University of Alberta

Converting Textual Documents to RDF Triples,
Covering Syntactic and Semantic Structures

by

Kimia Hassanzadeh

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering

Fall 2013
Edmonton, Alberta

Dedication

This thesis work is dedicated to the most wonderful people in my life. My parents, Behjat Tadayon and Heshmatollah Hassanzadeh for their unconditional love and support and always believing in me. And my brother, Bardia Hassanzadeh, my best friend from the very first moment of my life and I will never be able to thank him enough for always being there for me.

Abstract

An important contribution of the Semantic Web is a new format of data representation called Resource Description Framework (RDF). In RDF every piece of information is represented by a triple: <subject-property-object>. RDFs are densely interlinked between each other and are becoming very popular format of representing data on the web. As of August 2011, the last available data, more than 31 billion of triples exist on the web.

In this set of work, we propose a system for information extraction from plain text in form of RDF triples. The proposed method is independent of prior knowledge-base and domain-specific patterns, and is applicable to any textual resources. Our approach is capable of identifying grammatical structure of an input sentence and analyzing its semantic to generate meaningful RDF triples of information, readable by human users and software agents. Through several experiments, we evaluate this approach by demonstrating the quality of our results.

Acknowledgement

I wish to thank my supervisors Dr. Marek Reformat and Dr. Witold Pedrycz who were more than generous with their expertise, precious time and patience throughout the entire process. I would also like to thank Dr. Jie Han and Dr. Ergun Kuru for agreeing to serve on my committee.

And finally, my special gratitude to Electrical and Computer Engineering department of University of Alberta for their endless friendly supports and assistance.

Table of Contents

List of Tables

List of Figures

1. Introduction

The invention of web is one of the key catalysts of a revolution in storing, retrieving, transmitting and manipulating data, or in another word "Information Technology". Data modeling started with traditional connected tables, or databases, is being developed and evolved to become smarter and more self-describing. New markup languages lead to a better – more understandable and more expressive – data presentation, not only for human but also for machines. However, this means that any type of data should follow a common framework; and this is quite difficult in the case of textual resources such as reports or articles.

There are a number of challenges that need to be addressed to fully capture the essence of a text and translate it into machine-readable representation. This task is not trivial since the extracted information must reflect not only a conceptual level of a text, but also its grammatical structure. In other words, grammar structure of a text directly affects comprehension or conceptual level, of that text.

By incorporating two Natural Language Processing (NLP) tools *Stanford*[1] and *Senna*[2], we propose a method for translating plain text documents into a machine-readable format which covers both syntactic/grammatical and semantic/conceptual information.

In the remainder of this Chapter, main concepts are briefly introduced along with the motivation. Chapter 2 covers important topics and summarizes related works in this field. Chapter 3 discusses technical aspects, tools used in this project and full description of process. Case studies and a set of experiments with detailed analysis and evaluation are provided in Chapter 4. And finally, in Chapter 5 potential future works and conclusion are discussed.

1.1. Semantic Web and Linked Data

"Semantic Web" was introduced by Tim Berners-Lee, which entered new concepts into web technology including Resource Description Framework (RDF[3]) and Web Ontology Language (OWL[4]). The web of unstructured documents is converted into a

[1] A NLP tool for studying grammatical structure of a sentence; such as subject and verb

[2] A NLP tool for studying semantic structure of a sentence; such as experiencer, patient or something that undergoes an experiment, time and location

[3] A framework for representing information by three elements (subject , relation , object)

[4] A logic based language used in Semantic Web for representing knowledge

web of structured data. Web of structured data is the ultimate goal to achieve, where software agents are able to follow links of a type domain-to-domain, mine information, and answer queries. [Heath & Bizer, 2011]

Companies are encouraged to publish their data in public. The idea is referred to as "Linked Data" and according to "Linking open-data community project" the number of interlinked data has vastly increased to almost 30 billion of RDF links. DBpedia[5], FOAF[6] and GeoNames[7] are some of the most famous datasets that have joined Linked Data Cloud project. [Heath & Bizer, 2011]

Linked Data aim is to connect semantically structured datasets all across the web so machines can interact and share knowledge with each other or answer various queries. In order to have documents interlinked, there are a few principals to follow such as identifying concepts, their properties, the relationships among them and building proper links to the concepts from other datasets. [Heath & Bizer, 2011]

The first step in publishing data as a part of Linked Open Data (LOD[8]) project is recognizing main concepts in dataset along with their features. This step is also known as building the Graph Database. Data is being modeled as triples of three elements <subject - predicate – object> or RDF triple. Predicate or property describes a feature of its subject and object is the value of that feature.

Here is a simple example of converting a sentence into a RDF triple: "Albert Einstein was born in Ulm", <**Albert Einstein** as subject - **born** as predicate - **Ulm** as object>. As this process continues, more predicates are attached to Albert Einstein, like his birth-date, study-field, residence and etc. This branching grows for the other concepts too, which eventually turns the dataset into a graph of nodes as subjects or objects, and edges as predicates.

[5] A knowledge base of extracted information from Wikipedia for Semantic Web
[6] Friend Of A Friend is a Semantic Web structure for describing people including first name, last name and etc.
[7] Geographical database in Semantic Web for describing locations
[8] LOD project is cloud of connected datasets which is designed for query purposes

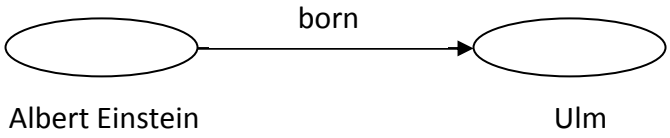

Figure 1 subject-predicate-object, example

The next step is to provide Unique Resource Identifiers (URIs), for all concepts and features. URIs are necessary for network interactions via http protocols. There are some specific schematics for building well-formatted URIs in a domain.

In the previous example:

- "Albert Einstein" has a URI: http://dbpedia.org/page/Albert_Einstein
- "Born" as a property has a URI: http://dbpedia.org/property/birthPlace
- "Ulm" also needs a URI : http://dbpedia.org/page/Ulm.

  "Ulm" can further develop more connections because it is a city and can connect to another dataset describing its population, geographical location and so forth.

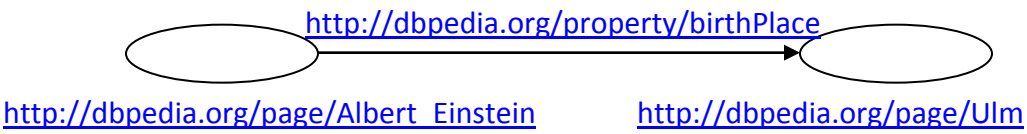

Figure 2 valid URIs for subject-predicate-object, example

The third step, URIs must lead to valid information based on RDF standard. Depending on the user request in an http header, server responds using different formats of data such as HTML, RDF/XML, N-Triples, turtles or Notation3. Once a human user requests a URI about Albert Einstein, server responds with HTML format whereas a machine-readable format is retrieved as the response to a SPARQL-based query. This process is well known as content negotiation. [DuCharme, 2011]

The final step is establishing links to other available datasets. As mentioned before, many datasets are public and new datasets can be constructed using their URIs. For example, for a triple <Albert Einstein is a person>, the term "person" is a general

concept defined in the FOAF dataset. Therefore, it is recommended to use the URI pointing to “person” from FOAF rather than creating a new URI pointing to a new definition of “person”.

In Figure 3, rdf:type is a property which is defined in RDF domain, foaf:Person is also a predefined class “Person” in FOAF domain. Creating linked structured data requires basic understanding of Semantic Data Modeling which is covered in Section 2.1.

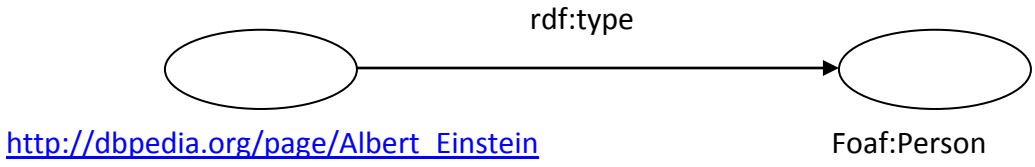

Figure 3 using available structure, example

1.2. Motivation

In this project, our main goal is converting a plain text to units of information understandable for machine. These units of information are RDF triples extracted based on both syntactic and semantic structure of the text. Syntactic structure, or grammar, elucidates skeleton of the sentences such as verbs, modifiers, subjects, objects and etc., as well as phrasal chunks including noun phrase, verb phrase and etc. On the other hand, semantic structure expresses the conceptual level of a sentence in form of “Who” does/did “What”, “Where”, “When”, “How”, “Why” and etc.

Once, all information is available, we analyze it and generate meaningful triples. Triples representing a context can be used as a basic level of understanding for machines. It proceeds to more complicated tasks such as querying and event detection.

1.3. Generating RDF from Text

The RDF standard format consists of three elements, subject, predicate and object. Therefore all pieces of information extracted by syntactic parser and semantic parser must be translated into this format. This requires analysis and understandings of the outputs generated by syntactic/semantic parsers.

For example, a noun-phrase contains modifiers and a head-noun. If this noun-phrase has relations to other phrases or has a semantic role then the role and relation are better to be connected to head-noun other than modifiers. Otherwise, a very complex

mash of triples might be created which would lead to capturing useless or pointless information.

As explained in the example above, we suggest finding main concepts of text and treating other terms as extra/more information that are being connected to the main concepts. This approach also provides a comprehensive graph representation as a set of connected nodes. Our analysis is thoroughly explained in Section 2.3.

2. Background and Related Work

2.1. Data Modeling and RDF

Data Modeling has been a potential area of interest for software engineers, as it plays an essential role in information systems. Sharing and distributing data require models to have an efficient and accurate structure; most of data representation models are categorized under two main groups of modeling-techniques: top-down and bottom-up models. [Francesconi et al. 2010]

In bottom-up technique, modeling starts from unprocessed data and system/user requirements, which makes the model specific to one application, the model then grows in terms of complexity and completeness.

However, top-down strategy breaks the system down into subsystems by asking users about the abstract-system and the process continues iteratively until base elements are reached. Entity Relation Model (ERM) represents data by entities and their relations, which is known as a top-down modeling technique. In ERM, models are classified into three levels of physical, logical and conceptual data model. The first level handles database tables and their foreign keys connections; this level comes as a support for logical level. More details are added to the model's entities in logical level including the relations among entities. Conceptual level, on the other hand, doesn't cover every details and focuses on the whole extent of information to be represented in the model. [Chen,1975]

ERM levels give a comprehensive demonstration of entities, relations and more importantly hierarchy of the data which is useful for building a more conceptual representation of model called "Semantic data modeling". In semantic data modeling approach, a concept is defined by its relations to the other concepts, known as its attributes. Ability of embedding definitions within the data model enables data to be distributed among applications with their meanings.

Semantic web was originated by one core idea that a word can reveal its semantics through the establishment of relationships, likewise discussed in ERM. A keyword "building" in a web document does not show what meaning it is referring to, either a construction or process of creating something. Whereas connecting this keyword to other keywords such as "architect", "plan" and "construction site" declares more details on it. [Hebeler, Fisher, Perez- Lopez, & Dean, 2009]

Semantic data modeling, also known as a conceptual description of knowledge, has its own modeling syntax, RDF. A typical RDF file consists of statements about resources, or concepts, of the model. Statements are simple triples of (subject - property - object) and a collection of all RDF triples represents a directed graph. [DuCharme, 2011]

Modeled data might be available in different versions. RDFa and RDF/XML are two standard formats confirmed by W3C, there are also other non-standard formats such as N-triples and Turtle. Here is a brief explanation of each format along with their pros and cons:

- RDF/XML: This method expresses the graph in XML syntax. Although it is readable by machine, it never became popular. Its complex structure makes it difficult for human users to follow; Human intervene is sometimes necessary for data management purposes. [Heath & Bizer, 2011]
- N-Triples: Each statement is represented as a line of three URIs and each URI is inside of an angle bracket, lines are separated by dots. It is one of the easiest RDF versions, yet it lacks hierarchical format for resources. Because, each line contains one piece of information about a resource and gathering all pieces of information about that resource takes extra effort and text processing. [DuCharme, 2011]
- Turtle: Unlike N-Triples, turtle brings all relevant pieces of information under each resource category. Prefixes are summarized at top of the file and redundant URIs are omitted which makes it easier for human user to understand the content. This approach is a combination of RDF/XML and N-Triples. [DuCharme, 2011]
- RDFa: It is one of the most popular methods among RDF serializations as triples are embedded within HTML or XML document. So instead of having separate machine versions and human versions, all data is inside one content. Still software agents are needed to extract RDF data. [Heath & Bizer, 2011]

The desired format is specified in Http-header-request by user or is chosen by server automatically. This is called Content-Negotiation process and handles HTTP requests whenever different formats of a file exist in one single URI. Following is an example on how servers deal with content negotiation:

Example: Content Negotiation

"Dave Smith" is a target concept to explain in this example. There is a single URI containing 2 types of references; they are pointing to an "About Page" information with HTML and RDF/XML formats:

http://biglynx.co.uk/people/dave-smith.html

http://biglynx.co.uk/people/dave-smith.rdf

The extension is specifying what type of data is requested. Both of the accesses are attempted to 1 URI and this is where Content negotiation, or 303 redirect, handles the situations by reading the accept header. But, there is one more URI without any extensions, known as URI for real world object:

http://biglynx.co.uk/people/dave-smith

This option is provided for the server to automatically decide what type of document to send to the user. If the user is accessing the address via browser, server assumes it is a human user and redirects him to html document.

