

Spatio-Temporal Analysis on FEMA Situation Updates with Automated Information Extraction

Chi-Chun Pan
Penn State University
University Park, PA, 16802
cpan@ist.psu.edu

Prasenjit Mitra
Penn State University
University Park, PA, 16802
pmitra@ist.psu.edu

Auroop R. Ganguly
Oak Ridge National
Laboratory
Oak Ridge, TN 37831
gangulyar@ornl.gov

ABSTRACT

With the advent of the World-Wide-Web, there is an over-abundance of textual information. Information present in digital documents can be utilized better if it can be extracted automatically and scalably from text and visualized using visualization tools. In this paper, we present an automated information extraction and visualization tool for human sensor data. Our system consists of three main components: FactXtractor, GeoTagger, and FEMARepViz. Named entities and entity relations are extracted using FactXtractor. We have proposed a novel stripped dependency tree kernel for a Support Vector Machine (SVM) based classifier to identify semantic relationships among entities. GeoTagger disambiguates location entities. Built on top of the first two system components, FemaRepViz is an application that segments text documents, identifies the topic of the segments, extracts location entities, disambiguates them, and visualizes the extracted information on Google Earth or Google Map. Our empirical evaluation shows that the system achieves reasonable accuracy.

1. INTRODUCTION

Recently, there is a growing interest in analysis of spatio-temporal information. Essentially, people are eager to know what happened, when, and where. Such information could be crucial for decision making during emergency situations. For example, in 2004, an earthquake of magnitude 9.3 Richter in the Indian Ocean created a series of dreadful tsunamis and took more than 300,000 lives from South Asia to East Africa. Ideally, we would want a decision support system that can quickly process emergency alarms and local news from Indonesia and promptly provide critical information to the emergency response agencies in other countries. Government authorities could evacuate cities to prevent the tragedy and thousands of people could be saved.

The key of successful crisis management is rapid responsiveness which relies on efficient information processing. Unfortunately, emergency situations usually come with an overload of information created by people from different aspects, which is extremely difficult for human beings to process efficiently. Hence, a tool that automatically extracts useful pieces of information from vast sources of

textual data is vital for such scenarios. However, even the most advanced information extraction system cannot achieve human-level accuracy. Automatic information extraction often generates errors especially when ambiguity exists in the underlying data. Therefore, there is a need for new tools that can deliver automatically extracted information (what? when? where?).

Extracting named entities from text documents is an important task because named entities, often, represents the main subject or provides critical information. Extracting such information automatically and visualizing extracted named entities spatio-temporally is a challenging problem. Although research in natural language processing and information retrieval have addressed the information extraction problem, automatically processing context-sensitive text is extremely hard; the challenge lies in designing solutions that provide acceptable accuracy and scalability. With NLP techniques, computer systems could process text documents and extract useful information. Named entities recognition (NER) techniques using hand-crafted rules or machine-learning methods enable computers to identify and tag entities within text. However, connecting named entities with meaningful relationships still remains as an extremely difficult problem. Relationship extraction is a hard problem because of the nuances of natural language and the absence of NLP tools that are sound and complete in identifying and reasoning with such relationships from any text document. Therefore, existing research has attempted to use statistical methods to identify and extract relationships.

Automatic extraction of spatio-temporal information is a challenging problem. Consider a scenario where an end-user has to examine a large number of documents. The end-user is not interested in reading and understanding the entire text in the large set of documents. Instead the focus of the end-user is to examine the key information in the documents. When the amount of extracted information is large, the end-user also seeks tools that can filter and display only information relevant to the end-users task at hand. Tools that assist end-users to explore extracted named entities and their relationships obviate the need for the end-user to manually read all the documents and assimilate the information. While such a manual effort is likely yield more accurate results, the end-user does not have enough time to read the vast amounts of information available, say, on the World-Wide-Web. For example, the end-user, a sociologist, may be interested in identifying the travel patterns of eminent scientists of the 18th century from all the documents available. Such an user will find a tool that extracts all pairs of named entities, like person-names and geographical locations, such that the person visited a particular geographical location. If the named entities and their relationships can be extracted automatically, one can display

