FemaRepViz: Automatic Extraction and Geo-Temporal Visualization of FEMA National Situation Updates

Chi-Chun Pan*		Prasenjit Mitra†

The Pennsylvania State University

Abstract

An architecture for visualizing information extracted from text documents is proposed. In conformance with this architecture, a toolkit, FemaRepViz, has been implemented to extract and visualize temporal, geospatial, and summarized information from FEMA National Update Reports. Preliminary tests have shown satisfactory accuracy for FEMARepViz. A central component of the architecture is an entity extractor that extracts named entities like person names, location names, temporal references, etc. FEMARepViz is based on FactXtractor, an entity-extractor that works on text documents. The information extracted using FactXtractor is processed using GeoTagger, a geographical name disambiguation tool based on a novel clustering-based disambiguation algorithm. To extract relationships among entities, we propose a machine-learning based algorithm that uses a novel *stripped dependency tree* kernel. We illustrate and evaluate the usefulness of our system on the FEMA National Situation Updates. Daily reports are fetched by FEMARepViz from the FEMA website, segmented into coherent sections and each section is classified into one of several known incident types. We use ConceptVista, Google Maps and Google Earth to visualize the events extracted from the text reports and allow the user to interactively filter the topics, locations, and time-periods of interest to create a visual analytics toolkit that is useful for rapid analysis of events reported in a large set of text documents.

Keywords: visual analytics, geo-temporal visualization, text processing, knowledge discovery, geospatial analytics

Index Terms: H.4.2 [INFORMATION SYSTEMS APPLICATIONS]: Types of Systems—Decision support;

1 Introduction

Successful crisis management requires rapid response. Unfortunately, first responders are overloaded with vast amounts of information that needs to be processed in a very short amount of time. Extracting required information totally automatically with high accuracy is not feasible (at least with today's technology). Therefore, semi-automatic methods are essential. Human beings can understand and analyze data more effectively if the data is presented in a visual mode. Ideally, the analyst or end-user should be presented with the automatically extracted information and the tool should interact with the end-user to enable the end-user to explore summaries of the data, correct extraction errors, select and examine articles of interest in more detail, etc. Such visual analytics tools can help first responders sift through large amounts of textual information fast in order to select texts reporting events of choice such that they can focus (drill down) into relevant articles and make decisions.

Spatial and temporal information are important attributes of data that must be recognized and treated as first-class objects. Such information forms a vital component of decision-making. Essentially, end-users want to know what happened, when, and where. First-responders find such information crucial for decision making during emergency situations. For example, in 2004, an earthquake of magnitude of 9.3 Richter occurred in the Indian Ocean. The earthquake resulted in a series of dreadful tsunamis and took more than 300,000 lives from south Asia to east Africa. Most residents of the coastal areas did not have advance warning of the tsunami. A decision support system that can quickly process scientific reports and observations and generate an emergency alarm would be of immense value in such situations. Local television and radio stations could then broadcast warnings in the impacted countries. Decision makers in one country could promptly provide critical information to the emergency response agencies in other countries. Thousands of people could evacuate earlier and their lives could be saved.

Although, in this paper we do not provide an automated decision support system, we outline the architecture of a visual analytics tool that the end-user can use to make better decisions. The tool, FEMARepViz, extracts and visualizes information from FEMA National Situation Updates (and could also be extended to handle other types of textual reports in the future) on a map and can show the progression of events over time. The FEMA reports were primarily weather reports but also had items related to bombings, chemical spills, etc. Our tool is especially useful when there are vast amounts of textual data that is impossible to process totally manually.

FEMARepViz is a hybrid information extraction system that extracts concept maps and spatial-temporal information from text documents using both rule-based and machine-learning methods. Daily reports are fetched from the FEMA website and automatically segmented into several incidents. Each incident then is classified into topics based on word frequency and tagged with timestamp and location names. The extracted information is stored in a repository and can be presented with visualization applications such as Google Earth. The system provides browsing and visualization of the incidents. Atop our extraction system, FactXtractor, like FEMARepViz, many other applications can be built such as emergency situation pattern analysis and real-time emergency monitoring.

1.1 FEMA National Situation Reports

The FEMA National Situation Updates contain information from a variety of sources including federal agencies, state and local government, and the news media. The reports are designed to provide in formation useful for emergency management planning and operational activities. Situation reports generally cover weather reports, earthquake activities, wildfire, and other incidents around the United States. The reports include location names indicating where the incidents happened. Sometimes persons or organizations involved in the incidents are also included in the reports. The richness of geographical information makes the FEMA National Situation Updates an excellent dataset for geo-temporal information analysis.

1.2 Our Contributions

The key contributions of this paper are:

*e-mail: julianpan@psu.edu
†e-mail:pmitra@ist.psu.edu

IEEE Symposium on Visual Analytics Science and Technology 2007
October 30 - November 1, Sacramento, CA, USA
978-1-4244-1659-2/07/$25.00 ©2007 IEEE

1. We design an information extraction system to automatically process text documents and create concept maps and geotemporal visualization. We demonstrate the usefulness of our system with the FEMA National Situation Updates.

2. We present a novel stripped dependency tree kernel for entity relation extraction.

3. We present a heuristic algorithm for location name disambiguation. We use simple rules and propose a clustering algorithm to determine the coordinates of locations.