According to Linked Data principals every concept in the model has a URI which must contain valid information [Heath & Bizer, 2011]. Separating triples which represent properties, classes from those representing instances is a mandatory step in this regard as it helps the model to be similar to its real world entity. Hence there are certain schemas for simulating concepts in semantic web such as RDF Schema (RDFS), OWL, Simple Knowledge Organization Systems (SKOS), FOAF and etc.

RDF and RDFS provide a set of basic classes and properties for describing new vocabularies in the system. Rdfs:Class, Rdfs:Property, Rdfs:Literal, Rdfs:domain and Rdfs:range are some useful classes and properties for declaring a concept.

Example: RDF/RDFS property and class

Dbpedia:Person rdf:type rdfs:Class

"Person" is a concept in Dbpedia dataset and the type of this concept is "Class" which makes it different from property or an instance of another class.

OWL helps explaining the definition of group and relations such as Owl:description, owl:sameAs and Owl:seeAlso.

Example: OWL property

Dbpedia:Albert-Einstein owl:sameAs http://data.nytimes.com/49783928729941204213

This triple is using owl:sameAs to show these two resources, subject and object, are the same.

FOAF describes people and their relationships like foaf:name, Foaf:mbox, foaf:knows and etc.

Example: FOAF property

Dbpedia:Albert-Einstein foaf:givenName "Albert Einstein";
foaf:name "Albert";
foaf:surname "Einstein".

The example is showing how to define a person's full name, first name and last name with triples.

SKOS and DCterms are other groups of standards for explaining hierarchy of classes, sub-classes, properties, sub-properties and so forth.

So providing more info via these triples builds up more connectivity with other datasets. Standards facilitate the query process; a SPARQL query script starts searching a RDF document, RDFS:label and RDFS:comment act as meta data as they contain extra information about the document content. [DuCharme, 2011]

RDF data modeling along with Linked Data creates a platform for web applications to play with datasets which are available throughout the web. Query languages have been around since the early emergence of databases, such as SQL, for information mining purposes as well as managing the data by adding new data, updating or removing it.

SPARQL is a query language designed for RDF datasets in 2004. It accepts standard query scripts and returns various formats of outputs, like a table, RDF triples and XML. One of the main benefits of SPARQL is machine-to-machine communication, which initiated the idea of communication SPARQL endpoints. [DuCharme, 2011]

SPARQL endpoint is an interface connected to a RDF triple-store; It accepts queries across the web and returns output. Endpoint is also referred to as a processor or a service as it is in charge of processing a received query over the dataset. Any endpoint has a HTTP address and follows SPARQL protocol. Many organizations have provided SAPRQL endpoints to their public datasets, so other web apps can interact with their

data. There is a list of all these organizations in W3C official website[9] [Hebeler, Fisher, Perez- Lopez, & Dean, 2009]. For example, a web application can connect to a movie database SPARQL endpoint and query for an actor and all his movies in 90s. Text-Processing section discusses how text processing is involved in understanding a query and what pieces of information can be obtained by parsing a text.

For all the advantages discussed so far, data must be represented semantically. Ontology is another term used for semantic data modeling; some ontology models are created by organizations manually, with help of experts of various domains designers. Once the model is built other developers are encouraged to reuse these existing ontologies for the sake of consistency. Ontology development is propelling toward automatic extraction of data; so depending on the origin of data-domain, approaches may differ. For example, creating ontology from a relational database is different from data mining from texts.

2.2. Text Analysis

Enabling machines to understand human-written materials is the major role of semantic web. But transforming an unstructured text into meaningful data requires parsing the text, identifying words and describing semantics for each word. Not only is it useful for querying documents on web, but also provides rich datasets for machine-learning tools to learn and predict future events.

Most of the methods for transforming a text into modeled data employ OL&P approach; which consists of two general phases of TBox and ABox production. In TBox, all concepts and relations are extracted from the domain and hierarchy of data is derived; at this step the backbone of ontology is formed. Whereas, ABox populates the extracted framework with instances discovered from that data. [Biemann, 2005]

For building TBox or backbone of the model, some methods focus on finding subject, verb and object of a sentence and then translate them into a triple. Harris' distributional hypothesis suggests looking for specific patterns of words within a sentence, because some groups of words are likely to occur together. For example, Hearst patterns help identifying "*A, B and other C*, or *A such as B and C"* patterns in a text, these patterns are translated in "is-a" or "part-of" relations. Other methods look for certain verbs or names and establish links to relevant resources such as name of organizations or famous

[9] http://www.w3.org/wiki/SparqlEndpoints

persons. Once the structure is agreed upon, text is scanned for populating the model matching the structure, which is ABox phase[10]. [Biemann, 2005]

All these efforts are made to reach one important goal which is extracting information. IE[11] can be considered as a series of actions human take to understand a text. The very first step for human to capture the message of text is being aware of the definition of each word individually. Then, he starts learning the grammatical structure of each sentence to find subject, verbs and other part of speech tags. At this level he is able to recognize the roles of each term or phrases in a sentence. But all these steps occur in sentence-level, for mining more information sentences must reveal their logical connections. [Mangassarian& Artail 2007]

The similar process is achievable for machines; and the focus is on 2 main types of dependencies that exist in a sentence, grammatical dependencies and semantic dependencies.

Grammatical dependencies reveal syntactic relations such as subject, object, direct object, adverbs, adjectives, phrases, coordinating conjunctions and too many other typed dependencies [Marneffe & Manning, 2008]. This approach focuses on analyzing the text from a linguistic point of view where grammatical relations are representing text structure. Grammatical relations appear with a set of base standards in any languages; therefore many grammar parsers use machine-learning techniques to learn these patterns from various text corpuses. Time efficiency, robustness and high accuracy are some of the main factors for evaluating these parsers [Marneffe, MacCartney, & Manning, 2006]. *Stanford* Parser is syntactic parser used in this project and it is discussed in section 2.3.

Semantic dependencies expose a different view of the text, in which a sentence breaks down into a frame of roles, "*who did what to whom, when, where, why, how and so forth"*. This method targets the verb(s) in a sentence and then seeks for all semantic roles of words or group of words relating to that verb(s) [Gildea & Jurafsky, 2002]. Information Extraction, Question Answering and Text Summarization are some of the NLP fields that need semantic structure for their analytical purposes. [Pradhan, Ward, Hacioglu, Martin, & Jurafsky, 2004] *Senna* Parser is the semantic parser used in this project and it is discussed in section 2.3.

[10] More discussion provided in 2.4
[11] Information Extraction

2.3. Tools and Systems

As parsers are the backbone of this code, they are discussed in more depth through this section; examples are provided. Input, output are explained as well as technical aspects of training phase for each parser such as training corpora, machine learning approach.

2.3.1. Syntactic Parsing

Stanford parser is a statistical NLP tool, written in Java, for generating grammatical structure of a raw text. This system is able to check over 50 grammatical relations and produces different formats of outputs and linguistic analysis. It is trained on the *Penn Wall Street Journal* Treebank and supports many languages such as English, Chinese, German, Arabic and French. Output contains different levels of dependency graphs and trees for various purposes.

Any system that uses machine-learning algorithms has to pass a training phase at its initialization step. Training-data used in this parser is a part of Treebank corpus, *Penn Wall Street Journal* Treebank. Treebank refers to a text corpus that is syntactically parsed or annotated, which usually has a tree form, *Penn-TreeBank* annotates phrasal tree structure of a text. [Marneffe, MacCartney, & Manning, 2006]

Annotating a text is simply labeling words with their grammatical title such as noun, verb, adjective or adverb. In linguistic analysis, this process is known as Part-Of-Speech, or *POS* tagging, which requires a preprocessing step on the raw text. In preprocessing, parser starts splitting the text into sentences and finding their tokens including numbers, punctuations, and all other words. Then, text is passed to *POS* tagger for generating the first layer of annotations for each token[12] [Cunningham, 15 April 2011].

As *Stanford* Parser is trained on *Penn Wall Street Journal* corpus Treebank, it is able to mine phrasal structure. It runs pattern matching algorithms against parsed-text to discover grammatical dependencies between pairs of words. These dependencies are referred to as *Stanford Dependencies*, or SD, and represent the text structure as a directed graph. [Marneffe & Manning, 2008]

Example[13]: a sample *POS* tagging result

Input: "Bell, based in Los Angeles, makes and distributes electronic, computer and building products."

---

[12] A list of POS tags is provided in Appendix III.

[13] Generated by *Stanford* parser online demo version ; Full list of POS tags in Appendix III

***POS* tagging**: Bell: NNP[14], based: VBN[15] , Los: NNP, Angeles: NNP, makes: VBZ[16], and: CC[17], distributes: VBZ, electronics: JJ[18], computer: NN[19], and: CC, building: NN, products: NNS[20]

According to [Marneffe & Manning, 2008], *Stanford* Parser tests 53 grammatical relations on all phrases of a sentence and assigns a head for each phrasal structure; then, chooses one root among all these heads. Here is a brief explanation of studying a phrase: parser takes a phrase, like a noun-phrase, and searches for the head word, abbreviations and different types of modifiers such as temporal modifiers, relative clause modifiers, possessive modifiers and etc. Sometimes, parser doesn't find any matches for a relation and leaves it as a dependent.

From a general view, *Stanford* parser splits the sentence into two parts, root and dependents; All *Stanford Dependencies* go under the category of dependents[21].

```
(ROOT
  (S
    (NP
      (NP (NNP Bell))
      (,,)
      (VP (VBN based)
        (PP (IN in)
          (NP    (NNP    Los)    (NNP
Angeles))))
      (,,))
    (VP (VBZ makes)
      (CC and)
      (VBZ distributes)
      (NP
        (UCP  (JJ  electronic)  (,,)
(NN computer)
          (CC and)
          (NN building))
        (NNS products)))
    (..)))
```

[Marneffe & Manning, 2008]

[14] Proper noun, singular
[15] Verb, past participle
[16] Verb, 3rd person singular present
[17] Coordinating conjunction
[18] Adjective
[19] Noun
[20] Noun, plural
[21] Definition of dependencies in Appendix I, Full hierarchy of dependencies in Appendix II

Example: a sample *Stanford* output (continuing previous example)

Input: *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products."*

Output: Figure 4

Stanford Parser has four representation styles, the basic dependencies tree, collapsed dependencies graph, and the propagated versions into two forms, a graph and a tree. Each representation style is discussed through an example; styles are also compared with each other:

Example: Basic typed dependencies (continuing previous example)

Input: *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products. "*

Output: Figure5 Basic typed dependencies; Figure6 Collapsed typed dependencies; Figure7 Collapsed typed dependencies with propagation of conjunctions; Figure8 Collapsed dependencies preserving the tree structure.

Figure5 shows basic typed dependencies; basic model contains dependencies discovered from phrase grammar structure. This level of dependencies limits each word to be only dependent to one other word, which means there is no cycle in the dependency graph and it makes it a tree. [Marneffe, MacCartney, & Manning, 2006]

- nsubj (makes-8, Bell-1)
- partmod (Bell-1, based-3)
- prep (based-3, in-4)
- nn (Angeles-6, Los-5)
- pobj (in-4, Angeles-6)
- cc (makes-8, and-9)
- conj (makes-8, distributes-10)
- amod (products-16, electronic-11)
- conj (electronic-11, computer-13)
- cc (electronic-11, and-14)
- conj (electronic-11, building-15)
- dobj (makes-8, products-16)

Figure 5 Basic typed dependencies

Another set of dependencies is then added to the basic graph, in Figure6, and often cycles are created in the graph. These dependencies try to simplify the model by merging some of the basic relations that result in having direct dependencies between content words. Conjuncts, prepositions and relative clauses are three target dependencies in this process. The resulting graph is called the collapsed dependency graph. [Marneffe & Manning, 2008]

- nsubj (makes-8, Bell-1)
- partmod (Bell-1, based-3)
- nn (Angeles-6, Los-5)
- prep_in (based-3, Angeles-6)
- root (ROOT-0, makes-8)
- conj_and (makes-8, distributes-10)
- amod (products-16, electronic-11)
- conj_and (electronic-11, computer-13)
- conj_and (electronic-11, building-15)
- dobj (makes-8, products-16)

Figure 6 Collapsed typed dependencies

Table1 compares basic model and collapsed model; pairs on the left, from basic dependencies, are merged together and created the pairs on the right side of the table, collapsed dependencies.

Table 1 Basic Dependencies Vs. Collapsed Dependencies

Basic	Collapsed
prep(based-3,in-4) pobj(in-4, Angeles-6)	prep_in(based-3, Angeles-6)
cc(makes-8, and-9) conj(makes-8, distributes-10)	conj_and(makes-8, distributes-10)
conj(electronic_11, computer-13) cc(electronic_11, and-14) conj(electronic-11, building-15)	conj_and(electronic_11, computer-13) conj_and(electronic-11, building-15)

The next style, Figure7, involves extending some of the dependencies, in order to add more information to the model. In this phase, *Stanford* Parser analyzes all conjunct dependencies from the collapsed version and generates additional dependencies for conjunct couples. Considering w1 and w2 as two words in a sentence with "and" dependency, conjunct_and(w1,w2), all existing dependencies for w1 are copied for w2. [Marneffe & Manning, 2008]

- nsubj (makes–8, Bell–1)
- nsubj (distributes–10, Bell–1)
- partmod (Bell–1, based–3)
- nn (Angeles–6, Los–5)
- root (ROOT–0, makes–8)
- prep_in (based–3, Angeles–6)
- conj_and (makes–8, distributes–10)
- amod (products–16, electronic–11)
- conj_and (electronic–11, computer–13)
- conj_and (electronic–11, building–15)
- dobj (makes–8, products–16)
- dobj (distributes–8, products–16)

Figure 7 Collapsed typed dependencies with propagation of conjunctions

Table2 shows additional dependencies added to model. The left-side pairs, collapsed dependencies, are the reasons why the right-side pairs, collapsed version with propagation of conjunction, are generated.