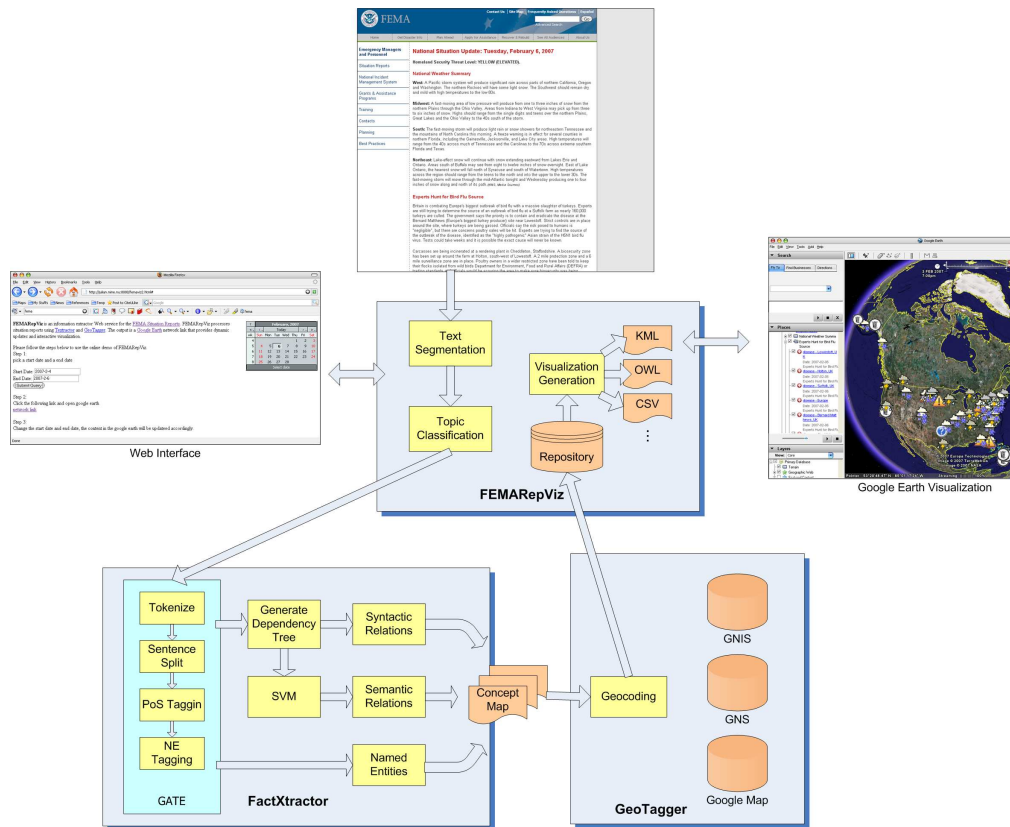


Figure 1: Overview of system architecture

the extracted triples as a graph and allow end-users to navigate the graph. Good visualizations make it easy for the end-users to examine the (extracted) information and make better decisions. In addition, grounding the geographical locations onto maps and sorting the segments of an article in chronological order could help the end-user. Using tools, like those outlined in this paper, the end-user can quickly explore the extracted information and only read promising documents in detail.

In this paper, we present an information system using text extraction and visualization techniques to support analysis of vast amount of human sensor data. We design a hybrid information extraction system, FactXtractor, to extract concept maps and spatial-temporal information from text documents using both rule-based and machine-learning methods. We outline FEMARepViz, a prototype application that processes text documents (FEMA National Situation Updates). FEMARepViz fetches daily reports from the FEMA website and segments them based on their topics, and extracts spatio-temporal entities from the reports. The extracted information is stored in a repository and can be presented with visualization applications such as Google Earth or Google Map. FEMARepViz utilizes our named entity and relationship extractor, FactXtractor, to identify spatio-temporal entities, and uses our spatio-spatial disambiguation tool, GeoTagger, to disambiguate the extracted spatio-spatial named entities. FEMARepViz provides an intuitive way to browse and visualize FEMA situation updates. There are many applications which can be built on top of our system such as emergency situation pattern analysis and real-time emergency update monitoring.

The novelty of FactXtractor lies in the use of a Stripped Dependency Tree (SDT) kernel for relationship identification among the named entities. Our empirical evaluation shows that SDT improves the precision (modestly) from 3% to 8% over an existing tree-kernel-based method proposed by Culotta and Sorensen [10] and also decreases the time taken to extract the entities and their relationships. GeoTagger uses a novel clustering-based spatio-spatial disambiguation algorithm based on the intuition that location entities extracted from the same part of a document are likely to be close to each other and thus form one (or a few) tight cluster(s).