The rest of the paper is organized as follows. The overview of system architecture is described in section 3 and technical details are presented in section 4. In section 2 we discuss related work. Finally we discuss future work in section 5 and conclude in section 6.

2 Related Work

RSOE HAVARIA AlertMap[4] is a world-wide disaster information system. The system reports real-time event updates for a variety of incidents such as earthquake, active volcano, and tropical storms. However, unlike our system, RSOE HAVARIA AlertMap collects structured data from different data sources. Every event has been classified by its data source (such as European flu data from EISS and Volcano information from SWVRC) and come with detailed information such as timestamp and location coordinates. Although our system is focused on processing unstructured data, in the future, it can be enhanced to utilize information from structured data sources for, say identifying event co-reference.

HEALTHmap[2] is a global disease information system that collects disparate data sources and provides a visualization of the current state of infectious diseases and their effect on human and animal health. The system gathers unstructured text from Google News, ProMED-mail[1], and alerts from the World Health Organization[2] and EuroSurveillance[3]. HEALTHmap processes text and extracts disease names and the location names appearing in the text. Locations are limited to countries and some major cities in certain countries. HEALTHmap is a customized system for a specialized domain of events, while our system is designed to process a broad range of events.

Zelenko, *et al.*, [18] proposed using tree kernels over shallow parsing trees to extract person-affiliation and organization-location relations. They created data samples by performing shallow parsing over sentences. Compared with deep parsing techniques, the results from shallow parsing are more reliable [19]. The kernels then compare similarity between parse trees by recursively matching nodes between two parse trees starting from the root nodes. In their experiments, kernel-based approaches have better performance than feature-based approaches with several different learning algorithms.

Culotta and Sorensen [11] defined a slightly more general version of tree kernels than that proposed by Zelenko, et al., [18] with a richer sentence representation. Their kernels are based on dependency trees instead of syntactic parse trees. Dependency trees include more information by considering syntactic relations between words. The experimental results show the dependency tree kernels have good precision but low recall on the ACE 2004 corpus. They combined a bag-of-words kernel into the dependency tree kernel to boost the performance. Harabagiu, *et al.*, [13] combined dependency trees with shallow semantic parsing and reported average F1-score of 78.41% on ACE 2004 corpus[17]. Greenwood and Stevenson [12] further extended the dependency tree kernels by using linked chain patterns and structural similarity measurement. Their approach can improve the performance over iterations; nonetheless the maximum F-score is only 0.329 due to low recall.

Roth and Yih [15] described a linear programming (LP) framework to detect named entities and entity relations. Unlike most NLP systems with pipelined architectures, their approach considers outcomes of different but mutually dependent classifiers simultaneously. The learning algorithm is a variation of the Winnow algorithm. They annotated the TREC data set with named entities and relations and then used the LP framework on it. Their results indicate that by optimizing the global interests instead of concentrating on task-specific constraints and accumulating errors within pipeline processes, the performance improved significantly.

Bunescu and Mooney [8] proposed a shortest path dependency kernel which outperformed tree kernels on the ACE 2004 corpus. Their approach treats the directed dependency graph as an undirected graph and finds the shortest path between two entities. Then they compare the shortest path example with a Cartesian product kernel to compute the number of common features on the path. The key idea of the shortest dependency path is to consider only the information relevant to entity relations. The advantage of dependency path kernels is that they do not consider the depth of entity nodes, hence they yield better recall. However, their approach only considers the predicate-argument structures between words. Entities in sentences such as "While in Paris, John never visited the Louvre." may never be linked.

Our novel approach combines the advantages of dependency tree kernels and that of an information elimination technique similar to that proposed by Bunescu and Mooney [8]. We propose to use *stripped dependency trees.* We show by empirical evaluation on the same data set used in [15] that our approach has better precision and improved recall than the dependency tree kernel proposed by Culotta [11].

3 System Architecture

As shown in Figure 1, our system has the following major components:(a)Text Extraction and Processing Module, (b) Disambiguation Module, and (c) Visualization and User Interaction Module.

The text extraction and processing module processes a set of documents and extracts named entities (like person, place, and organization names). It also segments a text into different contiguous segments having the same topic. The extraction module also detect events. For the current application, segmenting the document itself based on topics resulted in segregating the different events and we did not have to perform complex temporally-based event detection.

The disambiguation module is responsible for disambiguating named entities. For person names, the role of the disambiguation module is to establish which names refer to the same person, e.g., "Mohandas Gandhi" and "M.K. Gandhi", or if the same name actually refers to different people, e.g., two occurrences of "James Smith" may refer to different people in two different contexts. This module is also responsible for coreference resolution. The disambiguation module processes geographical named entities and using the information available about the contexts in which they occur, disambiguates the geographical named entity to an exact location. For example, it would generate an exact latitude and longitude for a reference to the town "Springfield" using the context information to derive which of the several "Springfield"'s in the U.S.A. (or beyond) it is.