Table 2 Collapsed dependencies Vs. Collapsed dependencies with propagation

Collapsed	Collapsed with propagation of conj
nsubj(makes-8, Bell-1) conj_and(makes-8, distributes-10)	nsubj(distributes-10, Bell-1)
Dobj(makes-8, products-16) conj_and(makes-8, distributes-10)	dobj(distributes-10, products-16)

In the last version of dependencies graphs Figure8, tries to maintain additional dependencies but as a tree. So it omits all dependency pairs which break the tree structure. Here is an example of sentence being processed in different version [Marneffe & Manning, 2008]:

- nsubj (makes-8, Bell-1)
- partmod (Bell-1, based-3)
- nn (Angeles-6, Los-5)
- root (ROOT-0, makes-8)
- prep_in (based-3, Angeles-6)
- conj_and (makes-8, distributes-10)
- amod (products-16, electronic-11)
- conj_and (electronic-11, computer-13)
- conj_and (electronic-11, building-15)
- dobj (makes-8, products-16)

Figure 8 Collapsed dependencies preserving the tree structure

The tree version of collapsed dependencies for this example is identical to the collapsed dependencies, Figure8, dobj(distributes-10, products-16) and nsubj(distributes-10, Bell-1) are eliminated. [Marneffe & Manning, 2008]

2.3.2. Semantic Parsing

Syntactic parsers do not provide any explanation about the meaning or implication of the text; they only deliver the backbone of a sentence grammatically. For complicated NLP tasks such as query answering, processor needs to analyze verbs, participants, time and places for each sentence in order to reply back various questions. This process is called Semantic Role Labeling or Shallow Semantic Parsing. The general scenario of these kinds of parsers is finding several roles in any sentences based on predefined frames; these roles are also known as arguments or Args. FrameNet and *PropBank* provide large hand-annotated corpuses for semantic parsers to learn annotating new sentences automatically.

In the process of interpreting a sentence, a verb acts as the primary element for connecting different parts of a sentence together, each part as one role. This methodology breaks the language into frames, which in fact, simulates how human understands a context. Framing varies from very specific to very general. In a domain-specific frame, the focus is on the main verb, for example the frame-set for verb “eat” contains “eater”, ”eaten” and etc. While in a more general structure, frame elements cover a wide range of roles such as list of 9 roles, Fillmore’s list which includes Agent, Experiencer, Instrument, Object, Source, Goal, Location, Time and Path. FrameNet and *PropBank* are two major data-banks for identifying roles based on human annotated texts. [Gildea & Jurafsky, 2002]

FrameNet project offers more than 1000 semantic frames and was originated by Charles J. Fillmore. Frames cover wide variety of subjects along with a textual description. For instance, “Judgment” frame has following roles: Judge, Evaluee, Reason and Statement. Each frame is invoked by a certain set of keywords; in Judgment-frame example blame, admire, praise or fault are some of such keywords. Semantic role classifiers aim to find the best match among these frames and attach an appropriate role to each phrase in a sentence to provide information for further semantic analysis. [Gildea & Jurafsky, 2002]

PropBank stands for proposition bank and is a similar project to FrameNet. It follows the same routine for labeling elements in a sentence but it is not as domain-specific as FrameNet. Instead, roles are named by numbered-arguments: Arg0, Arg1, Arg2, Arg3, Arg4 and ArgM. The first argument, Arg0, is the agent, causer or experiencer. Arg1 represents the patient which is being affected by the experience. Arg2 is instrument; Arg3 and Arg4 indicate starting-point and ending-point respectively. The last argument in *PropBank* ArgM, is a family of all modifiers in a context such as Temporal, Locative, Directional and etc.

The next section explains *Senna*[22], which is employed for shallow semantic parsing in this project. It uses 200Mb RAM and the results are claimed to be 97% accurate. *Senna* designers believe that multilayer neural network has good potential in discovering hidden representations, yet keeps independency of the model from the large linguistic knowledgebase. The results follow *PropBank* labeling style; hence *PropBank* will be discussed with more detail. [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]

Most of the NLP tools are based on statistical models. They rely on input features from experts and then add more features by analyzing syntactic parsing tree; Such as *POS* labels for each word, their positions in relation to the verb, their paths to the verb, the phrasal segment they belong to, the verb tense and etc. This process creates computational overhead for the system for complex texts. [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]

Unlike statistical models, *Senna* creates several layers of features exploited from unlabeled data, using neural network techniques. Neural network is an adaptive system for real-world problem solving; each layer provides a basis for other layers and makes the approach almost independent from any prior knowledge. Because of following a neural network structure, *Senna* does all text analysis from scratch. [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]

The first layer makes feature-vectors for each word in the input-sentence; vectors are then combined and feed further levels of the neural network architecture. *POS* tagging layer manages a special tagging schema, IOBES, for assigning the best syntactic role to each word. This tagging schema indicates boundaries of phrases, or chunks, by 4 simple letter, I, O, B, E and S; I as inside, O as outside, B as beginning, E as ending and S as single. These code-letters attach to a phrase, for instance in a Noun-Phrase, S-NP tag marks a word as start of a Noun-Phrase. [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]

NER in *Senna*, decides whether a chunk represents a specific entity or not. There are four main categories for entities in this layer including locations, person-names, organizations and miscellaneous, which in total covers 8,000 entities. And the most important reason why *Senna* is so fast in semantic role labeling comes from an initial distinction that it has from other semantic role labelers; it doesn't rely on parse trees because its dataset is unlabeled, instead uses a language model that was trained with objective functions to retrieve syntactic information. [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]

[22] Semantic/Syntactic Extraction using a Neural Network Architecture

PropBank annotators mainly focus on phrasal structures from *Penn-TreeBank* to assign proper roles to phrases. For each verb in the tree, *PropBank* brings a set of arguments Arg0-5; these arguments are all gathered as a frame file for that verb. Frame files provide role specifications about each argument by using examples.

Here is a sample frame file taken from [Bonial, Babko-Malaya, Choi, Hwang, & Palmer, 2010] for verb "leave", with the definition of "moving away":

Arg0: entity leaving

Arg1: place left

The above example simply shows the regular labeling for a sentence like "Mary left the room". But "leave" has other implications as well, such as "giving": "Mary left her daughter her pearls". And for this usage another role set is provided in *PropBank*:

Arg0: giver

Arg1: thing given

Arg2: beneficiary

As seen, some verbs like "leave" may infer various meanings; therefore their frame files contain more than one set of arguments, or role-sets, but with different IDs attached to the end of the verb:

Role-set leave.01 'move away'

Role-set leave.02 'give'

Annotators examine instances from the corpus and decide which construction suits the given sentence the best. However, some of the roles may be absent in the target sentence; for instance, "The door opened" does not have Arg0 as the causer or Arg2 as the instrument. [Babko-Malaya, 2005]

In some cases, annotator has to decide between two roles for a word. They often apply ranking strategy of Arg0 > Arg1 > Arg2 >… to make the final decision. [Bonial, Babko-Malaya, Choi, Hwang, & Palmer, 2010]

Senna output is a raw text file and needs preprocessing in order to capture desired information for this project. Here is a sample output for a sentence:

Example: *Senna* input/output

Input: "Workers on a gas production platform in the Bass Strait want their barge returned to port after a major outbreak of salmonella and gastroenteritis."

Output: Table3

Table 3 Senna Sample Output

Token	POS[23] -tag	Chunk -IOBES	NER	Predicate	SRL	SRL
Workers	NNS	S-NP	O	-	B-A0	O
on	IN	S-PP	O	-	I-A0	O
a	DT	B-NP	O	-	I-A0	O
gas	NN	I-NP	O	-	I-A0	O
production	NN	I-NP	O	-	I-A0	O
platform	NN	E-NP	O	-	I-A0	O
in	IN	S-PP	O	-	I-A0	O
the	DT	B-NP	O	-	I-A0	O
Bass	NNP	I-NP	B-LOC	-	I-A0	O
Strait	NNP	E-NP	E-LOC	-	E-A0	O
want	VBP	S-VP	O	want	S-V	O
their	PRP$	B-NP	O	-	B-A1	B-A1
barge	NN	E-NP	O	-	I-A1	E-A1
returned	VBD	S-VP	O	returned	I-A1	S-V
to	TO	S-PP	O	-	I-A1	B-A4
port	NN	S-NP	O	-	I-A1	E-A4
after	IN	S-PP	O	-	I-A1	B-AM-TMP
a	DT	B-NP	O	-	I-A1	I-AM-TMP
major	JJ	I-NP	O	-	I-A1	I-AM-TMP
outbreak	NN	E-NP	O	-	I-A1	I-AM-TMP
of	IN	S-PP	O	-	I-A1	I-AM-TMP
salmonella	NN	B-NP	O	-	I-A1	I-AM-TMP
and	CC	I-NP	O	-	I-A1	I-AM-TMP
gastroenteritis	NNS	E-NP	O	-	E-A1	E-AM-TMP
.	.	O	O	-	O	O

Table3 summarizes the output in rows of information for each token. The first two columns are syntactic information of POS-tags and IOBES boundaries for phrases.

Third column is showing whether the parser has recognized the entity or not, and in here “Bass Strait” is recognized as a location. Fourth column picks the verbs from the input sentence and in their preceding columns there are Semantic roles assigned to each token:

[23] POS tags in Appendix III

1- *Verb: want*

$[Workers\ on\ a\ gas\ production\ platform\ in\ the\ Bass\ Strait\ ]_{Arg0}$
$[their\ barge\ returned\ to\ port\ after\ a\ major\ outbreak\ of\ salmonella\ and\ gastroenteritis]_{Arg1}$

2- *Verb: returned*

$[their\ barge]_{Arg1}$
$[to\ port\ ]_{Arg4}$
$[after\ a\ major\ outbreak\ of\ salmonella\ and\ gastroenteritis]_{ArgM-TMP}$

As shown above, output has a raw text format and for having RDF triples of phrase, there needs to be text processing techniques involved. Triples should maintain the graph shape that means everything must be kept connected.

2.4. Related Work

BIEQA, discussed in [Abulaish & Dey, 2006], is a query answering engine designed for biological questions from MEDLINE abstracts. It triggers a set of journal abstracts to find relevant information by text processing and pattern matching. “Verbs” are the center of analysis in BIEQA and any other information not containing a set of specific verbs is ignored, which can be considered as a negative point in case of losing information. Pattern matching also counts as a weak property for three reasons:

1. It makes the system tied to biological questions with certain patterns
2. Another leakage for losing info that unfit the pre-defined patterns
3. Experts are needed for proposing patterns

Although pattern matching and text processing are fairly generic tasks, this query engine is designed only for MEDLINE abstracts, thus making it inapplicable for information mining from other texts. Moreover, extracted relations are not only coming from grammatical structure of input text, but also from tagged annotations within input; and BIEQA software diagram, Figure9, does not discuss whether it accepts un-annotated texts, or untagged, as well or input is required to be tagged. The last criticism on this article is on text processing step, *POS* tags are mentioned which cover syntactic structure of a text while semantic structure is not brought to attention. [Abulaish & Dey, 2006]

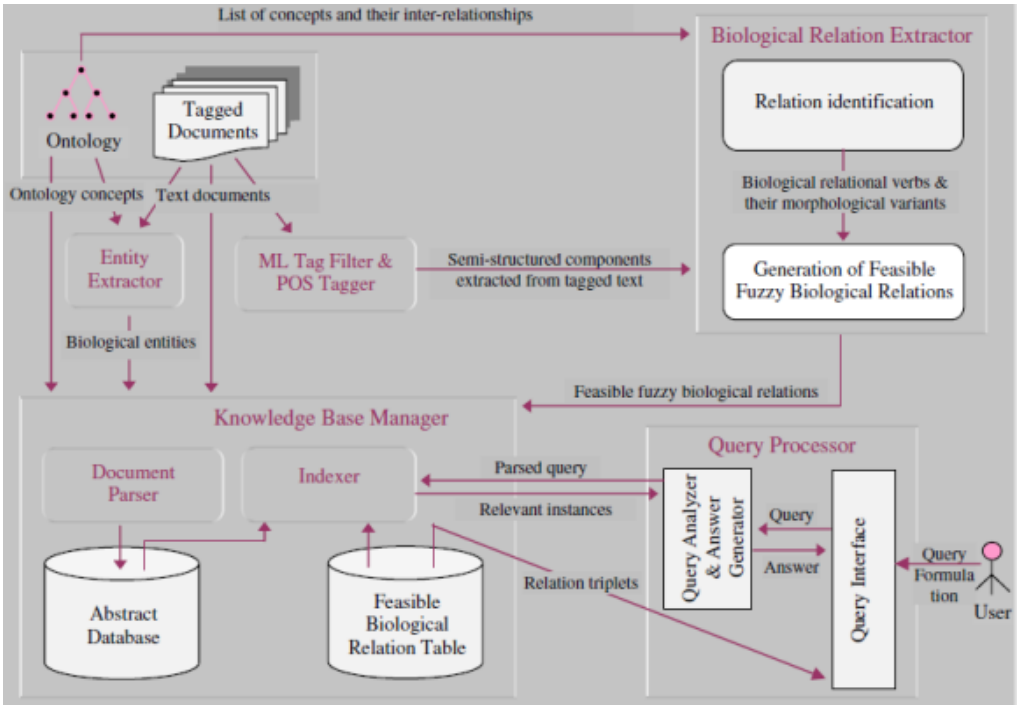

Figure 9 BIEQA Architecture [Abulaish & Dey, 2006]

TextOntoEx [Dahab, Hassan, & Rafea, 2007] is another tool in the area of knowledge extraction from text for ontology learning. Unlike BIEQA, it considers semantic relations, but with a pattern-based strategy. A bundle of patterns are tested on input text to spot specific semantic relations, which ultimately rises the issue of losing information that did not match predefined patterns. In the evaluation section, 100% precision is reported on semantic-relations discovery which hides the fact of ignoring any semantic relations not fitted the built-in patterns.