1.1 FEMA Situation Reports

We use the FEMA National Situation Updates as the data source as a case study. The FEMA National Situation Updates contains reports from a variety of sources including US federal agencies, state and local government, and the news media. The updates are design to provide in use of emergency management planning and operational activities. Situation reports generally include location names indicate where the incidents happened. Sometimes persons or organizations involved in the incidents are also included in the reports.

1.2 Our Contributions

The key contributions of this paper are:

1. We design an information extraction system automatically processing text documents and create concept maps and spatio-temporal visualization. We demonstrate the usefulness of our system with the FEMA Situation Updates.

2. We present an improved dependency tree kernel for entity relation extraction. The experimental results show that our approach outperforms existing approaches.
3. We present a heuristic algorithm for ambiguous geographical names resolution. We use simple rules and a clustering algorithm to determine the coordination of locations.

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

2. RELATED WORK

RSOE HAVARIA AlertMap[4] is a world-wide disaster information system. The system reports real-time event updates for variety of incidents such as earthquake, active volcano, and tropical storms. However, unlike our system, RSOE HAVARIA AlertMap collects structured data from difference data sources. Every event has been classified and come with detail information such as timestamp and coordination. Although our system is focused on processing unstructured data, it would be interesting to integrate with structured data for 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-mail¹, and alerts from the World Health Organization and EuroSurveillance. HEALTHmap processes text and extracts disease names and the location names appear 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.*, [16] proposed using tree kernels over shallow parsing trees to extract person-affiliation and organization-location relations. They created data examples by performing shallow parsing over sentences. Compared with deep parsing techniques, the result from shallow parsing is more reliable. The kernels then compare similarity 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 [10] defined a slightly general version of tree kernels from [16] with a richer sentence representation. Their kernels are based on dependency trees instead of syntactic parse trees. Dependency trees include more information by considering syntactical 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.*, [12] combined dependency tree with shallow semantic parsing and reported average F1-score of 78.41% on ACE 2004 corpus. Greenwood and Stevenson [11] further extended the dependency tree kernels by using linked chain patterns and structural similarity measurement. Their approach can improve the performance over

¹The Program for Monitoring Emerging Diseases: <http://www.promedmail.org/>

iterations; nonetheless the maximum F-score is only 0.329 due to low recall.

Zhao and Grishman [18] described several combinations of syntactic kernels for relation extraction. They studied the polynomial combinations of five kernels to obtain both syntactic and local dependency information in sentences. Their evaluation shows that composition kernels could result better performance on the entity relation detection task. Zhang *et al.* [17] proposed a composite kernel which combines a simple entity kernel and a convolution parse tree kernel for relation extraction. They also defined relation instance spaces to limit the context information for computation. Their results indicate that by restricting context information, the performance of relation extraction could be further improved.

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 constrains and accumulating errors within pipeline processes, the performance improved significantly.

Bunescu and Mooney [6] 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. The shortest path example then will be compared 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.

Our approach combines the advantages of dependency tree kernels and that of the information elimination technique similar to [6]. We use tree kernels on 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 [10].

3. SYSTEM ARCHITECTURE

Our system consists with three main components: FactXtractor, GeoTagger, and a FEMA report visualization platform (FEMARepViz) [13]. Every component is designed as a standalone Web service to ensure system flexibility and interoperability.

FactXtractor is an information extraction Web service for Named Entity Recognition (NER) and Entity Relation Extraction (RE). FactXtractor processes text document using an open source text processing platform (GATE[1]) and identify entity relations using Stripped Dependency Tree kernels. Figure 3 shows the major steps of the processing flow. The output of FactXtractor is a concept map formatted in OWL. Concept map can be visualized with ConceptVista². We will discuss the methods used in FactXtractor in section 4.

