization/Name]^[money]

and retrieve a set of relationships involving corporate acquisitions. We produce data frames with the following event variables:

- Buyer
- Target company
- Monetary amount involved in acquisition
- COUNTRY (Reuters document metadata)
- DATE (Reuters document metadata)

In a few seconds, InFact returns a 10,409 action (row)- data frame that we feed to S-PLUS®. In Figure 4a, we show time series plots of the number of company acquisitions per month that happened in the US vs. the rest of the world. In Figure 4b, we plot the total monetary amounts involved in the acquisitions as a function of time. Next, (Figure 5) we select the 6 countries that have the largest number of acquisitions and produce Trellis graphs showing the number of acquisitions as a function of time. Trellis Graphics® are designed to display data in a series of panels using a conditioning variable (COUNTRY, in this case). Each panel contains a subset of the original data corresponding to values of

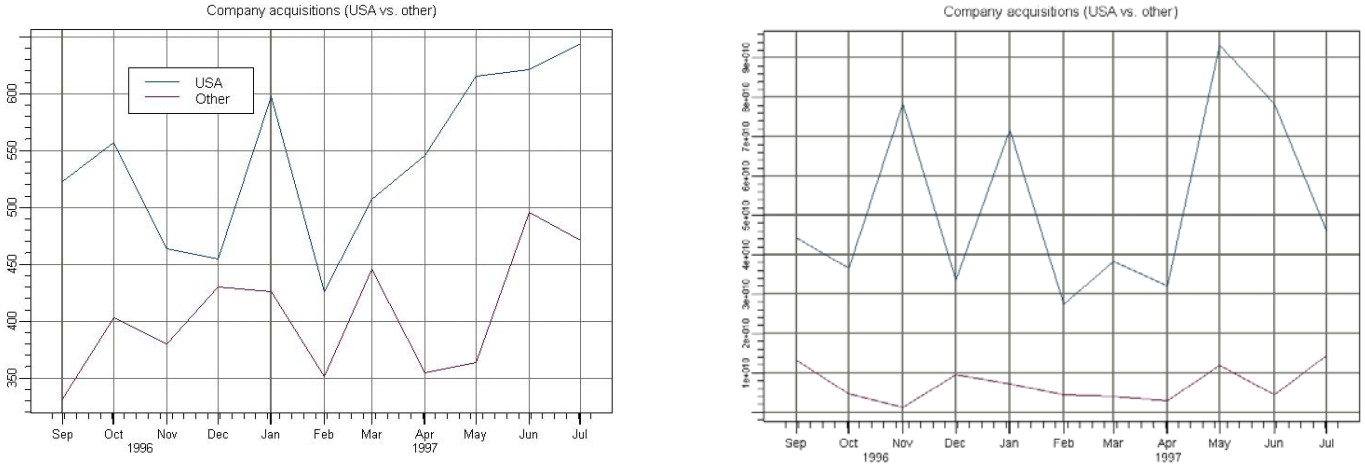

Figure 4 shows (a) time-series plots of company acquisitions in the US and other countries; (b) time-series plots money amount involved in the acquisitions