The visualization module presents the information extracted from the text on a map for a particular time-point (or time-period). The user can interact with this module to filter the data being displayed based on topics, time periods, geographical locations, or people of interest. The user can provide feedback to correct incorrect items shown on the map, e.g., to move a displayed event from one location to another. The feedback provided by the user

[1] http://www.promedmail.org

[2] http://www.who.int/csr/alertresponse/en/

[3] http://www.eurosurveillance.org/

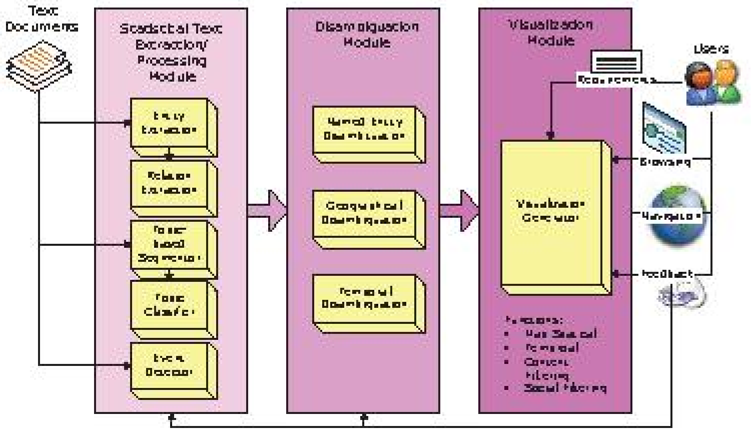


Figure 1: Overview of system architecture

and the user's activities can be logged for future improvement of the extraction from text.

To provide the functionality of the three modules indicated above, our system utilizes three main components: FactXtractor, GeoTagger, and FEMARepViz. Every component is designed as a standalone web service to ensure system flexibility and interoperability.

3.1 Information Extraction

FactXtractor is an information extraction web service for Named Entity Recognition (NER) and Entity Relation Extraction (RE). FactXtractor processes text documents using an open source text processing platform (GATE[1]) and identifies entity relations using Stripped Dependency Tree kernels. Figure 2 shows the major steps of the processing flow. The input of FactXtractor is a document or a set of documents in plain text format. First, the text is processed by a shallow parser to get parts of speech (PoS) tags. In addition, it identifies other linguistics features such as noun chunks and verb groups. In the second step, we use the Named Entity Tagger provided by GATE to extract named entities. Next, a deep parser processes the text to construct dependency trees. Syntactic relationships (subject-verb-object) can be easily extracted using dependency trees. FactXtractor extracts relationships using the Stripped Dependency Tree Kernels. The output of FactXtractor is a graph of the extracted entities and their relationships formatted in OWL [7] (see Figure 7). This extracted graph can be visualized with ConceptVista[4]. We will discuss the methods used in FactXtractor in section 4.

3.2 Geographical Name Disambiguation

GeoTagger is a geocoding Web service. GeoTagger maps a location name that appears in text to its correct co-ordinates on a map. GeoTagger uses the U.S. Geological Survey (USGS) Geographic Names Information System (GNIS)[5] for U.S. locations, National Geospatial-Intelligence Agency (NGA) GEOnet Names Server (GNS)[6] for locations outside U.S., and Google Map for global locations . We use Google Map as a secondary reference because GNIS and GEOnet contain many locations which are only used in local. For example, State College, MS is listed in GNIS but is not listed in Google Map. If a location is listed in GNIS or GEOnet but not in Google Map, we do not consider it as a candidate for assigning location names. We will discuss the details in section 4.

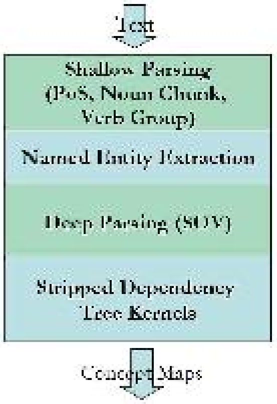


Figure 2: Text Processing Flow

FEMARepViz is a visualization generation Web service for the FEMA Situation Reports. FEMARepViz processes situation reports using FactXtractor and GeoTagger. Processed reports are stored in a repository and can be retrieved by a Web interface. The output is a KML document that provides dynamic updates and interactive visualization. Figure 3 illustrates the information processing flow and connections between the web services.

4 Methods

4.1 Named Entity Recognition

We use GATE[1] to extract named entities. GATE is distributed with an Information Extraction component set called ANNIE. ANNIE contains a rule-based engine. ANNIE extracts named entities using gazetteers and language features like Part-of-Speech, etc. The named entities we extracted include person, location, organization, and time. Table 1 shows the performance evaluation of the GATE named entity extraction module on the CoNLL 2004 corpus.

4.2 Entity Relation Extraction

We use an approach applying kernel methods on dependency trees for relation extraction. Kernel methods are widely used in a variety of machine learning problems with many popular algorithms such as Support Vector Machines[10] (SVM) and Perceptron[6].