GeoTagger is a geocoding Web service. The primary task for Geo-

²available at: <http://www.geovista.psu.edu/ConceptVISTA>

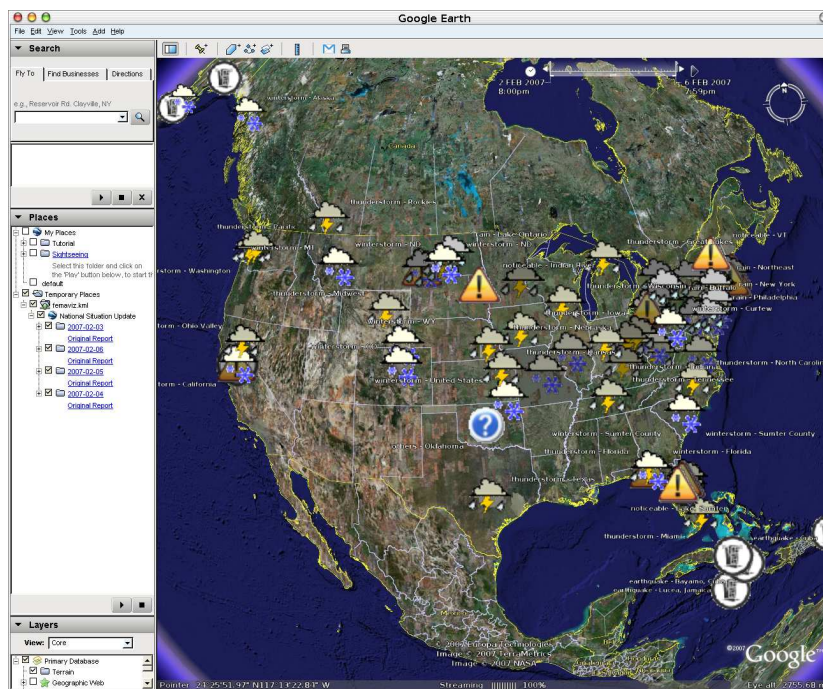


Figure 2: Processed FEMA Situation Updates from Feb 3, 2007 to Feb 6, 2007.

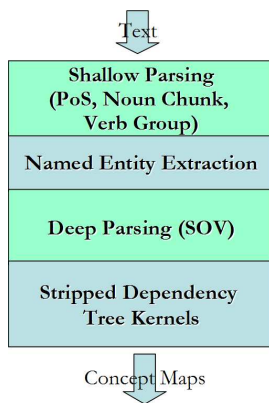


Figure 3: Text Processing Flow

Tagger is to resolve coordination or location entities. GeoTagger uses the U.S. Geological Survey (USGS) Geographic Names Information System (GNIS)³ for U.S. locations, National Geospatial-Intelligence Agency (NGA) GEOnet Names Server (GNS)⁴ for locations outside U.S., and Google Map for global locations. The reason we incorporate Google Map to the gazetteers is that GNIS and GEOnet contain many locations which are only used in local. By comparing the coordination given by Google Map, we can eliminate some locations in computation. We will discuss the details in section 4.

FEMARepViz is a visualization generation Web service for the FEMA Situation Reports. FEMARepViz processes situation reports using FactXtractor and GeoTagger. Processed reports are

³<http://geonames.usgs.gov/pls/gnispublic/>

⁴<http://earth-info.nga.mil/gns/html/index.html>

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 1 illustrate the overview of the system architecture.

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-base engine to extract named entities with gazetteers and language features such as Part-of-Speech. The named entities we extracted include person, location, organization, and time. We modify the build-in rules and gazetteers in ANNE to improve the results for US locations and organizations. Figure 4 illustrates a tagged text from the FEMA Situation Updates.

4.2 Entity Relation Extraction

We use an approach based on kernel methods for relation extraction. Kernel methods are widely used in a variety of machine learning problems with many popular algorithms such as Support Vector Machines[9] (SVM) and Perceptron[5]. Kernel methods are popular because of their simplicity: the only operation for the classification algorithms is computing dot products between pairs of data samples. Furthermore, one may replace dot products with a Mercer kernel, which maps feature vectors in \mathbb{R}^d into a new space \mathbb{R}^d [8]. More precisely, a kernel function is a binary function $K : X \times X \rightarrow [0, \infty]$ where X is the feature space.

4.2.1 Dependency Trees

A dependency tree is a tree representation of a parsed sentence that shows the relations between words [10]. 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, and so on. To generate dependency trees, we first

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. (Media Sources)

Figure 4: A segment of the FEMA Situation Updates with entities highlighted

process each sentence by NLP tools to obtain the dependency information, named entities and word features. We then generate a dependency tree for all pairs of named entities.