Text2Onto [Cimiano & Völker, 2005], has a probabilistic approach on constructing ontology. A number of algorithms are stored inside the model of which the best algorithm is chosen by the controller. But before sending the input text to algorithms, a linguistic pre-processing step runs over the text by GATE framework. GATE [Cunningham H. , 2002], is a NLP tool for text analysis and user can define/provide annotation schemas for it; GATE's framework enables user to find data-structures, annotations, phrases and linguistic components out of texts depending on the format of input file. There are several interesting features in GATE to discuss:

First, if input is enriched with markup tags, or XML, GATE highlights every details of it using the embedded schema; user can write his own schema in xml and add it to resources for the GATE's processor. GATE can successfully draw out as many as annotations embedded in an XML file, whereas it is not able to achieve much detail in handling plain texts; tokens, paragraphs, sentences and some recognized entities are the only extracted information mined by GATE from a plain text.

Second, it has a gazetter-list for looking up entities such as cities, organizations, dates and so forth to add more annotations to the output. This feature is mostly similar to NER[24] in other types of NLP parsers. List can be extended depending on user preference.

Third, GATE allows users to detect specific patterns in the text by accepting JAPE rules. As discussed earlier, this method might cause missing information as well as making the model dependent to the experts to design patterns.

Since the text-processing portion of Text2Onto [Cimiano & Völker, 2005] relies on GATE, which is not able to extract any relations among tokens other than those matched the JAPE patterns, may not be considered as a strong tool for information mining from plain texts.

OntoPlus [Novalija, Mladenic, & Bradesko, 2011] extends ontology by adding new concepts driven from texts. The main effort in this tool goes over measuring the relevancy of extracted concepts to an existing ontology by ranking them; Cyc is the knowledgebase to extend which is used during their experiments. Ontologies are assumed to have textual descriptions or comments for their concepts. User gives a list of related keywords and their description to the system; Domain Information Module, DIM, extracts relevant information from the texts based on these keywords provided for the system. DIM performs a preprocessing step on texts such as tokenization, stemming and stop-word removal. It represents texts using bag-of-words as features and TFIDF as the values. The rest of this work is dedicated to calculating the similarities between terms and ontologies; for each term system generates two lists, first is a ranked-list of concepts taken from existing ontologies' contextual-descriptions and second is a set of suggested term-concept relations. User then decides which terms and relations are qualified to add to the ontology.

The text-processing part, discussed in OntoPlus [Novalija, Mladenic, & Bradesko, 2011] , is highly dependent to user inserted-keywords and terms are evaluated from the point of relevancy to the existing ontologies. As a con, there might be loss of information which is not referred by the user. Another point that system is not considering is relations among terms, it only give a list recommended term-concept relations for the user to pick from and does not mine any other/new relationships within texts.

FRED [Presutti, Draicchio, & Gangemi, 2012] summarizes a set of requirements for knowledge extraction from text such as capturing the semantic structure, mapping concepts to appropriate RDF/OWL matches while fulfilling lined-data principals and etc. For semantic structure, it uses a deep parser called Boxer. Boxer's approach for

[24] Named Entity Recognition

discovering complicated relations is based on DRT[25]; it uses VerbNet to collect roles that may be involved in a sentence and chooses a relevant frame from FarmeNet for that sentence. FRED claims that it has improved Boxer accuracy in detecting frames by providing a complete mapping between VerbNet and FrameNet. Linguistic frames, generated by Boxer, are then translated into RDF representations following a bunch of heuristic rules. These rules are direct translations of Boxer syntax, for instance rel_name(x,y) from Boxer is turned into owl:objectProperty, so it really depends on how Boxer performs on input text. Heuristic rules apply to different situations. For variables, or new terms, detected by Boxer, it creates a class; if that term is a part of a phrase, a bigger combination is considered as a super class, here is an example:

Paul Newman hit the window with an open hand.

"Hand" is a Class, hand is an instance of it. "OpenHand" is a superclass and "Hand" is a subclass of it.

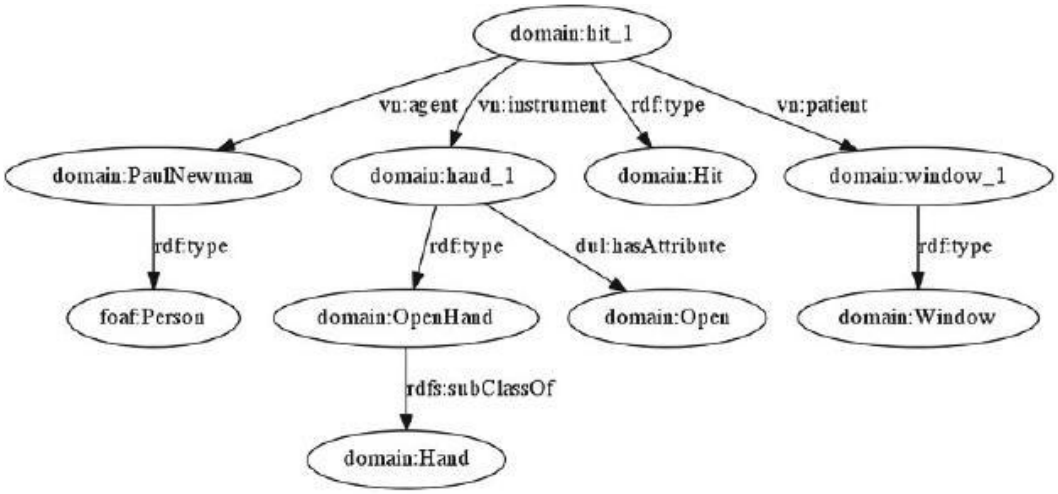

Figure 10 FRED graph for a sentence [Presutti, Draicchio, & Gangemi, 2012]

Graph shows reasonable design but as mentioned above the results depend on the performance of parser. Additionally, as author discussed deep parsers, Boxer is only being compared to SEMAFOR, whereas there are other parsers such as *Senna* which is not talked about. And there are no methods of evaluation for the tool.

[25] Discourse Representation Theory

3. Generating RDF from Text

3.1. Overview

In a nutshell, conversion of a plain text into a data graph of RDF triples is performed based on processing of outputs of two parsers, syntactic parser "*Stanford*" and semantic parser "*Senna*". In this chapter, we explain the algorithm, tools and some preliminary results.

3.2. Used Tools

- Syntactic parser[26]:

    - Software name: The *Stanford* Parser, A statistical parser [Marneffe, MacCartney, & Manning, 2006]
    - Version: 2.0.3 (released 2012/07/09)
    - Requirement: Java 5 (JDK 1.5.0+). It must be installed separately[27].
    - Multilingual: English, Chinese, French, German
    - Supporting Platforms: Linux,Windows, MacOs
    - Command line: *./lexparser.sh inputfile.txt > output.txt*
    - Max-heapsize: it is 150m by default, but user can increase the value in the shell file
    - Authors: Marie-Catherine de Marneffe, Bill MacCartney and Christopher D.

- Semantic parser[28]:

    - Software name: *Senna [Collobert, Weston, Bottou, Karlen, Kavukcuoglu, & Kuksa, 2011]*
    - Version: 3.0 (released August 2011)
    - Requirement: It requires about 200MB of RAM and should run on any IEEE floating point computer
    - Supporting Platforms: Linux, Windows, MacOS
    - Command line: ./*Senna* <inputfile.txt> output.txt
    - *Senna* is written in ANSI C, with about 3500 lines of code. It requires at least 200MB of RAM and should run on any IEEE floating point computer.

[26] Please refer to section 2.3 Tools for more info
[27] "java –version" command shows the current java installed on platform
[28] Please refer to section 2.3. Tools for more info

- Authors: R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. Kuksa

- Programming Language:

  - Name: *Python*
  - Version: 2.7.3 (released April9, 2012)
  - Supporting Platforms: Linux, Windows, MacOS
  - Libraries:
    - NLTK: Natural Language Toolkit provides variety of text-processing libraries for working with human language data. This toolkit needs to be installed individually and it does not come with *Python* by default. NLTK has a package called corpus reader that covers different corpora formats such as plain-text, tagged, *PropBank* and other corpora. Based on the library imported, user is able to find out more info about the term frequencies, length of the document, words, sorting the words, tags and etc.

      PropBank is a library imported from NLTK in this project; it is employed to find role-sets assigned to specific verbs, used in input, for tagging terms with proper role-names. Role-sets are defined in separate files, or Frameset files, and may contain more than one set; each file delivers description of argument roles with examples.

    - En: it is another library used in this project and needs individual installation. We use this library to convert the verbs into their simple present tense format.

- Visualization:

  - Software Name: *Gephi*
  - Version: 0.8.1 beta (released Feb 2011)
  - Platforms: Linux, Windows, MacOS

3.3 Process Description

The developed procedure for generating RDF triples is broken into four rounds: pre-processing, sentence simplification, triple transformation and additional triples; additionally, the final results are translated in DOT format for graph visualization.

Input: input is a plain text which does not contain any annotations or tags. Our developed system can accepts one sentence at a time or can be called multiple times for

processing a paragraph. Most of the test cases in this section are from ProMED reports, a set of medical reports of water virus detection.

Goal: the goal is to mine all relations existing among terms in a sentence from the input context. Relations are categorized into two groups of syntactic and semantic relations. The first group comes from *Stanford* parser, and the second group from *Senna* parser; both of these parsers are discussed in detail in the rest of the section. Major part of this developed system contains text processing using *Python* programming language.

3.3.1 Pre-processing: Round 1

The very first step in dealing with a text is performing a pre-processing all over the content. For instance, some symbols or characters are obstacles for parsers to fully process a sentence. This step includes various modifications on the text such as:

A. Removing a number of symbols such as comma, hyphen, slashes and other noisy characters from input sentence. This process simply finds these symbols and replaces them with null character.

B. Finding dates in the sentence: These two parsers are prune to make mistakes when there is information about dates in a sentence. Here is an example for illustrating how *Senna* parses a date in a wrong way, and how it should have captured it:

Example: Dates/temporal problem in *Senna* Output

Input: “At last report on 11 May 2012, 15 human cases of Samonella Infantis infections were confirmed in the USA .

Original Output: Table 4

Correct Output: Table 5

Table 4 Senna output, Dates/temporal problem

Term	POS	Chunk	NER	Predicate	SRL
at	IN	S-PP	O	-	O
last	JJ	B-NP	O	-	O
report	NN	E-NP	O	-	O
on	IN	S-PP	O	-	O
11	CD	B-NP	O	-	O
May	NNP	I-NP	O	-	O
2012	CD	I-NP	O	-	O
,	,	I-NP	O	-	O
15	CD	I-NP	O	-	B-A1
human	JJ	I-NP	O	-	I-A1
cases	NNS	E-NP	O	-	I-A1
of	IN	S-VP	O	-	I-A1
Salmonella	NNP	B-NP	S-MISC	-	I-A1
Infantis	NNP	I-NP	O	-	I-A1
infections	NNS	E-NP	O	-	E-A1
were	VBD	B-VP	O	-	O
confirmed	VBN	E-VP	O	confirmed	S-V
in	IN	S-PP	O	-	B-AM-LOC
the	DT	B-NP	O	-	I-AM-LOC
USA	NNP	E-NP	S-LOC	-	E-AM-LOC
.	.	O	O	-	O

There are two columns in the above table that require explanations, Chunk and SRL[29]:

1. Chunk defines margins of a phrase containing beginning, inside and ending terms of it; “11” is the beginning of a noun phrase, “B-NP”, but the ending is assigned to “cases”, E-NP. Whereas this chunk boundary should only cover “11 May 2012”. Such problems arise because of informal grammar that is commonly used in natural language writings. Additionally, this issue might cause another problem shown in SRL column.
2. SRL column has no indication of roles being assigned to “at last report” or “on 11 May 2012” by parser; whereas “15 human cases of Salmonella Infantic Infections” is A1 or thing-confirmed and “in the USA” is recognized as AM-LOC or Location.

[29] Semantic Role Label

Table 5 illustrates proper analysis of the sentence; Beginning and ending terms of “11 May 2012” are fixed and the SRL shows AM-TMP. This is how the *Senna* output should look like in order to provide correct information for text mining tool.