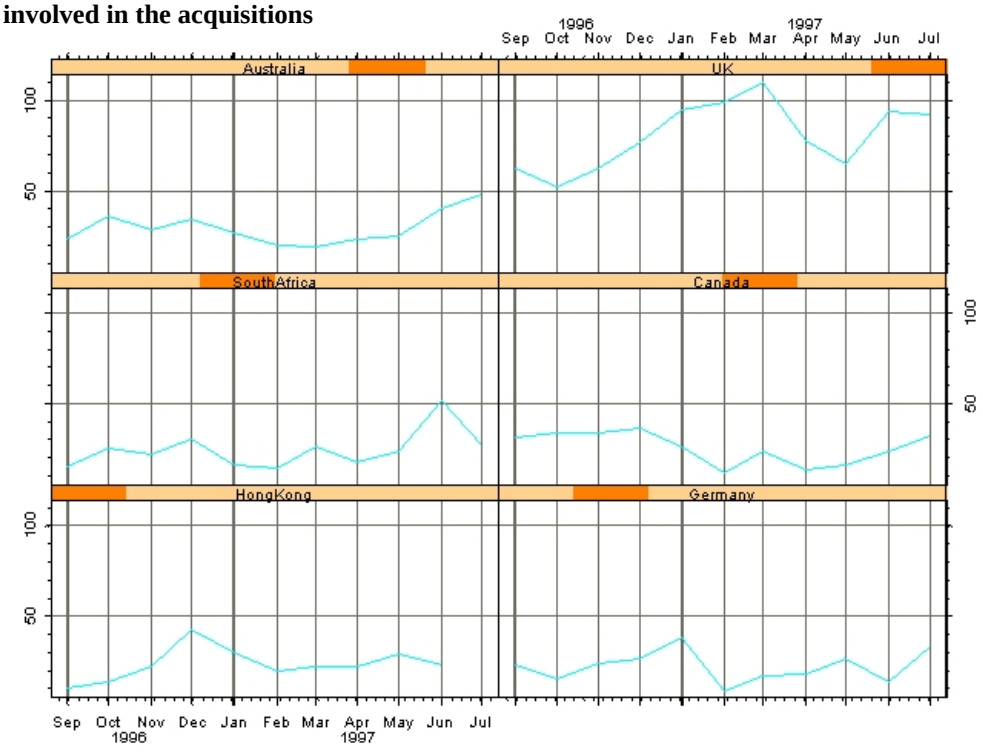

Figure 5 shows a trellis graph where each panel contains a time-series plot of company acquisitions in each of the top 6 countries other than the US.

4.2 Graph Visualization and Analysis

Relationship searches on large corpora can quickly yield thousands of results, if they are based on sole word co-occurence. It is difficult to display networks of such size and visually detect interesting subnetworks or patterns in them. However, with InFact, we can filter relationship based on contextual information, like types of entities, types of actions, action directionality, action modifiers, metadata or Boolean constraints, etc. Next we apply relational graph methods [21] in which we represent entities as the graph nodes and their relationships as the edges between such nodes. Each node is labeled with the corresponding entity name. Edges are labeled with the pairwise relationships between entities and the corresponding degrees for these relationships.

In this experiment, we visualize actions that imply hostile relationships between countries. First we extract a data frame of relationships from the Reuters Corpus with the query

[country]>'attack' OR 'invade' OR 'occupy'>[country]

and export data frames with the following relationship attributes:

- Source Country
- Target Country

Figure 6 shows the resulting graph. The vertices (shown in different colors) are grouped into subgraphs of interconnected country nodes. Colors are automatically assigned on the basis of the strength of semantic relationships. The edges within clusters are painted in black, and edges outside clusters are painted in gray. The clusters are computed using the JUNG [8] implementation of the Edge-Betweenness clustering algorithm [9]. This method uses properties calculated from the whole graph, which makes it robust against false positives and negatives.