For example, Figure 5 is the dependency tree for sentence “John bought a house near Seattle”. For similarity measurement purposes, each node will be assigned with set of features. Typically, features include Part-of-speech (POS) tag, named entity tag, and so on.

To generate dependency trees, we first process each sentence by NLP tools to obtain the dependency information, named entities and word features. We then generate a dependency tree with Algorithm 1 for all pairs of named entities. The algorithm creates a tree structure by connecting each node with its head node. A head node is the first argument in a word-word dependency. Usually a node only represents one word except for noun phrases. We collect all words in a noun phrase into a single node.

4.2.2 Stripped Dependency Tree

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.

Based on this observation, we define a stripped dependency tree (SDT) as follows: a SDT is a subtree of a dependency tree. Every node in an SDT has at least one descendant node that is an argument of a given relationship. 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. Since we have to per-

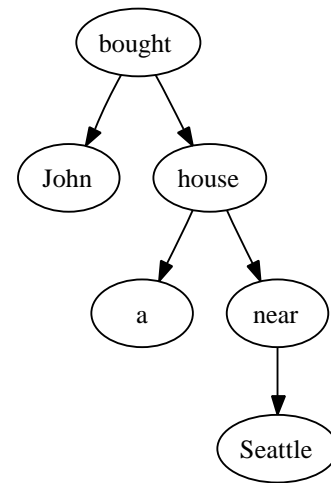


Figure 5: Dependency tree for “John bought a house near Seattle”

form 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 6 illustrates how a dependency tree could be converted to a stripped dependency tree.

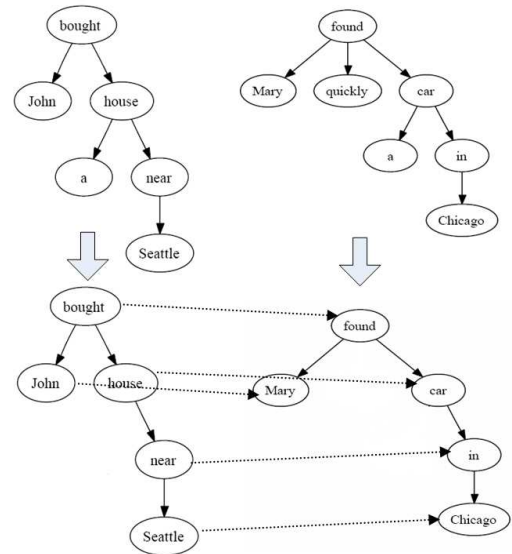


Figure 6: Converting SDT from a dependency tree

The advantages of an SDT are:

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 irrelevant nodes. SDTs should give better results on similarity measurement for entity relationship.

Algorithm 1: The Dependency Tree Generation Algorithm

Input : Sentence S , Token arg_1 , Token arg_2
Output : DepTree T

```

begin
   $V = r$ 
   $E =$ 
   $T = V, E$ 
  Parse  $S$ 
  for Each word  $w$  in  $S$  do
    construct node  $n$  for  $w$ 
    if word  $w$  has a headword  $w^h$  then
      construct node  $m$  for  $w^h$ 
      if  $n$  is not in  $V$  then
        collect features for  $n$ 
        add  $n$  to  $V$ 
      if  $m$  is not in  $V$  then
        collect features for  $m$ 
        add  $m$  to  $V$ 
      if  $n$  is part of a noun phrase start with  $m$  then
        merge  $n$  into  $m$ 
      else
        add an edge  $e = (n, m)$  to  $E$ 
    for Each  $n$  in  $V$  has no incoming edge do
      add  $e = (r, n)$  to  $E$ 
  do a depth first search on  $T$ , find the minimum subtree  $T$ 
  contains both  $\text{arg}_1$  and  $\text{arg}_2$ 
  return  $T$ 
end

```

We use MINIPAR to parse each sentence; we then generate a dependency tree for the sentence using output of MINIPAR. For example, the MINIPAR output of a sentence “John bought a house near Seattle” will be:

```

(
E0 ((      fin C      *      )
1  (John   ~ N      2  s      (gov buy))
2  (bought buy V    E0  i      (gov fin))
E2 ((      john N   2  subj  (gov buy) (antecedent 1))
3  (a      ~ Det   4  det    (gov house))
4  (house  ~ N     2  obj    (gov buy))
5  (near   ~ Prep  4  mod    (gov buy))
6  (Seattle ~ N    5  pcomp-n (gov near))
)

```

MINIPAR can achieve about 88% in precision and 80% in recall for determining dependency relationships⁵. By linking each word with its head node of the MINIPAR output, the dependency tree for the given sentence can be created (Figure 5).