[4] available at: http://www.geovista.psu.edu/ConceptVISTA

[5] http://geonames.usgs.gov/pls/gnispublic/

[6] http://earth-info.nga.mil/gns/html/index.html

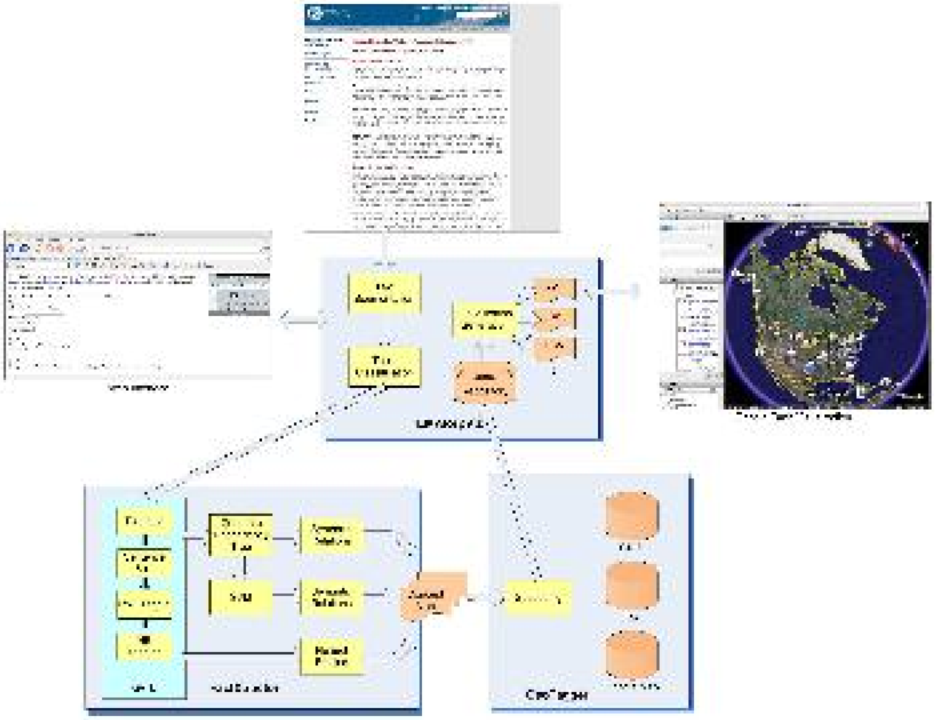


Figure 3: Information processing workflow

Table 1: Evaluation of the GATE named entity extraction module

	PER	LOC	ORG
Precision	0.89	0.81	0.65
Recall	0.74	0.79	0.69
F_1	0.8	0.8	0.66

A dependency tree represents of a parsed sentence and shows the relations between words [11]. A node in a dependency tree is the corresponding word or words in the original sentence. The dependence between words could be *verb–subject*, *verb–object*, *verb–adjective*, etc. First we process each sentence by NLP tools to obtain the dependency information, named entities and word features such as PoS tags. We then generate a dependency tree for every pair of named entities in a sentence. A similarity score between two trees can be computed recursively from the roots of the trees (see [11] for details).

One important observation of dependency trees is that if a node does not contain a descendant with a relation argument in its subtree, the node usually will not be involved in the relationship between entities. For example, in Figure 5, removing the node "a" from both trees will not change the fact that the two trees are similar to each other.

Based on this observation, we define the concept of *stripped dependency tree* (SDT) as follows: an SDT is a subtree of a dependency tree. Every node in an SDT has at least one descendant node that is a relation argument of a relation. We only consider binary relations in this paper. Hence, each relation has two relation arguments. For example, in a relation (*Pittsburgh*, *located_in*, *Pennsylvania*), *Pittsburgh* and *Pennsylvania* are the relation arguments. Nodes are removed if they do not have a descendant with relation argument. Stripping a dependency tree can be done in $O(n)$ with a depth first search (DFS), where n is the number of nodes in the dependency tree. Because we have to perform a DFS to find the minimum subtree that contains the specified entities, adding the node elimination will not increase the computational complexity. Note that there will be no non-matching nodes in the SDT. Hence, there is no difference between the contiguous tree kernel and the sparse tree kernel for SDT.

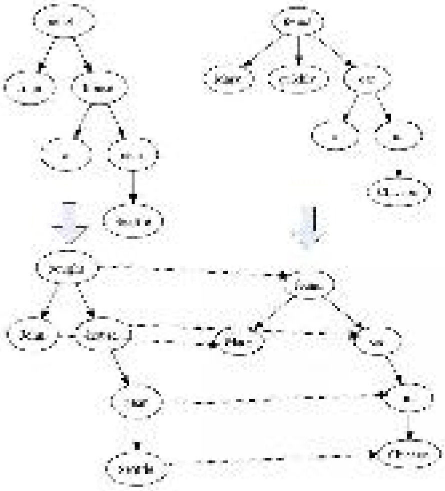


Figure 5: Dependency trees for "John bought a house near Seattle" and "Mary quickly found a car in Chicago".

The advantages of SDTs are as follows:

1. The SDT is computationally efficient. We can use the continuous tree kernel algorithm to compute the result. An SDT usually has fewer nodes than the original dependency tree.