Table 5 Date/Temporal problem; Applied solution

Term	POS	Chunk	NER	Predicate	SRL
at	IN	S-PP	O	-	O
last	JJ	B-NP	O	-	O
report	NN	E-NP	O	-	O
on	IN	S-PP	O	-	O
11	CD	**B-NP**	O	-	**B-AM-TMP**
May	NNP	**I-NP**	O	-	**I-AM-TMP**
2012	CD	**E-NP**	O	-	**E-AM-TMP**
,	,	,	O	-	O
15	CD	B-NP	O	-	B-A1
human	JJ	I-NP	O	-	I-A1
cases	NNS	E-NP	O	-	I-A1
of	IN	S-VP	O	-	I-A1
Salmonella	NNP	B-NP	S-MISC	-	I-A1
Infantis	NNP	I-NP	O	-	I-A1
infections	NNS	E-NP	O	-	E-A1
were	VBD	B-VP	O	-	O
confirmed	VBN	E-VP	O	confirmed	S-V
in	IN	S-PP	O	-	B-AM-LOC
the	DT	B-NP	O	-	I-AM-LOC
USA	NNP	E-NP	S-LOC	-	E-AM-LOC
.	.	O	O	-	O

Although *Senna* Parser does not always parse dates with mistakes; In this project we try to improve the quality of final results by pre-identifying all dates in input. Hence we define specific type of dates or date-pattern in the following format:

(\d{1,2}\s\w+\s\d{1,4}) is a pattern that finds any portion of text with (2 digits, a space, a word, space and 4 digits); like 2 May 2009.

This frame collects all parts of text fitting the pattern; Then we put them all into a list. At the same time we add a triple < found-date ; is ; date > to the final list of triples. This list helps fixing triples parsed from *Stanford* Parser in further dependency translations.

Example: Date triples

Input: "At last report on 11 May 2012, 15 human cases of Samonella Infantis infections were confirmed in the USA."

Extracted triples: Two important triples extracted from the example sentence are shown below. Triples are the output of combining Sanford and *Senna* parsers.

- < report ; on ; 11/05/2012 >

- < 11/05/2012 ; is ; date >

Note: Months may have different typed formats like January or Jan, these styles are all generalized as a numbered date; so Jan or January is modified to 1, Feb or February to 2 and so forth.

C. Capital names; Parsers cannot always handle capital names in a sequence and most of the time these names are considered as two individual names while they might be the name of a company or a country. The following example demonstrates how *Stanford* and *Senna* consider the structure.

Example[30]**:** Capital Names in *Senna* and *Stanford*

Input: "Yet the Missouri based maker of Diamond, Premium Edge, Kirkland Signature, and other pet food brands has not called special attention to the expansion of the recall to cat food beyond amending a statement on the company Internet recall site.

Output: Figure 9 *Stanford* Output, Table 6 *Senna* Output

Modified Output: Figure 10

[30] Complete output in Appendix V

```
partmod(Missouri-3, based-4)
dobj(based-4, maker-5)
prep_of(maker-5, Diamond-7)
nn(Edge-10, Premium-9)
prep_of(maker-5, Edge-10)
conj_and(Diamond-7, Edge-10)
nn(Signature-13, Kirkland-12)
prep_of(maker-5, Signature-13)
conj_and(Diamond-7, Signature-13)
nn(site-43, company-40)
nn(site-43, Internet-41)
nn(site-43, recall-42)
```

Figure 11 Capital Names; *Stanford* Output

In this set of results from *Stanford*, Premium Edge, Kirkland Signature and company internet recall are all parsed alike as a noun compound, or nn, instead of a single noun.

Table 6 Capital Names; Senna Output

Term	POS	Chunk	NER	Predicate	SRL (called)	SRL (amending)
Yet	CC	O	O	-	O	O
the	DT	B-NP	O	-	O	O
Missouri	NNP	I-NP	S-LOC	-	O	O
based	VBN	I-NP	O	-	O	O
maker	NN	E-NP	O	-	O	O
of	IN	S-PP	O	-	O	O
Diamond	NNP	B-NP	S-LOC	-	O	O
,	,	I-NP	O	-	O	O
Premium	NNP	I-NP	O	-	O	O
Edge	NNP	E-NP	O	-	O	O
,	,	O	O	-	O	O
Kirkland	NNP	B-NP	S-ORG	-	O	O
Signature	NNP	E-NP	O	-	O	O

NER column shows that *Senna* is not able to fully detect “Premium Edge” and “Kirkland Signature” as two organization names or ORG.

The main problem with this analysis is that a noun compound does not deal with its terms equally and there is always one head-term while other terms are dependents. As it breaks down a name into terms, it is actually capturing one part of that as a head term; which means only a part of that name instead of all parts it. So “maker of Premium Edge” is captured as “maker of Edge” in this dependency prep_of(maker,Edge) instead of prep_of(maker, Premium Edge).

Applied solution for achieving the best results on detecting Capital Names is finding sequence of words starting with capital letters with regular expression patterns. Parts of the text matched the pattern is merged into one single term by removing spaces in the middle.

```
prep_of(maker-5, PremiumEdge-10)
conj_and(Diamond-7, PremiumEdge-10)
prep_of(maker-5, KirklandSignature-13)
conj_and(Diamond-7, KirklandSignature-13)
```

Figure 12 Capital Names; Modified Output

We deliver the pre-processed text to *Senna* and *Stanford* parsers. *Stanford*'s output[31] contains all grammatical dependencies in a raw text format. All these dependencies are put in our dependency list, Dep-List. *Senna* output[32], on the other hand, has numerous columns of information for each term, all terms with VB tag such as VBP, VBN or VBZ in the first-column[33] of output, are gathered in our second list called Verb-List. Both lists are used in Round 2, where we split a sentence with multiple verbs into several sentences each with single verb.

[31] For seeing a sample Stanford output please refer to 2.3.1
[32] For seeing a sample Senna output please refer to 2.3.2
[33] POS tags, Appendix III

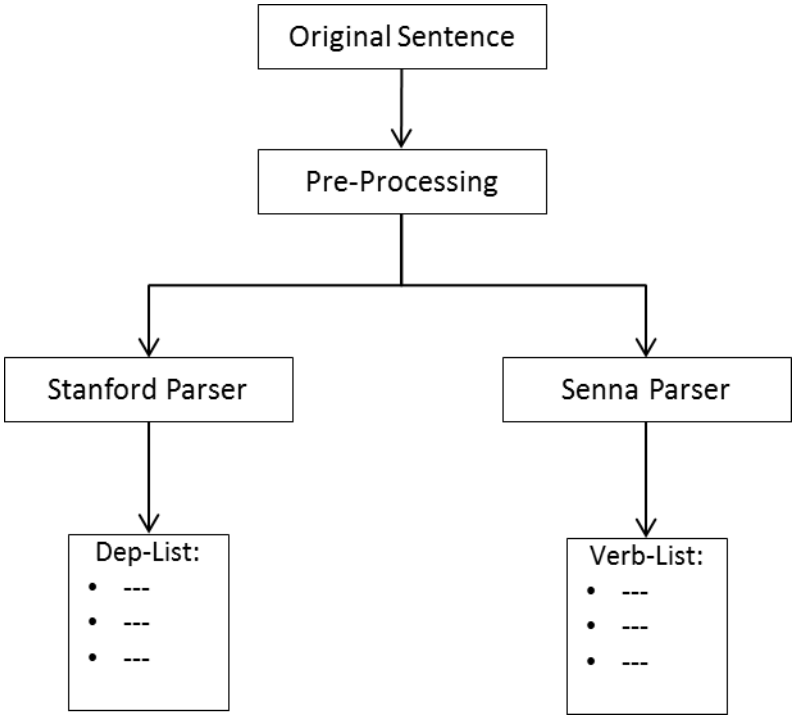

Figure 13 Pre-processing step, Senna and Stanford output lists

3.3.2 Sentence Simplification: Round 2

In this phase our main goal is splitting the input sentence which may contain more than one verb into sentence(s) with one verb each. The idea is to simplify long sentences so *Senna* is able to discover more roles. Instead of parsing single complex sentence, in which verbs connect multiple shorter sentences together, a number of simplw sentences are sent to *Senna* and *Stanford* separately.

Example:

Original Sentence: "in Singapore, there is close cooperation between medical and veterinary services which leads to good surveillance of zoonotic diseases."

Sentence 1: in Singapore, there is close cooperation between medical and veterinary services.

Sentence 2: close cooperation between medical and veterinary services leads to good surveillance of zoonotic diseases.

Moreover, we here observed that in some cases, breaking the sentence into simpler ones has improved *Senna*'s accuracy in finding more roles in the sentence:

Example:

Input: "The Missouri, maker of Diamond, Premium Edge, Kirkland Signature, and other pet food brands has not called special attention to the expansion of recall for cat food beyond amending a statement on company Internet recall site."

***Senna* output:**

-Roles identified by parsing the original sentence

Verb: called

[The Missouri, maker of Diamond, Premium Edge, Kirkland Signature, and other pet food brands]$_{\mathrm{Arg0-caller}}$

[special attention]$_{\mathrm{Arg1-item\ being\ called}}$

Verb: amending

[a statement on company Internet recall site]$_{\mathrm{Arg1-item\ being\ amended}}$

-Roles identified by parsing split sentences

Verb: called

[The Missouri, maker of Diamond, Premium Edge, Kirkland Signature, and other pet food brands]$_{\mathrm{Arg0-caller}}$

[special attention]$_{\mathrm{Arg1-item\ being\ called}}$

[the expansion of recall for cat food]$_{\mathrm{Arg2-attribute\ of\ Arg1}}$

Verb: amending

[a statement]$_{\mathrm{Arg1-item\ being\ amended}}$

[company Internet recall site]$_{\mathrm{Arg2-secondary\ prediction}}$

As it is shown in split sentences "Arg2-attribute of Arg1" and "Arg2-secondary prediction" are detected in addition to previous roles from original sentence.

The process of splitting sentence into smaller sentences is explained below:

Input: Dep-List, Verb-List, Original Sentence

Goal: Finding all sequences of terms connected to each verb

1. Fetching a verb from Verb-List
2. Scanning Dep-List for any terms connected to the verb from the 1st step; adding these terms to a dynamic list called Token-List

   Note: Token-List does not contain any verbs
3. For any term in Token-List rescan the Dep-List to find all connections, new terms are added to Token-List

   Note: if a verb is encountered in the process of finding a sequence of terms, that term has reached its stopping point; while other terms may still continue to find remaining connections.
4. The process continues until Token-List no longer extends.
5. New sentence is created by scanning terms from Token-List and putting them in the order they have in Original sentence.
6. The whole process repeats for all verbs in Verb-List

This sequence of actions assures that for all verbs, there is a list of related terms in a form of a sentence. Each sentence is transferred to *Senna* and *Stanford* for further information mining.

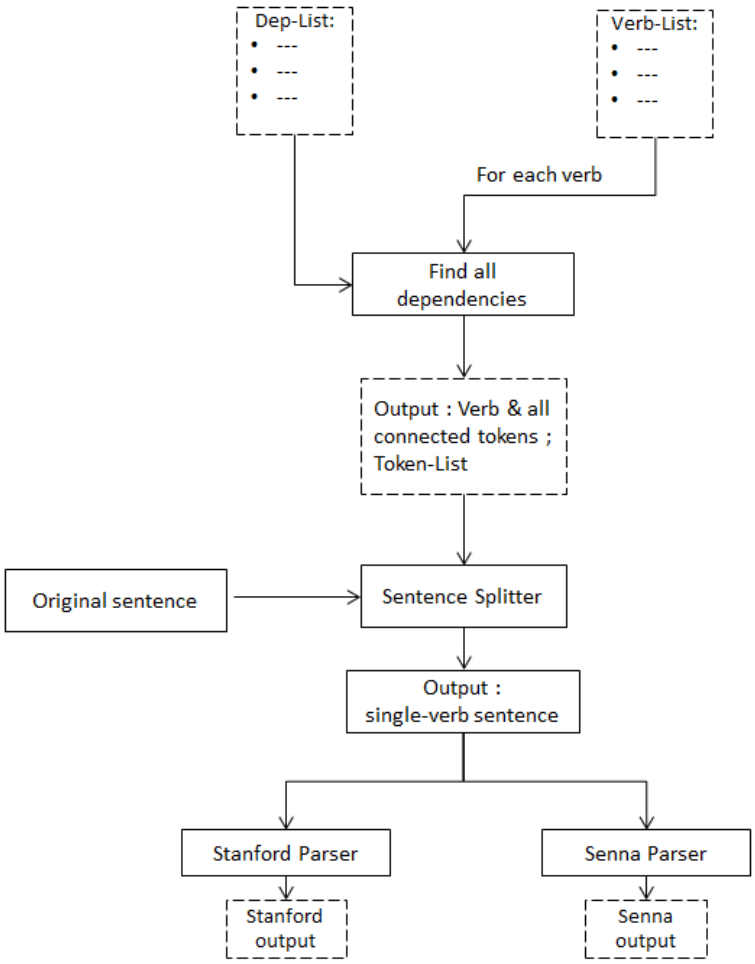

Figure 14 Sentence Simplification

Note: Sentences are moving out of the coverage of other influencing verbs, it means that a sentence, or a part of it, might have had a role of an agent or a patient for another verb. These roles are retrieved in Round4 for the sake of keeping all information. The results of *Senna* and *Stanford* are input for Round 3.

3.3.3 Statement Translator: Round 3

Statement-Translator is the core of the developed process; it receives *Senna* and *Stanford* outputs for each sentence, created in Round2, and applies various translation steps on them. Process is discussed in Translating *Senna* output and Translating *Stanford* output:

- **Translating *Senna* output**

There are 2 categories to consider, Roles and NER.

- Roles: *Senna* is a semantic parser; it assigns roles[34] to term(s) of each sentence based on the verb of that sentence. A few issues are discussed below:

Issue1, *Senna* only provides an abstract role such as A0, A1, ..., and A5 without any further consideration of the proper name for these roles. Here is a brief example of this issue:

Example: Args Vs. Roles

Input: "Close cooperation between medical and veterinary services leads to good surveillance of zoonotic diseases."