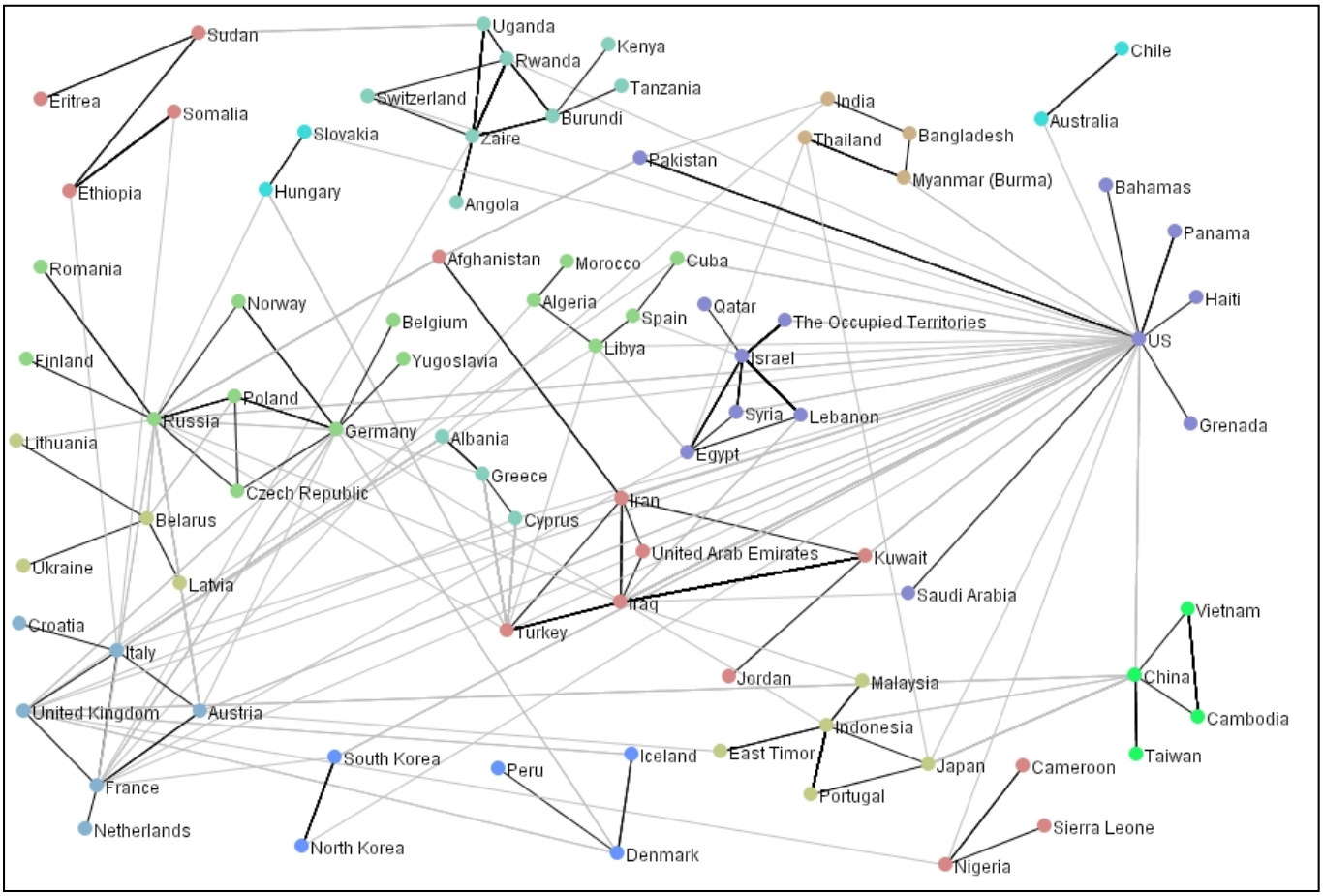

Figure 6 shows country relations presented as graphs, in which the nodes are countries and the dark edges represent hostile interactions between the countries from August 1996 to August1997.

4.3 Latent Semantic Analysis

In this section we employ the statistical data frame functionality to classify entities according to their action patterns. In this scenario, sets of entities, each represented by an M-dimensional vector, can be viewed as a matrix. One device that is particularly effective for event clustering and classification is LSR (Latent Semantic Regression), an algorithm that belongs to the class of Latent Semantic Analyses (LSA). Insightful holds three patents on LSR [18], [19], [20]. LSR constitutes an alternative to the classical approach to Latent Semantic Indexing (LSI) [16], which is based on Singular Value Decomposition (SVD) of a term-document matrix. LSR overcomes the scalability and speed limitations of LSI by casting the measurement of the distance between queries and document clusters as an inverse optimization problem [18]. Our proprietary algorithm can establish semantic links and learn patterns and associations from hundreds of thousands of index terms in a few seconds [17]. LSR alleviates the variability and reduces noise in term usage, while providing an explicit mechanism to explore latent terms and information. Another interesting capability of LSR is that the nature of the feature matrix is irrelevant, thus allowing handling of more complex linguistic descriptors than just keyword counts or inverse document frequencies. The ability to establish latent semantic relationships at the syntactic or semantic level across hundreds of thousands of documents in a few seconds will enable more robust cross document and event tracking than is currently possible.

Table 1 shows how, given a person, LSR can find other persons with similar action patterns. In this experiment, we first produce a person-action frequency matrix as described in Section 5. Then, we apply a standard TF.IDF weighting function to the matrix elements. We show that LSR can cluster together individuals that fit similar action profiles and uncover unexpected patterns of activity. Using any person name as query (left column), we return a ranked list of people that have similar activity pattern in the corpus. For instance, the query *"Andre Agassi"* returns a list of tennis players, while the query *"Steve Young"* returns a list of football quarterbacks. The corpus used for this last experiment was TIPSTER [15].