2. The matching process for SDTs will not be interrupted by ir-

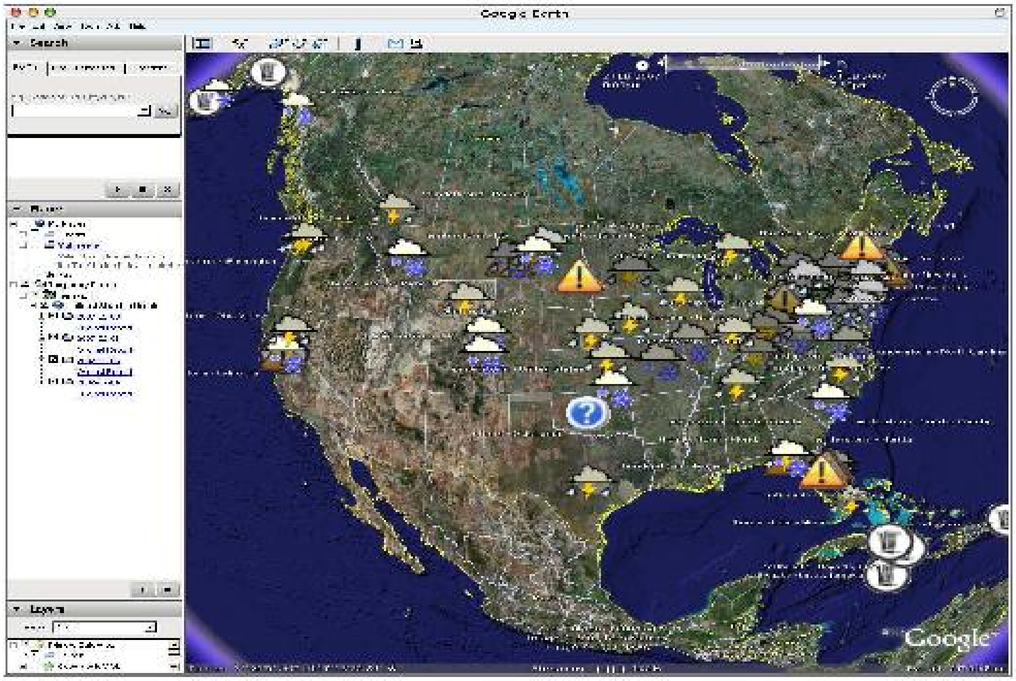


Figure 4: Processed FEMA National Situation Updates from Feb 3, 2007 to Feb 6, 2007.

relevant nodes. SDTs should give better results on similarity measurement for entity relationships.

Some important information may be discarded when a dependency tree is converted to an SDT. For example, a sentence "John likes Mary." should not be matched with "John doesn't like Mary.". To handle this, and to improve the performance of the algorithm, we have made some enhancements; we describe them below.

We use MINIPAR to parse each sentence; we then generate a dependency tree for the sentence using output of MINIPAR. MINIPAR can achieve about 88% in precision and 80% in recall for determining dependency relationships[7]. By linking each word with its head word of the MINIPAR output, dependency trees can be created for all the sentences.

Table 2: Annotated relations in the training corpus.

Relations	Example
loc, located_in, loc	(Seattle, located_in, Washington)
per, work_for, org	(Steve Jobs, work_for, Apple)
org, orgBased_in, loc	(Microsoft, orgBased_in, Redmond)
per, live_in, loc	(Bush, live_in, D.C.)
per, kill, per	(Oswald, kill, JFK)

For each node, we assign a set of features generated by GATE. We used the features used by Culotta and Sorensen [11]. To improve the performance, we added some additional features. They are orthography, word root, verb voice, verb negation, and synonyms obtained from WordNet. Table 3 lists all the features that we used in our experiments. Note that we represent WordNet hypernym and synonym set by their set-id. WordNet will give a list of hypernym sets ordered by estimated frequency and likewise for the synonym sets. We use both hypernym and synonym sets to capture the semantics of words.

Table 3: List of feature assigned to each tree node.

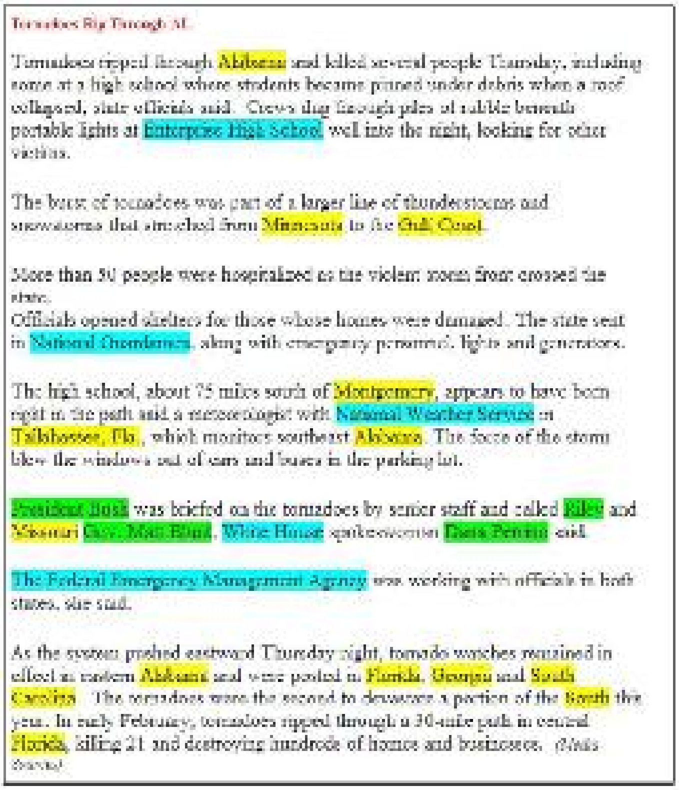
Feature	Example
word	John
POS	NN
General POS	N
Entity type	Person
Relation argument	arg_1
WordNet hypernym	4274300
WordNet synonym	4274300
Orthography	upperInitial
Chunk tag	NP
Word root	john
Verb voice	active
Verb negation	yes

We have implemented the tree kernel algorithms in Java and used it in an SVM. We augment the LibSVM[9] implementation to use the tree kernels. We use the CoNLL 04 corpus[8] to evaluate the performance of tree kernels. The corpus consists of articles from several different sources such as WSJ and AP. There are 5925 sentences in the corpus. Among those sentences, 5336 named entities and 2040 binary relations are manually annotated. The annotated named entities include 1685 persons, 1968 locations, 978 organizations and 705 others. The relations between those named entities include 406 located_in, 394 work_for, 451 orgBased_in, 521 live_in, and 268 kill. Table 2 shows examples for each relation. Since our extractor already captured the *Subject-Verb-Object* structure using dependency trees, we only trained the SVM for the first four relations.