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.

For each node, we assign a set of features generated by GATE. We used the features used in [10]. To improve the performance, we added some additional features. They are orthography, word root, verb voice, verb negation, and synonyms obtained from WordNet.

⁵<http://www.cs.ualberta.ca/~lindek/minipar.htm>

Table 1: 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)

Table 2: 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

Table 2 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.

We have implemented the tree kernel algorithms in Java and used it in an SVM. We augment the LibSVM[7] implementation to use the tree kernels. We use a CoNLL 04 corpus⁶ 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 1 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 our improved tree-kernels results with the dependency tree kernels in [10] and 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 of positively annotated relations. We use the following equation to compute the F_1 score.

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

Table 3 shows the performance comparison between F_1 scores for both LP and SVM with tree kernels using 5-fold cross-validation.

⁶available at <http://l2r.cs.uiuc.edu/~cogcomp/Data/ER/conll04.corp>

Table 3: Comparison of Relations Classification Results demonstrating the superiority of our tree kernel + SDT method over the LP method (the best results from [15]).

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

The results indicate that tree kernel approaches outperform the linear programming approach for all relations. The tree kernel with SDT further improved the original dependency tree kernel described in [10]. As shown in 3, combining contiguous tree kernel with SDT didn't improve the precision very much. However, the recall of the tree kernel with SDT is 3% to 8% better than the original dependency tree kernel. It shows by reducing noise introduced by unnecessary nodes in dependency trees, the tree kernel can perform 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 in [10]. The performance didn't vary much nor improve with different λ values.

4.3 Geo-coding

Grounding a location name with a correct coordination is challenging. Essentially, we want to disambiguate places with the same name such as Springfield, IL and Springfield, PA. If additional information is not available, the geocoding for Springfield becomes arbitrary. In our system, a dedicate component called GeoTagger is used for geocoding tasks.

GeoTagger takes a short text as input then determines a geographical scope for the text. Geographical scopes are determined based on the locations with higher certainty (e.g. common sense). For example, United States and Europe can result larger a geographical scope than Pennsylvania and Georgia. Determining the geographical scope for a text can eliminate unlikely candidate locations. We categories geographical scope as four levels: World, Continent, Country, and Province/State. For each text segment, the highest level of geographical scope will be used to select candidate coordination. 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 selected. The coordination of location names are obtained by querying gazetteers. We use GNIS for U.S. locations, NGA GEONet for locations outside U.S., and Google Map for global locations.

Within a geographical scope, each location name could have several coordination candidates. Intuitively, GeoTagger then finds possible clusters with smallest distance between locations. The clustering algorithm works as follows. We use Euclidean distance as the distance metric.

Note that although our clustering approach performed reasonably well to ground most common locations, it could still generate false positives. In Fugire 4, the word "South" should refer to the South of the United States. However, our system will identify it as "South, KY" because it is in the GNIS database and close to other locations.

4.4 Topic Classification

Algorithm 2: Algorithm for location selection

Input : Geographical scope GS , Set of location names N , Maximum number of clusters K , threshold ω