***Senna* output**[35]: cooperation→ A0, surveillance → A1

Proper output: cooperation → factor, surveillance →result

As demonstrated in the example, *Senna* doesn't provide the detailed-name of the role, A0 is the agent/experience and A1 is the patient/undergoes the experience. We resolve this problem in the following way:

1) Taking the verb of sentence

2) Changing the verb to its present-tense format; using "en" library in *Python*

3) Finding the first role-set of that verb in *PropBank* from NLTK library in *Python*

4) Replacing A0-5 with their real role names from the role-set

Issue2, *Senna* assigns a role to a set of terms or a phrase; it causes complexity in data graph shown in Figure 13. To reduce the complexity and redundancy of final data-graph we limit roles are only to the terms which are directly connected to the verb of the sentence.

[34] For more information about roles please refer to 2.3.2
[35] For full output please refer to Appendix V

Example: Simplifying Graph

Input: "36 of more than 200 workers have fallen ill in 2 weeks since the outbreak, union said."

data-graph (not simplified): Figure 13 ; **data-graph (simplified):** Figure 14

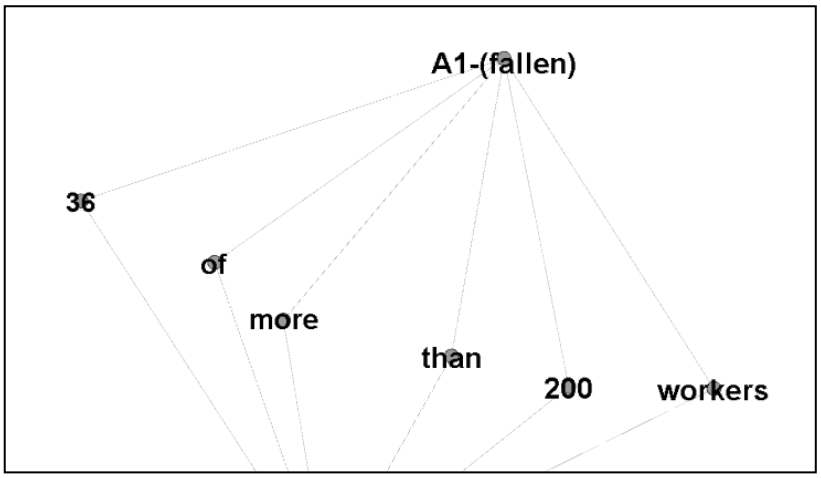

Figure 15 Data-Graph, a part of not-simplified version

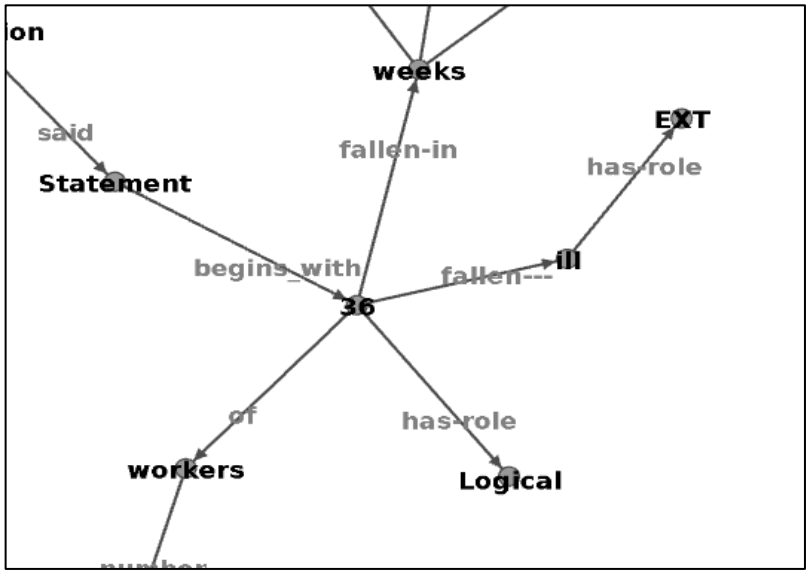

Figure 16 Data-Graph, a part of simplified version

In Figure 13, only a part of data-graph is depicted to show some unnecessary edges between terms and A1 role, in the simplified version we omit many of these edges and separates essential terms from extras.

Figure 14 shows the simplified data-graph. In this version, noun phrase "36 of more than 200 workers", "36" is assigned as A1 and the other terms are linking to it such as "36-of-workers". This approach is based on the fact that any phrasal structure in English language grammar has one head and a group of words, which are the other words in that phrase, as its dependents. [Lockwood, 2003]

Considering these dependents as extra information about the head-term, reduces the edges of the data-graph. *Stanford* output is influential in this process as it marks grammatical dependencies; the process is explained in three easy steps below:

1. Find all terms directly connected to the verb from *Stanford* output
2. Assign role-names discovered from *Senna* output, only to this list of terms; < head-term ; is ; role-name >
3. The rest of *Stanford Dependencies* are remained to be translated during "Translating *Stanford* output" phase

- NER: another category of relations are terms that are recognized as a name of an entity like a location, a person and etc. Such information is provided in the 3rd column of *Senna* output[36]. If a term does not have any specific NER the space is left empty.

Example: Location

Input: "laboratory testing that conducted by state public health laboratories in Connecticut, Maryland, Pennsylvania and Wisconsin"

***Senna* output**[37]: Connecticut → S-LOC, Maryland → S-LOC, Pennsylvania → S-LOC, Wisconsin → S-LOC

Triples: (Connecticut, is, Location), (Maryland, is, Location), (Pennsylvania, is, Location), (Wisconsin, is, Location)

Note: the "S" in S-LOC means single term or indicates that it is a single term not a phrase.

[36] For more information about senna output presentation please refer to 2.3.2.
[37] For complete output please refer to Appendix V

- **Translating *Stanford* output:**

Stanford parser looks for grammatical structure of the text in a verb-oriented basis. Dependencies discovered by *Stanford* do not have equal amount of importance; some of them may be considered as extra information while others play a major part in conveying the essence of the sentence such as subject and objects. As *Stanford* Parser generates term-to-term relations, additional effort is needed to translate pairs connected to verbs. Details are provided into two levels, **Essentials** and **Extras**:

- **Essentials**: these types of dependencies are important because of showing the backbone of the sentence. The goal is to translate them into < subject ; verb ; object > triple format. Having considered that dependencies come in a pair-wise order, any pair that contains verb must be matched with another pair containing that verb in a proper order.

Example: "in Singapore, there is close cooperation between medical and veterinary services which leads to good surveillance of zoonotic diseases."

***Stanford Dependencies*:** Figure 15

Goal: finding the essential dependencies

< cooperation ; leads-to ; surveillance >

< cooperation ; is-in ; Singapore >

```
prep_in(is-5, Singapore-2)
expl(is-5, there-4)
root(ROOT-0, is-5)
amod(cooperation-7, close-6)
nsubj(is-5, cooperation-7)
nsubj(leads-14, cooperation-7)
amod(services-12, medical-9)
conj_and(medical-9, veterinary-11)
amod(services-12, veterinary-11)
prep_between(cooperation-7, services-12)
rcmod(cooperation-7, leads-14)
amod(surveillance-17, good-16)
prep_to(leads-14, surveillance-17)
amod(diseases-20, zoonotic-19)
prep_of(surveillance-17, diseases-20)
```

Figure 17 Stanford Translation; Stanford Dependencies

For generating such outputs we create two lists Subject-List and Object-List; Subject-List contains all the terms appeared on the left side of the verb and they have a dependency with the verb in *Stanford* output and the same for Object-List for the terms on the right side of the verb. Since there is only one verb in input sentence, for any item in Subject-List there is a relation to all items in Object-List with that verb. Sometimes a verb comes with a preposition, so conditional modifications may apply to relations. The process is summarized in Figure 16.

- Subject-List: There are certain names for dependencies that mark subject-role of a term for a verb such as nsubj, csubj, nsubjpass; once a pair is detected with these names the term is added to Subject-List.
- Object-List: any other pair directly connected to the verb of the sentence is considered as an object. All items in this list have a connection with all subjects in Subject-List.

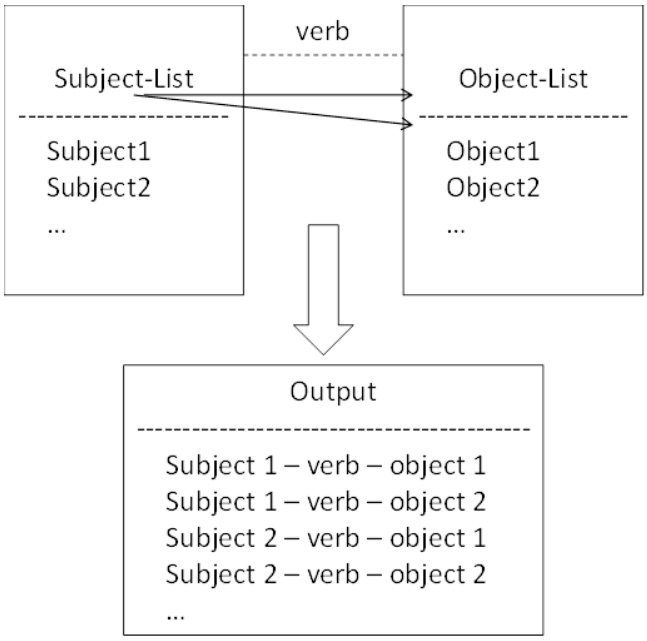

Figure 18 Summary of Subject-Object relations

Here are some triples we created from *Stanford* output:

Example: Translating prepositions

Input: "in Singapore, there is close cooperation between medical and veterinary services which leads to good surveillance of zoonotic diseases."

Subject-list: [cooperation]

Object-List: [prep_to(leads-14, surveillance-17), prep_in(is-5, Singapore-2)]

Combination: <cooperation ; leads-to ; surveillance >,

<cooperation ; is-in ; Singapore >

- **Extras:** As mentioned above, some of the terms in a sentence act as extra information about other terms such as adjective, adverbs, modifiers, numbers, abbreviations and etc.

In order to have a consistent translation for *Stanford Dependencies*, we define a dictionary for the dependency names and their equivalent relation:

Table 7 Dictionary for translating Stanford Dependencies

Dependency	Definition	Equivalent
abbrev	Abbreviation	sameAs
acomp	Adjectival complement, it comes in an adjectival phrase and describes a verb. Ex: He looks happy. → acomp(looks, happy)	Is
advmod	Adverbial modifier, it is used for modifying the meaning of a word. Ex: relatively slow → advmod(slow,relatively)	moreDetail
agent	It complements a passive verb. Ex: results achieved by students were significant. → agent(achieved, students)	by
appos	Appositional Modifier, it is a noun phrase which serves as a modifier for another noun phrase. Ex: He was John, her boss → appos(John, boss)	sameAs
attr	Attributive; it is a verb complement. Ex: What is that? → attr(is, what)	--
csubjpass	Clausal Passive Subject; it serves as a clausal subject for a passive clause. Ex: that he left was noticed by everyone. → csubjpass(noticed, left)	moreDetail
dobj	Direct Object; it is a noun phrase that comes as the object of a verb. Ex: He gave her all the old books. → dobj(gave, books)	object
iobj	Indirect Object; this object appears in sentences with direct object. Ex: He gave her all the books. →iobj(gave, her)	to
neg	Negation Modifier Ex: She did not like the taste. → neg(like, not)	not
nsubj	Nominal Subject; a noun phrase as subject of the sentence. Ex: that house was expensive. → nsubj(expensive,house)	subject
csubj	Clausal subject; it is the subject of the sentence in form of a clause. Ex: what she said makes sense. →csubj(makes,said)	subject
nsubjpass	Passive Nominal Subject. Ex: Results were achieved by students. → nsubjpass(achieved, results)	subject
num	Numeric Modifier.	number

	Ex: 3 men were killed in the accident. → num(men,3)	
number	Element of compound number; it represents currency amount. Ex: I lost $ 3 million. Number($,million)	currency
partmod	Participial Modifier; it modifies a noun phrase. Ex: Truffles picked during the spring are tasty. → partmod(Truffles, picked)	moreDetail
poss	Possession Modifier. Ex: my notes → poss(notes, my)	possession
prep	Prepositional Modifier; Ex: money is in the wallet. → prep(is,in)	On, in, at,…
quantmode	Quantifier Modifier. Ex: about 200 people. → quantmod(200,about)	quantity
tmod	Temporal Modifier; it modifies a phrase by mentioning the time. Ex: Last night, he went to the airport. → tmod(went,night)	time

"nn" and "amod" are two dependencies which are not covered in Table 7. "nn" as noun compound modifier, and "amod" as adjective modifier normally appears in noun-phrases. When a term is marked as "nn" or "amod", it means that term comes as a modifier to head of noun-phrase [Marneffe & Manning, 2008].[38]

As dependency pairs of "nn" or "amod"-type show that a term belongs to a bigger noun-phrase, they are gathered in a group of terms to serve as extra information, or parent of head-noun, which forms a hierarchical representation for that noun-phrase. The same example that was studied in Essential dependencies is used in here:

Example: Translating Noun-Compounds and Adjective Modifier

Input: "in Singapore, there is close cooperation between medical and veterinary services which leads to good surveillance of zoonotic diseases."

Stanford Dependencies: Considering only "nn" and "amod" of Extra dependencies discussed above

amod(cooperation , close)

amod(services , medical)

amod(services , veterinary)

[38] Other dependencies which are covered in translation do not provide useful information such as attr/verb-complements (like to-be, to-seem) , aux/auxiliary of a clause (like has died, should leave) and etc.

amod(surveillance , good)

amod(diseases-20, zoonotic-19)

RDF Triples:

(services - superClassOf- medical veterinary services)

(diseases - superClassOf- zoonotic diseases)

Here is the summary of whole translating process for *Senna* and *Stanford* outputs. There is more information to add to the final data analyzed in Round 4.