5. CONCLUSION

We have developed a system for flexible extraction of statistical data frames from large text collections. We have demonstrated the feasibility and benefit of this text-to-data framework, by performing a number of statistical analysis and visualization experiments on sample data frames and matrices.

Our approach rests on a sophisticated multivariate parameterization of text content, which enables interactive search and tracking of arbitrary events in large corpora. This representation makes use of natural language tags to (1) achieve a deeper level of knowledge discovery and (2) promote integrated storage and analysis of unstructured and structured sources of information.

6. ACKNOWLEDGMENTS

This work was partially supported by Dr. Joseph Psotka of the US Army Institute under contract No. W74V8H-05-C-0016. We also acknowledge many helpful discussions with Carsten Tusk, Navdeep Dhillon, and Matt Brown at Insightful.

7. REFERENCES

[1] Kao, A. and Poteet S. Report on KDD Conference 2004 Panel Discussion: Can Natural Language Processing Help Text Mining? ACM SIGKDD Explorations. Volume 6, Issue 2, December 2004.

[2] Manning, C.D., and Schutze H. Foundation of Statistical Natural Language Processing. The MIT Press, 2000.

[3] Hearst, M. Untangle Text Data Mining. Proceedings of ACL '99: the 37th Annual Meeting of the Association for Computational Linguistics, June 1999.

[4] Hirschman, L., Park, J.C., Tsujii, J., Wong, L. and Wu, C.H. Accomplishments and challenges in literature data mining for biology. Bioinformatics 18 (12) 1553-1561 (2002)

[5] Cohen, K.B., and Hunter, H. Natural language processing and systems biology. In Dubitzky and Pereira, Artificial intelligence methods and tools for systems biology. Springer Verlag, 2004.

[6] Proceedings of the Seventh Message Understanding Conferences (MUC-7). Morgan Kaufmann Publishers, 1997.

[7] NIST. Automatic Content Extraction (ACE) program. http://www.nist.gov/speech/tests/ace/

[8] JUNG, Java Universal Network/Graph Framework. http://jung.sourceforge.net

[9] Girvan M, Newman MEJ. Community Structure in Social and Biological Networks. Proc. Natl. Acad. Sci. USA 2002, 99:7821–7826.

[10] Tan, A. H. 1999, Text mining: The state of the art and the challenges, in Proceedings of the Pacific Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases, pp. 65-70.

[11] Montes-Y-Gomez, M. Gelbukh, A. Lopez-Lopez, A. Baeza-Yates, R., Text mining with conceptual graphs, In IEEE International Conference on Systems, Man, and Cybernetics, 2001

[12] Uramoto, N., Matcuzawa, H., Nagano, T., Murakami, A. Takeuchi, H., and Takeda, K. A text-mining system for knowledge discovery from biomedical documents. IBM Systems Journal, Vol. 43, No. 3, 2004, pp. 516-533

[13] Nasukawa T. and Nagano, T. Text analysis and knowledge mining system. IBM Systems Journal, Vol. 40, No. 4, 2001, pp. 967-984.

[14] Reuters Corpus, Volume 1, English language news stories, 1996-08-20 to 1997-08-19. Available at NIST: http://trec.nist.gov/data/reuters/reuters.html

[15] Tipster Corpus. Available at LDC: http://www.ldc.upenn.edu

[16] Deerwester, S., Dumais, S., Furnas, G.W., Landauer, T.K., and Harshman, R. Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41:391-407, 1990.

[17] Marchisio, G. and Liang, J. Experiments in trilingual cross-language information retrieval. Proc. Symp. Document Image Understanding Technology, 2001.