Output : Map of coordinations M

begin

```

 $M \leftarrow \emptyset$  for each  $n \in N$  do
   $Coors_n \leftarrow$  coordinations within  $GS$  match  $n$ 
  for  $k = 1$  to  $K$  do
    Arbitrarily select  $coor_0 \in Coors_n \rightarrow M(n)$ 
     $Coors_n = Coors_n - \{coor_0\}$ 
     $D_k = \text{k-means}(k, M)$ 
    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

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 $c^* \in C = \{c_1, \dots, c_{|C|}\}$ that has the largest posterior probability given the text with the following equation.

$$c^* = \arg \max_{c \in C} \{\Pr(c|D)\}$$

where D is the text segment. We use the DynamicLMClassifier implementation in LingPipe[3]. Texts segments will be classified into one of nine categories commonly appear in FEMA situation reports. The categories include disease, noticeable, snow, wild-fire, 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.

4.5 Geo-temporal Visualization

Finally, the extracted concept map is generated as an OWL file and display with ConceptVista (Figure 7). Geo-coded incidents are generated in Google Earth KML format for daily reports and upload to Google Earth by a Network Link. A timestamp is added to each incident indicates the temporal relationship between incidents. Users can move the time-slide wedge to change the timeline and exam the incidents happened during the time period around the world (Figure 8).

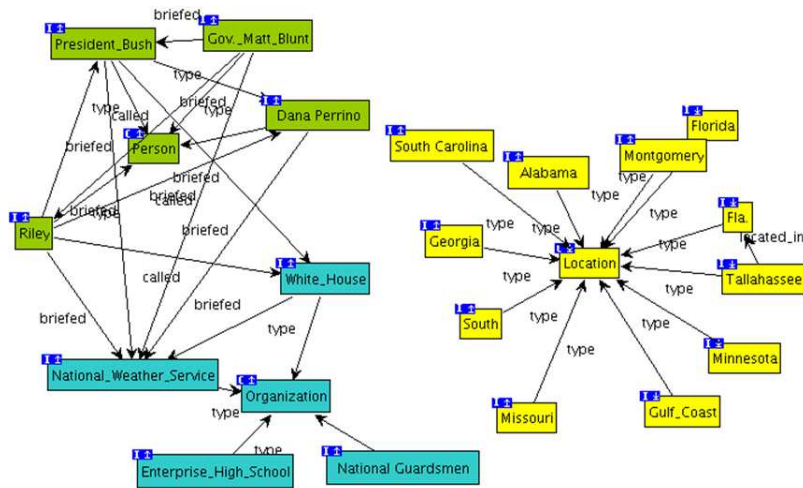


Figure 7: The Concept map generated from the segment of the FEMA Situation Updates shown in Figure 4



Figure 8: Geo-temporal visualization

5. CONCLUSION AND FUTURE WORK

We design an automated information extraction system to support analysis of vast amount of textual data. We use the FEMA Situation Updates to demonstrate the usability of our system. We use FactXtractor to extract named entities and links entities with syntactic and semantic relations. Concept maps can be created to visualize entity relations. Furthermore, we use GeoTagger to resolve geographical ambiguity and create geo-temporal visualization with FEMARepViz and Google Earth.

Since no computational approach can achieve human-level accuracy for complex information extraction tasks, visualization could be a solution to bridge the gap between fully automated system and manual data processing. Our system can be easily adapted for different type of text documents by providing proper training sets. We believe that our system can benefit users who have needs to analyze massive geo-temporal information in an efficient manner.

In the future, we plan to build statistical analysis modules on top of our information extraction system. Currently, two types of forecasting modules are under development.

Periodic Incidents Forecasting: Periodic incidents can be pre-

dicted with time series analysis such as moving average or exponential smoothing. Incidents periodically appear in FEMA 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 conditional occur after other incidents. For example, flood may happen in some region after heavy rain. Tornadoes may cause by unusual heat or thunderstorms. Such correlation can be found in historical reports.

Moreover, we plan to add human sensor data with varying reliability and types of events including ProMED-mail, Global Disaster Alert and Coordination System, news media, and Internet blogs.

Acknowledgements

This chapter has been partially funded by the National Visualization and Analytics Center (NVAC) and partially funded and co-authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the U.S. Department of Energy. NVAC is 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. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Governme.

6. 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 AlertMap. <http://hisz.rsos.hu/alertmap/woalert.php?lang=eng>.
- [5] 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.

- [6] 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.
- [7] 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>.
- [8] M. Collins and N. Duffy. Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems 2001*, 2001.
- [9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
- [10] A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, 2004.
- [11] 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.
- [12] 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.
- [13] C. Pan and P. Mitra. Automatic extraction and geo-spatial visualization of fema national situation updates. *First Annual DHS University Network Summit on Research and Education*, Poster, March 2007.
- [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] D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106, 2003.
- [17] M. Zhang, J. Zhang, J. Su, and G. Zhou. A composite kernel to extract relations between entities with both flat and structured features. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 825–832, Morristown, NJ, USA, 2006. Association for Computational Linguistics.
- [18] 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.