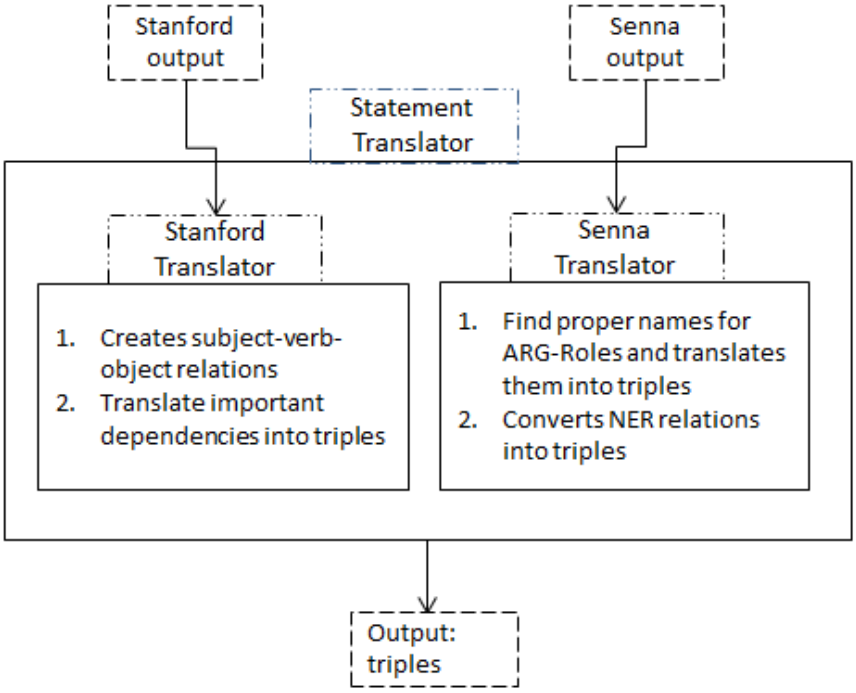

Figure 19 Statement Translator

3.3.4 Final Processing; Round 4

In Round 2, we split the input sentence with multiple verbs to smaller sentences with single verb in each. Although it helps parsers, specifically *Senna*, to achieve better role assignments, it may cause loss of information due to splitting that happens between sentences. Example below demonstrates how some roles are missing as sentences are disconnecting from each other:

Example: Missing Roles

Input: "36 of more than 200 workers have fallen ill in 2 weeks since the outbreak, their union said."

***Senna* Output** (Original Sentence):

Verb: fallen

[36 of more than 200 workers]$_{\text{Arg1}-\text{thing undergoing change}}$

[ill]$_{\text{Arg2}-\text{end state}}$

[2 weeks since the outbreak]$_{\text{Arg}-\text{TMP}-\text{time}}$

Verb: said

[36 of more than 200 workers have fallen ill in 2 weeks since the outbreak]$_{\text{Arg1}-\text{utterance}}$

[their union]$_{\text{Arg0}-\text{sayer}}$

***Senna* Output** (Processed Sentences):

Verb: fallen

[36 of more than 200 workers]$_{\text{Arg1}-\text{thing undergoing change}}$

[ill]$_{\text{Arg2}-\text{end state}}$

[2 weeks since the outbreak]$_{\text{Arg}-\text{TMP}-\text{time}}$

Verb: said

[their union]$_{\text{Arg0}-\text{sayer}}$

Two groups of roles from the example above, shows that "Arg1-utterance" from verb "said" is missing. Recovering such lost roles needs retrieving roles for each term in the original sentence before any modifications. In this regard, for each verb of Verb-List all terms being affected are grouped along with their role.

Arg-List is a list of verbs; Each verb has its own sub-list of terms/roles from the original sentence. Example:

Arg-List:[fallen:{Arg1:{36, of ,more, than, 200 ,workers, } ,Arg2:{ill}, Arg-TMP:{ 2, weeks, since, the, outbreak } } ,

said:{Arg0:{Their, union} ,Arg1:{ 36, of, more, than, 200, workers, have, fallen, ill, in, 2, weeks, since, the ,outbreak }} **]**

There is another list called DCT-List or Directly Connected Terms which keeps all terms with direct link to verbs, taken from *Stanford* output Dep-List. For making this list, we scan all dependencies from *Stanford* output and take all of those with verbs. Example:

Stanford Output:

nsubj(fallen-8, 36-1)

mwe(than-4, more-3)

quantmod(200-5, than-4)

num(workers-6, 200-5)

prep_of(36-1, workers-6)

aux(fallen-8, have-7)

ccomp(said-19, fallen-8)

advmod(fallen-8, ill-9)

num(weeks-12, 2-11)

prep_in(fallen-8, weeks-12)

det(outbreak-15, the-14)

prep_since(weeks-12, outbreak-15)

poss(union-18, their-17)

nsubj(said-19, union-18)

root(ROOT-0, said-19)

DCT-List: **[** fallen:{36, have, ill, weeks} said:{union, fallen}**]**

In this module, we take the single-verb sentence and do a checking process for any missing roles assigned by other verbs:

1- A verb is taken from Verb-List, Example : “said”

2- Two lists of roles are retrieved, one contains all roles from split sentence, and the other one has all roles from original sentence or Arg-List. Example:

- ***Split sentence****:* “36 of more than 200 workers have fallen ill in 2 weeks since the outbreak.”
- ***Roles from split sentence***:

  { Arg1-(fallen) {36, of, more, than, 200, workers} ,

  Arg2- (fallen) {ill},

  Arg3- (fallen) {2, weeks, since, the outbreak} }
- ***Roles from Arg-List***:

  { Arg1-(fallen) {36, of, more, than, 200, workers},

  Arg2- (fallen) {ill},

  Arg3- (fallen) {2, weeks, since, the outbreak},

  Arg1-(said) {36, of, more, than, 200, workers, have, fallen, ill, in, 2, weeks, since, the, outbreak},

  Arg0-(said) {their, union} }

3- We gather all terms of split sentence, or single-verb sentence. For each term, we check whether it is in Arg-List or not. Example:

- ***All terms from split Sentence:***

  {36, of, more, than, 200, workers, have, fallen, ill, in, 2, weeks, since, the , outbreak}
- ***Checking each term:*** the first term is ***“***36”
- “36” is in Arg-List :
  1. Arg1-(fallen) {36, …} → <u>We already have this one in</u> “**Roles from split sentence**”
  2. Arg1-(said) {36, …} → <u>Missing role</u>

4- If that term is in Arg-List, we check DCT-List to see if this term is directly connected to verb or not.

- ***DCT-List***: **[** fallen:{36, have, ill, weeks} said:{union, fallen}**]**
- "36" is in DCT-List

5- If Term is in DCT-List as well, we keep its role from Arg-List , and add this triple(term, is , this-role) to our output.

- ("36"- is - Arg1(said))

  Interpretation: "36" is the head word of "36 of more than 200 workers", and it is directly connected to verb "fallen". These two properties make it as a good choice for creating a new triple.

Note: DCT-List helps reducing the complexity of data-graph by limiting the roles to those terms which are directly connected to the verbs. As addressed earlier, for each phrase there is one head term that makes other terms act as extra information about it. So the role is only effective for the head term.

Example: "36 of more than 200 workers have fallen ill in 2 weeks since the outbreak, their union said."

Additional Triples:

(36 of more than 200 workers have fallen ill in 2 weeks since the outbreak - is - Utterance)

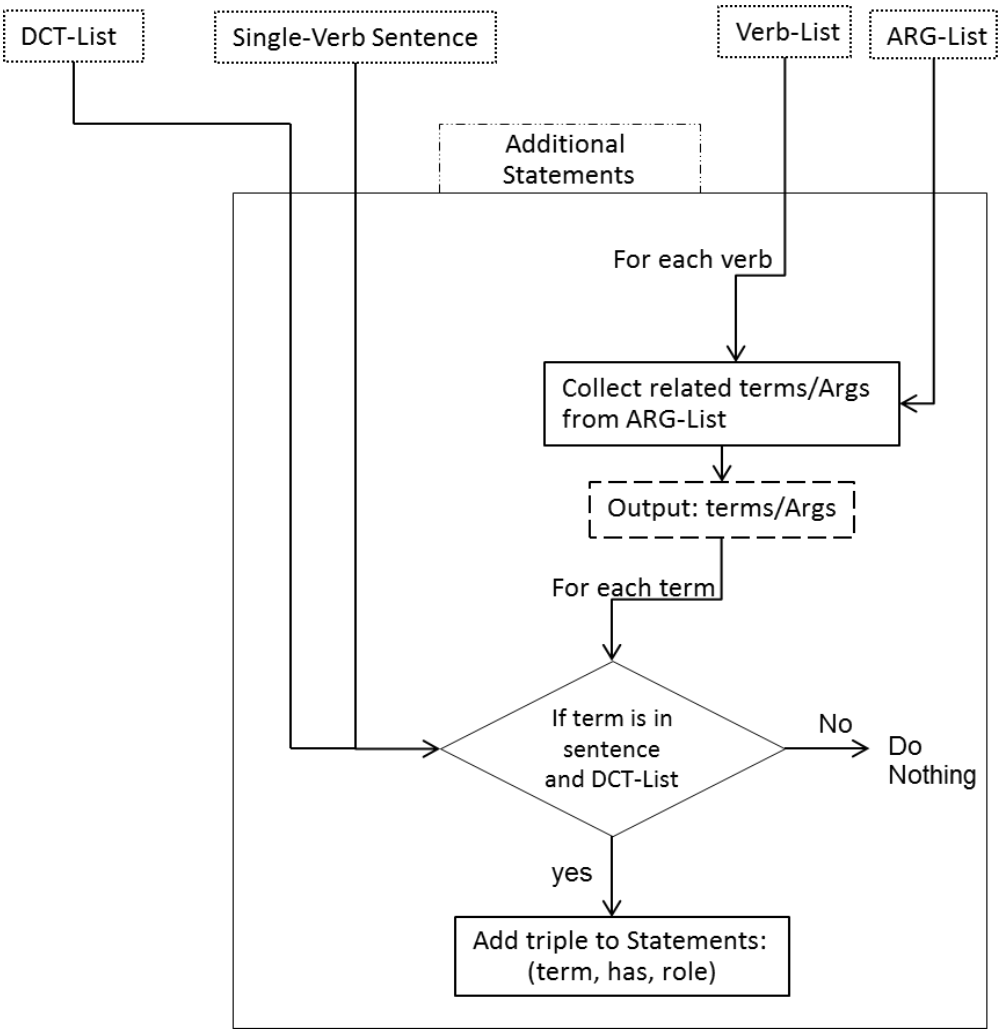

Figure 20 Final Processing

3.3.5 Output Generation: Visualization

The final phase of this process is providing output for users in two types of textual format and graphical visualization of triples that can be viewed by using *Gephi*. Textual output structure lists the original sentence at top, followed by split sentences with all triples extracted from each. Example shows how a typical sentence is presentenced in result.txt:

Example: Sample generated triples in a text format

Input: "in Singapore, there is close cooperation between medical and veterinary services which leads to good surveillance of zoonotic diseases.

Output:

(cooperation - is-in - Singapore)

(services - superClassOf - veterinary services medical)

(cooperation - between - services)

(Singapore - is - LOC)

(cooperation - superClassOf - close cooperation)

(surveillance - superClassOf - good surveillance)

(surveillance – has-role - result)

(cooperation – has-role - factor)

(surveillance - of - diseases)

(cooperation - leads-to - surveillance)

(cooperation - between - services)

(services - superClassOf - medical veterinary services)

(cooperation - is - LOC)

(diseases - superClassOf - zoonotic diseases)

(cooperation - leads-in - close)

This format of output is mainly readable for machines or query engines, whereas human users find it hard to follow triples and their connections; Hence output is translated into DOT format for further visualization via *Gephi*. DOT is able to describe both undirected and directed graphs; it gives a graph a name and defines relations as labeled edges between nodes of the graph. In this regard, each triple is separated into 3 parts of node1, edge and node2; all triples are gathered within 2 curly braces and each triple is represented in 1 line with ";" at the end. More features are listed in DOT's user manual including color, font, node shapes and size and etc which are not necessary to point in this project. [Gansner, Koutsofios, & North, 2010]

DOT file is readable by graph visualization tools such as Graphviz, *Gephi* and etc. *Gephi* is an interactive tool available open source for analyzing graphs and networks in a 3D environment. User can color or select a set of nodes, he can trace all paths and edged outgoing or incoming to a node; the graph is flexible to any changes such as stretching or compressing nodes and edges. And the result can be saved in JPEG, PNG or other image formats for other purposes. *Gephi* is a professional tool for network analysis but for sake of simplicity only applicable features to this project are discussed in this set of work. [Bastian, Heymann, & Jacomy, 2009]

Example: Sample generated triples in a DOT format

Input: Farmers have lost more than 20 cows.