We compared the tree-kernels results with the best results published in [15] obtained using a linear programming algorithm. The F_1 is computed by considering both precision and recall. Precision is the ratio of the number of correctly predicted positive answers to the number of total predicted positive answers. Recall is the ratio of the number of correctly predicted positive answers to the number

[7] http://www.cs.ualberta.ca/~lindek/minipar.htm

[8] available at ttp://l2r.cs.uiuc.edu/~cogcomp/Data/ER/conll04.corp



Tornadoes Rip Through AL

Tornadoes ripped through Alabama and killed several people Thursday, including some at a high school where students became pinned under debris when a roof collapsed, state officials said. Crews dug through piles of rubble beneath portable lights at Enterprise High School well into the night, looking for other victims.

The burst of tornadoes was part of a larger line of thunderstorms and snowstorms that stretched from Minnesota to the Gulf Coast.

More than 50 people were hospitalized as the violent storm front crossed the state.
Officials opened shelters for those whose homes were damaged. The state sent in National Guardsmen, along with emergency personnel, lights and generators.

The high school, about 75 miles south of Montgomery, appears to have been right in the path said a meteorologist with National Weather Service in Tallahassee, Fla., which monitors southeast Alabama. The force of the storm blew the windows out of cars and buses in the parking lot.

President Bush was briefed on the tornadoes by senior staff and called Riley and Missouri Gov. Matt Blunt, White House spokeswoman Dana Perino said.

The Federal Emergency Management Agency was working with officials in both states, she said.

As the system pushed eastward Thursday night, tornado watches remained in effect in eastern Alabama and were posted in Florida, Georgia and South Carolina. The tornadoes were the second to devastate a portion of the South this year. In early February, tornadoes ripped through a 30-mile path in central Florida, killing 21 and destroying hundreds of homes and businesses. (Mark Stevens)

Figure 6: A fragment of the FEMA National Situation Updates with entities highlighted

of positively annotated relations. We use the following equation to compute the F_1 score.

$$F_1 = \frac{2 * Prec. * Rec.}{Prec. + Rec.}$$

Table 4 shows the performance comparison between F_1 scores for both LP and SVM with tree kernels using 5-fold cross-validation. The results indicate that tree kernel approaches outperform the linear programming approach for all relations tagged in the testing corpus. The tree kernel with SDT further improved the original dependency tree kernel described in [11]. As shown in 4, combining contiguous tree kernel with SDT did not improve the precision very much. The recall of the tree kernel with SDT is 3% to 8% better than the original dependency tree kernel. By reducing noise introduced by unnecessary nodes in dependency trees, the tree kernel performed better prediction for identifying relations between entities. We also experimented with different values for the decay factor λ. Our observations are in line with those observed by Culotta and Sorensen [11]. The performance did not vary much with different λ values.

4.3 Geo-Coding

Grounding a location name with a correct coordinate is challenging. Essentially, we want to disambiguate places with the same name such as "Springfield" to the unambiguous "Springfield, IL" and "Springfield, PA". If additional information is not available, the geo-coding for "Springfield" becomes arbitrary. In our system, we have implemented a dedicated component called GeoTagger that is used for geo-coding tasks.

GeoTagger takes a short text as input then determines a geographical scope for the text. The geographical scope of a text segment is determined based on the locations with higher certainty. We define geographical scope in four levels: *World*, *Continent*, *Country*, and *Province*. For each text segment, the largest geographical scope will be used to select candidate coordinates. For example, if "United States", "Pennsylvania" and "Georgia" are extracted from a text segment, then the geographical scope is determined at the "Country" level. Determining the geographical scope for a text can eliminate unlikely candidate locations. For example, if the highest geographical scope is "Country" and two countries, the "United States" and "Germany" have been extracted from the text, then only locations within these two countries will be considered in the disambiguation algorithm.

Even within a geographical scope, each location name could have several candidate locations. We perform geo-spatial disambiguation based on the following intuition: *Location names that occur close together in the same document segment refer to places that are geographically close.* For each location name, GeoTagger obtains the latitude-longitude of all possible places with the same location name, provided the place is within the geographical scope. GeoTagger then clusters locations such that each cluster contains only one occurrence of a location with a particular location name. That is, two Springfields cannot belong to one cluster, because the objective of clustering is to separate the possible locations and choose the location that is closest to the other (possibly non-ambiguous) location names.

The clustering algorithm works as follows. For each location name n in a set of location names N, we construct a list of latitude-longitudes of places with the name n. GeoTagger uses the k-means clustering algorithm with a parameter k to control the maximum number of clusters. The distance between two points is computed using the Euclidean distance between their coordinates. The result of the k-means algorithm is a one-to-one mapping between location names and their coordinates. The algorithm is listed in Algorithm 1. GeoTagger increments the value of k starting from $k = 2$ until clusters are obtained that cover all location names in the set N such that no cluster contains two locations with the same name.