[18] Marchisio, G., Inverse inference engine for high performance Web search. United States patent No. 6,510,406, January 2003

[19] Marchisio, G. Extended functionality for an inverse inference engine based Web search. United States patent No. 6,757,646, June 29, 2004

[20] Marchisio, G. Internet navigation using soft hyperlinks. United States patent No. 6,862,710, March 2005

[21] Messmer, B. T. and Bunke, H. Efficient subgraph isomorphism detection: A decomposition approach. IEEE Transactions on Knowledge and Data Engineering, 12(2):307-323, 2000

About the authors:

Jisheng Liang is a research scientist at Insightful Corporation. His research interests include natural language processing, text mining, and document analysis. He earned a Ph.D. in 1999 from the Department of Electrical Engineering, University of Washington.

Krzysztof Koperski is a research scientist at Insightful Corp. He is doing research on interactive classification models, Bayesian information retrieval, data fusion, data modeling for very large databases, and query optimization. He received his Ph.D. degree in Computer Science from Simon Fraser University.

Thien Nguyen is a senior software developer with 20 years of experience. He has a Master's degree in Software Engineering from Seattle University.

Giovanni Marchisio is the Director of the Text Analysis and Search business unit at Insightful Corp. He has 15 years of experience in commercial software development related to search, computational linguistics, image mining, and multimedia information retrieval. He holds a Ph.D. degree in Geophysics and Planetary Physics from University of California, San Diego.

Table 1 shows clustering of person names based on their common action patterns in a corpus.

Person Name	Person names with similar actions	Common actions
nelson mandela	yasser arafat, lech walesa, john paul ii, hussein, vaclav havel, mandela, eduard shevardnadze, gerry adams, mangosuthu buthelezi, f w de klerk, winnie mandela	serve, address, tell, renounce, make, lead, meet, call for, speak, arrive, receive, visit, say, express, meet with, hold, call, take, end, thank, remain, praise, give, attend, urge
andre agassi	steffi graf, john mcenroe, monica seles, martina navratilova, boris becker, ivan lendl, michael chang, stefan edberg, uc irvine, pete sampras, gabriela sabatini, jimmy connors, royal, chris evert, chang, jennifer capriati, becker, jim courier, andres gomez	play, beat, win, defeat, lose, overcome, reach, wear, route, take, rally, withdraw, skip, serve, get, come, make, hit, miss, throw, force, pull, begin, need, rocket
rupert murdoch	donald trump, campeau, carl c. icahn, robert campeau, ted turner, carlo de benedetti, murdoch, thomson, harold simmons, irwin jacobs, kirk kerkorian, paul bilzerian, time warner, de benedetti, beatrice	own, sell, buy, acquire, control, make, launch, take, negotiate, include, pay, raise, put, purchase, reach, say, offer, use, need, seek, retain, take over, announce, face, hold
darryl strawberry	jose canseco, kevin mitchell, will clark, jack clark, eddie murray, wally joyner, cecil fielder, mark mcgwire, matt williams, joe carter, robby thompson, chili davis, lance parrish, howard johnson, dante bichette, don mattingly, kal daniels, dave henderson, roberto alomar	hit, drive in, play, tie, homer, stroke, get, snap, drop, leave, double, bang, line, walk out of, bat, drill, follow, lead, take, miss, go, finish, quit, expire, give
clint eastwood	tom cruise, oliver stone, arnold schwarzenegger, martin sheen, andy warhol	film, acclaim, capture, direct, include, receive, wish, remove, ask for, make, press, cop, move in, weave, extradite, shoot, spring, get across, donate, accept, plan, love, act, star, end
steve young	montana, jim everett, jim mcmahon, steve bono, troy aikman, john elway, phil simms, mike pawlawski, mark vlasic	throw, play, complete, get, start, lob, finish, share, run, sustain, wear, seem, come in, take, recover, rally, scramble, spot, find, make, convert, lead, game, replace, pick up
dan rather	tom brokaw, peter jennings, bryant gumbel, ted koppel, jane pauley, bernard shaw, beck, brent musburger, connie chung, winston churchill	anchor, salute, costume, visit, chastise, report on, report, interview, question, spur, warrant, apologize, say, prompt, speak, go, decline, ask, call, televise, become, make, show, sit in, read
barbara bush	dan quayle, nancy reagan, charles, raisa gorbachev, diana, barbara	wear, visit, undergo, take, tell, make, read, attend, tour, deliver, join, spend, praise, speak, say, choose, begin, appear, get, suffer, sit, receive, pose, report, keep