Algorithm 1: Algorithm for location selection

```
Input  : Geographical scope $GS$, Set of location names $N$,
         Maximum number of clusters $K$, threshold $\omega$
Output: Map of coordinations $M$
begin
    $M \leftarrow \phi$ for $each n \in N$ do
        $Coors_n \leftarrow$ coordinates within $GS$ match $n$
    for $k = 1$ to $K$ do
        Arbitrarily select $coor_n \in Coors_n \rightarrow M(n)$
        $Coors_n = Coors_n - \{coor_n\}$
        $D_k$ = k-means($k$,$M$)
        for $each n \in N$ do
            for $each coor_i \in Coors_n$ do
                if $Replacing\ M(n)\ by\ coor_i\ reduce\ D_k$ then
                    $M(n) \leftarrow coor_i$
        if $\frac{D_k - D_{k-1}}{D_{k-1}} < \omega$ then
            break
    return $M$
end
```

Despite the simplicity of our geographical names disambiguation algorithm, the algorithm works reasonably well on the situation reports. However, geographical names that refer to large areas such as "Great Plains" or "The Mississippi River" may create difficulty since we only consider point-to-point distances. Extension of GeoTagger to handle non-point locations is left as future work.

Table 4: Comparison of relations classification results demonstrating improvement of our tree kernel + SDT method over the existing LP method[15] for the CoNLL 04 corpus.

Relations	LP			tree kernel			tree kernel + SDT		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1	Prec.	Rec.	F_1
located_in	54.5	64.0	58.9	78.6	58.4	67.0	79.7	61.5	69.4
work_for	69.2	50.5	58.4	77.7	61.2	68.5	76.3	65.1	70.2
orgBased_in	76.7	50.3	60.7	67.2	55.6	60.8	71.3	60.1	65.2
live_in	60.7	57.0	58.8	74.0	57.1	64.4	79.6	58.9	67.7

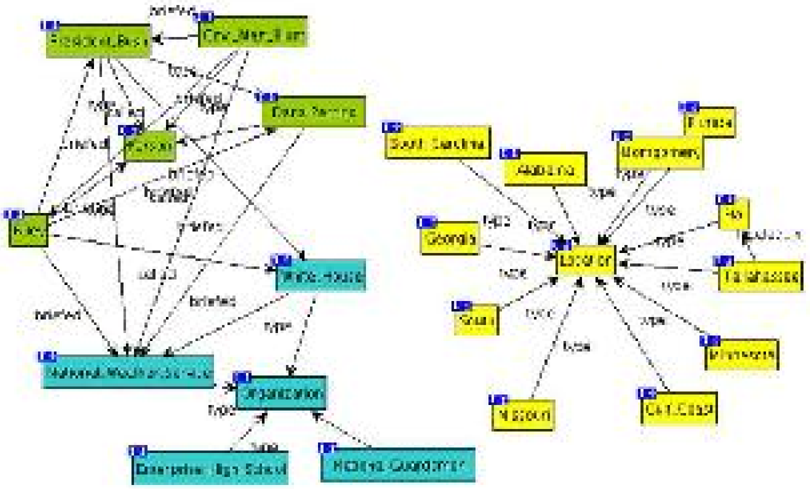


Figure 7: The Concept map generated from the segment of the FEMA National Situation Updates shown in Figure 6

4.4 Document Segmentation and Topic Classification

We use the n-gram language model described in [14] for topic classification. The classification model estimates the maximum likelihood for a sequence of words by computing conditional probability of previous $n-1$ words. A category then can be decided by picking $c^* \in C = \{c_1, \ldots, c_{|C|}\}$ that has the largest posterior probability given the text with the following equation.

$$c^* = \underset{c \in C}{\operatorname{argmax}} \{\Pr(c|D)\}$$

where D is the text segment. We use the DynamicLMClassifier implementation in LingPipe[3]. Text segments will be classified into one of nine categories that commonly appear in situation reports. The categories include *disease, noticeable, storm, wildfire, earthquake, rain, thunderstorm, winter storm,* and *others*. The *noticeable* category includes some uncommon incidents deserving of notice such as *tornadoes, power plane exploding, and chemical leakage*. 20 text segments are manually selected in each category as training corpus. Our preliminary evaluation indicates the topic classification model could achieve 93% accuracy.

For text documents where the topics are not known or manually labeled training examples are not available, a clustering based topic detection [5] and text segmentation algorithm can be employed [16].

4.5 Geo-temporal Visualization

We provide the following visualization capabilities with FEMARepViz.

First, the extracted graph between named entities that is generated as an OWL [7] file can be displayed with ConceptVista (Figure 7). In the named entity graph, entities are color coded for their entity type. Links between entities indicate relations among them. The ConceptVista visualization is useful for users to grasp the key concepts (people/organization/location) within the reports without reading them.

Second, FEMARepViz visualizes the situation reports using Google Earth. From each report, events are detected using the segmentation algorithm mentioned above and geo-coded. Each event is represented using an icon that shows the type of event that occurred (using one icon for each of the known categories listed above). Geo-coded incidents will be generated in Google Earth KML format and upload to Google Earth by a NetworkLink. If a particular event occurred in multiple places, the icon for that event is shown at all the locations on Google Earth. By using a right-click of the mouse, an user can see a summary of the event associated with an icon on a translucent pane that is hidden once the next icon is clicked.

Each situation report has a date associated with it. FEMARepViz extracts the first date entity in each report and takes it to be the date the report was published. A timestamp is added to each incident to indicate the temporal relationship between incidents. Our user interface has a time-slide wedge using which an user can change the timeline and examine the incidents that happened during the time period around the world (Figure 8).

Third, Google Earth is used to display the extracted information similarly to Google Map. However, in this version, the user can use a frame at the left-hand side to select topics, people, location and time that is of interest. This allows the user to filter out the entire set of events and display only a set of events that are of interest to the user. Moreover, Google Earth allows users to play the dated events as an animation. The feature is useful for visualizing a sequence of events such as spreading of a wildfire or movement of a hurricane over different locations.

5 Future Work

In the future, we plan to build statistical analysis modules on top of our information extraction system. Currently, two types of fore-

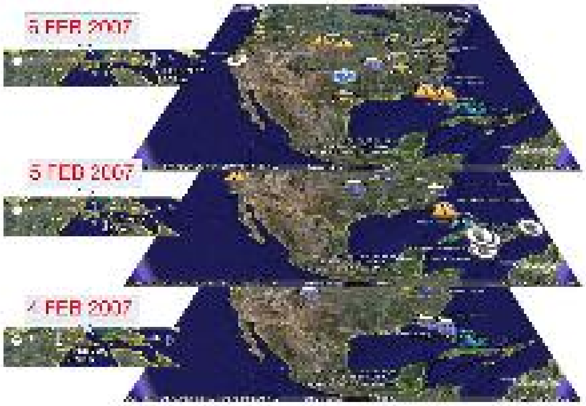


Figure 8: Geo-temporal visualization with Google Earth from February 4th 2007 to February 6th 2007.

casting modules are under development.

Periodic Incidents Forecasting: Periodic incidents can be predicted with time series analysis such as moving average or exponential smoothing. Incidents periodically appear in situation reports such as weather conditions are usually with seasonal factors. Hence, seasonal adjustment is necessary for forecasting. Using time series analysis we can predict the likelihood for periodic incidents in a region.

Correlated Incidents Forecasting: Some incidents may occur conditionally after other incidents. For example, a flood may happen in some region after heavy rain. Tornadoes may be caused by unusual heat or thunderstorms. Such correlation can be found in historical reports and used to forecast future incidents.

Although our relation extraction and geographical names disambiguation algorithms work reasonably well on the testing corpus and the situation reports, we will conduct more experiments to justify the significance of the performance improvement.

Moreover, we will add more data sources with varying reliability and types of events including ProMED Mail, Global Disaster Alert and Coordination System, news media, and Internet blogs.

6 Conclusion

We present an architecture and an implementation of a system that can be used to automatically extract information from vast amounts of textual data and present the extracted information visually to end-users. We use FEMA National Situation Updates as our test bed to demonstrate the usefulness of our system. Our system, FEMARepViz utilizes an entity-relationship extractor, FactXtractor, to extract named entities and links entities with syntactic and semantic relations. Named entity graphs can be created to visualize entity relations. Furthermore, we use GeoTagger to resolve geographical ambiguity and create geo-temporal visualizations with FEMARepViz and Google Earth.

Since there is no computational approach that can achieve human-level accuracy for complex information extraction tasks, visualization could be a solution to bridge the gap between fully automated extraction and search systems and manual data processing. We believe that our system can benefit users who have needs to analyze massive geo-temporal information in an efficient manner. We conjecture that our system architecture can form the basis for future visual analytics systems over text data, especially for information with important geo-spatial and temporal attributes.

Acknowledgements

This work was performed with partial support from the National Visualization and Analytics Center (NVAC), a U.S. Department of Homeland Security Program, under the auspices of the Northeast Regional Visualization and Analytics Center (NEVAC). NVAC is operated by the Pacific Northwest National Laboratory (PNNL), a U.S. Department of Energy Office of Science laboratory.

References

[1] GATE: General architecture for text engineering. http://gate.ac.uk.

[2] HEALTHmap. http://www.healthmap.org/.

[3] LingPipe. http://www.alias-i.com/lingpipe/.

[4] RSOE HAVARIA Alert Map. http://hisz.rsoe.hu/alertmap/woalert.php?lang=eng.

[5] *On-line new event detection and tracking*. ACM Press, 1998.

[6] A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[7] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider, and L. Stein. Owl web ontology language reference. Technical report, W3C.

[8] R. C. Bunescu and R. J. Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, October 2005.

[9] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[10] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[11] A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *ACL*, 2004.

[12] M. A. Greenwood and M. Stevenson. Improving semi-supervised acquisition of relation extraction patterns. In *Proceedings of the Workshop on Information Extraction Beyond The Document*, pages 29–35, Sydney, Australia, July 2006. Association for Computational Linguistics.

[13] S. M. Harabagiu, C. A. Bejan, and P. Morarescu. Shallow semantics for relation extraction. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI '05)*, pages 1061–1066, Edinburgh, Scotland, UK, 2005.

[14] F. Peng, D. Schuurmans, and S. Wang. Language and task independent text categorization with simple language models. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 110–117, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[15] D. Roth and W.-T. Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL-2004*, 2004.

[16] B. Sun, P. Mitra, L. C. Giles, J. Yen, and H. Zha. Topic segmentation with shared topic detection and alignment of multiple documents. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 199–206, New York, NY, USA, 2007. ACM Press.

[17] A. M. , et al. Ace 2004 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 2005.

[18] D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106, 2003.

[19] S. Zhao and R. Grishman. Extracting relations with integrated information using kernel methods. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 419–426, Morristown, NJ, USA, 2005. Association for Computational Linguistics.