Visualization Output: Figure 19

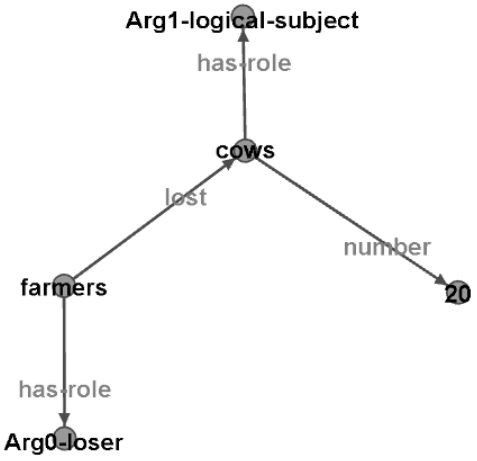

Figure 21 Full Visualization

4. Case Study and Experiments

As discussed in the literature review, there are not many tools that provide both syntactic and semantic relations between terms within a text. And for the purpose of evaluation, our results must be compared with tools which use a similar approach. Among all the tools summarized in literature review FRED has the closest output to our code. FRED[39] is an available parser that generates RDF/OWL ontologies and linked data triples from natural language sentences.

For this experiment, 15 sentences are randomly picked from an online ProMED report[40], and they are analyzed individually. Each sentence is followed by a table with three categories of outputs. The first one is "Human View". This set contains trivial relations among terms written by a human after reading that sentence. The second set is the result derived from FRED, and the third one is our code output. In two last categories, each triple is either marked as Yes/No/Not Useful or Question Mark (?) in the third column as "Evaluation".

- Yes: Meaningful and accepted triple according to input.
- No: Not meaningful or wrong information.
- Not Useful: meaningful but not valuable information.
- ? : Vague information, neither yes or no.

For each triple in the "Human View", relevant triples from FRED and our code are extracted and compared. The purpose of such comparison is figuring out how many of these triples, extracted by human, are covered in each tool.

In overall evaluation table, the total numbers of triples as well as qualified, not-qualified, not-useful and vague triples are calculated.

Lastly, in section 4.2, pros and cons of these two tools are summarized based on the observations.

[39] http://wit.istc.cnr.it/stlab-tools/fred/
[40] Please refer to Appendix IV.

4.1 Case Studies

Case Study 1: "To date farmers have lost more than 20 cows, and hundreds have been infected."

Table 8 Case Study 1

	Triples	Evaluation
Human View	Farmers - lost - cows Farmers - role - loser cows - role - thing_Lost Cows - number - 20 Hundreds - are - infected Hundred - of - cows	
FRED	Farmer - equivalentClass - Farmer	Yes
	Lose - POS - v	Not Useful
	Event - POS - E	Not Useful
	Lose_1 - a - Lose	Not Useful
	Lose_1 - a - Event	Yes
	Lose_1 - agent - farmer	Yes
	Lose_1 - patient - cow_1	Yes
	Date - POS - n	Yes
	Farmer - POS - n	Not Useful
	Cow_1 - a - Cow	Yes
	Cow - hasDataValue - 20	Yes
	Cow - hasQuality - MoreThan	Not Useful
	Hundred_1 - a - Hundred	Yes
	Cow - POS - n	Not Useful
	To - POS - v	Not Useful
	Infect - POS - v	Not Useful
	Infect_1 - a - Infect	Yes
	Infect_1 - a - Event	Yes
	Infect_1 - patient - hundred_1	Not Useful
	Hundred - POS - n	Not Useful
	To_1 - a - Date	Not Useful
	To - agent - to_1	No
	To - theme - lose_1	No
	MoreThan - POS - a	Not Useful
T2R	20 - is - EXT	Not Useful
	cows - has-role - patient,thing-falling	Yes
	farmers - lost - cows	Yes
	date - has-role - entity-losing thing	No
	farmers - has-role - entity-losing-thing	Yes

	farmers - lost-direct-object - 20	Yes
	farmers - lost-to - date	Not Useful
	cows - number - 20	Yes
	cows - infected - to	Not Useful
	hundreds - infected - to	Not Useful

Discussion:

1. Farmers –Lost– cows
   - FRED :

     (Lose_1 - a - Lose)
     (Lose_1 - a - Event)
     (Lose_1 - agent - farmer)
     (Lose_1 - patient - cow_1)

   - T2R:

     (farmers - lost - cows)

This Relation is the backbone of the sentence by having the verb relating subject and object of the sentence. This makes the triple the most informative triple among others.

FRED needs four triples to show the whole connection, while T2R is able to provide all the information in one line triple.

2. Farmers – role– Loser
   - FRED :

     (Lose_1 - agent - farmer)

   - T2R:

     (Farmers - has-role - entity-losing-thing)

Farmer has a main role as loser of cows. Both of the interpretations are correct.

3. cows – role– thing_lost
   - FRED :

     (Lose_1 - patient - cow_1)

   - T2R:

(Cows - has-role - patient, Thing-falling)

As shown above, both FRED and T2R are referring to cow as patient (or object) of the loss.

4. cows – number– 20

   - FRED :

     (Cows -hasDataValue- 20)

   - T2R:

     (cows - number - 20)

Both FRED and T2R provide the information

5. Hundreds – are – Infected , Hundreds –of– Cows

   - FRED :

     (Infect_1 -a- Infect)
     (Infect_1 -a- Event)
     (Infect_1 -patient- hundred_1)
     ?Connections to Cows Not Found

   - T2R:

     (Cows -infected - to)
     (Hundreds –infected – to)
     This triple " Date – is – entity-losing thing" shows that "Date" is identified as subject and it causes T2R Grammar parser to lose the connection.

Both FRED and T2R are unable to provide a set of complete triples.

Overall Evaluation:

Table 9 Overall Evaluation; Case Study 1

	FRED		T2R	
Total number of Triples	24		10	
Qualified Triples (Yes)	10/24	41.6%	5/10	50%
Unqualified Triples (No)	2/24	8 %	1/10	10%
Useless Triples (Not Useful)	12/24	50%	4/10	40%
Vague Triples (?)	-	-	-	-
Matched Triples	4/5		4/5	

Summary:

1- T2R extracts subject-verb-object as one triple, but FRED needs to provide four triples to explain the same triple.
2- T2R is able to refer to roles by their actual names, whereas FRED has only one type of role-set (agent and patient).
3- FRED generates 24 triples and almost half of them are unnecessary such as grammatical information about Part-Of-Speech tags ; T2R generates 10 triples without mentioning any grammatical points for an easier and cleaner set of results for the viewer.

Case Study 2: “We can’t reject anything.”

Table 10 Case Study 2

	Triples	Evaluation
Human View	We - can’t_reject - anything Anything –role – thing_to_reject We - role - rejecter	
FRED	Thing - POS - n	Not Useful
	Person_1 - a - Person	Yes
	Person - POS - n	Not Useful
	Reject_1 - a - Thing	?
	Reject _1 - agent - person_1	Yes
	Reject _1 - patient - reject_1	?
	Reject _1 - POS - v	Not Useful
	99t - POS - a	?
T2R	We - not-reject-direct-object - anything	Yes
	Not - has-role - AM-NEG	Not Useful
	Anything - has-role - thing-rejected	Yes
	Can - has-role - AM-MOD	Not Useful
	We - has-role - rejecter	Yes

Discussion:

1. We –can’t_reject– anything

   - FRED :

     (Reject _1 - a - Thing)
     (Reject _1 - a - Reject)
     (Reject _1 - agent - person_1)
     (Reject _1 - patient - reject_1)
     ?anything

   - T2R:

     (We -not-reject-direct-object- anything)

FRED can’t provide all information to satisfy subject-verb-object triple. In T2R “Anything” is recognized as direct object of verb reject, which matches the triple shown in Human View.

2. Anything –role– thing_to_reject

- FRED :

  None

- T2R:

  (Anything - has-role - thing_rejected)

FRED is unable to recognize any roles for “anything”; while T2R is providing the exact information.

3. We –role– rejecter

   - FRED :

     (Reject _1 - agent - person_1)

   - T2R:

     (Anything - has-role - thing_rejected)

FRED assigns “agent” to person_1 but there is no information about person_1 in its set of triples; while T2R is providing the exact information.

Overall Evaluation:

Table 11 Overall Evaluation; Case Study 2

	FRED		T2R	
Total number of Triples	8		5	
Qualified Triples (Yes)	2/8	25%	3/5	60%
Unqualified Triples (No)	-	-	-	-
Useless Triples (Not Useful)	3/8	37.5%	2/5	40%
Vague Triples (?)	3/8	37.5%	-	-
Matched Triples	0/3	0%	3/3	100%

Summary:

1- T2R extracts subject-verb-object as one triple, but FRED even by providing four triples is unable to make a right connection to the object.
2- T2R is able to refer to roles by their actual names, whereas FRED has only one type of role-set (agent and patient).

3- T2R matches all three triples from the human View, while FRED partially covers only the first one.

Case Study 3: "Information was being shared among clinics in an effort to find a common thread."

Table 12 Case Study 3

	Triples	Evaluation
Human View	Information - wasShared_Among - clinics Information - wasShared_To - find Find - object - Thread Information - role - thing_sahred Thread -role- thing_to_find	
FRED	Clinic - POS - n	Not Useful
	Find_1 - a - Event	?
	Find_1 - a - BecomingAware	Not Useful
	Find_1 - cognizer - effort_1	No
	Find_1 - phenomenon - thread_1	Yes
	Thread - POS - n	Not Useful
	Thread_1 - a - Commonthread	Not Useful
	Clinic_1 - a - Clinic	Yes
	Event - POS - E	Not Useful
	CommonThread - subclassOf - Thread	Yes
	Effort - POS - n	Not Useful
	Share_1 - a - Event	Not Useful
	Share_1 - patient - information_1	Yes
	Share_1 - among - clinic_1	Yes
	Share_1 - eventIn - effort_1	?
	BecomingAware - POS - F	Not Useful
	Information - POS - n	Not Useful
	Effort_1 - a - Effort	Yes
	Common - POS - a	Not Useful
	Information - equivalentClass - Information	Not Useful
T2R	Information - shared-among - clinics	Yes
	order - has-role - AM-LOC	No
	thread - has-role - AM-PNC	Not Useful
	information - shared-in - effort	No
	information - has-role - thing-shared	Yes
	clinics - has-role - AM-LOC	Yes
	information - has-role - finder	No
	thread - superClassOf - common_thread	Yes
	information - superClassOf - find-information	Not Useful

	thread - has-role - thing-found	Yes
	find - direct-object - thread	Yes

Discussion:

1. Information- wasShared_Among- clinics

   - FRED :

     (Share_1 - a - Event)
     (Share_1 - patient - information_1)
     (Share_1 - among - clinic_1)

   - T2R:

     (Information - shared-among - clinics)

   Both of the codes provide the information.

2. Information - wasShared_To - find , Find - object - Thread

   - FRED :

     (Find_1 - a - Event)
     (Find_1 - a - BecomingAware)
     (Find_1 - cognizer - effort_1)
     (Find_1 - phenomenon - thread_1)
     (Share_1 - eventIn - effort_1)

   - T2R:

     (information - shared-in - effort)
     (find - direct-object - thread)

FRED tries to convey the information with five triples, which requires extra efforts for user in order to follow the triples and find that information. T2R partially covers the information due to grammatical structure of the sentence.

3. Information - role - thing_shared

   - FRED :

     (Share_1 - patient - information_1)

- T2R:

  (information - has-role - thing-shared)

Both of the codes satisfy the target triple.

4. Thread - role - thing_to_find
   - FRED :

     None

   - T2R:

     (thread - has-role - thing-found)

FRED is unable to find the role for thread while T2R provided the exact information.

Overall Evaluation:

Table 13 Overall Evaluation; Case Study 3

	FRED		T2R	
Total number of Triples	20		11	
Qualified Triples (Yes)	6/20	30%	6/11	54.5%
Unqualified Triples (No)	1/20	5%	3/11	27.2%
Useless Triples (Not Useful)	11/20	55%	2/11	18.1%
Vague Triples (?)	2/20	10%	-	-
Matched Triples	3*/4	75%	3/4	75%

Summary:

1- Although FRED is able to provide equal amount of information for the first and the second triples, it requires extra processing for either human user or software agent to extract that information.
2- FRED generates 20 triples and only 30% of them were useful in this set, while more than half of the 11 triples from T2R are correct and useful.
3- Roles are better recognized by T2R than FRED.

Case Study 4: "Nothing has emerged as a common factor."

Table 14 Case Study 4

	Triples	Evaluation
Human View	Nothing - emerged_as - factor Nothing - role – thing-to-emerge Factor - is - common Factor – role – as-what	
FRED	Factor - POS - n	Not Useful
	CommonFactor - subClassOf - Factor	Yes
	Thing - POS - n	Not Useful
	Event - POS - E	Not Useful
	ComingtoBe - POS - F	Not Useful
	ComingToBe - disjoijntWith - ComingToBe	Not Useful
	Factor_1 - a - CommonFactor	Not Useful
	Common - POS - a	Not Useful
	Thing_1 - a - Thing	Not Useful
	Not_emerge_1 - a - Event	Not Useful
	Not_emerge_1 - a - comingToBe	?
	Not_emerge_1 - entity - thing_1	Yes
	Not_emerge_1 - place - factor_1	?
T2R	Factor - superClassOf - common_factor	Yes
	nothing - emerged-direct-object - factor	Yes
	nothing - has-role - thing-emerging	Yes
	factor - has-role - as-what	Yes

Discussion:

1. Nothing - emerged_as - factor

   - FRED :

     (Not_emerge_1 - a - Event)
     (Not_emerge_1 - a - comingToBe)
     (Not_emerge_1 - entity - thing_1)
     Connection to factor is lost

   - T2R:

     (nothing - emerged-direct-object - factor)

FRED cannot find the relation between subject and object; T2R completely matches the target triple.

2. Nothing - role - thing_to_emerge
    - FRED :

      None
    - T2R:

      (nothing - has-role - thing-emerging)

FRED does not have any information about the role for subject of the sentence; T2R matches the triple.

3. Factor - is - common
    - FRED :

      (Factor_1 - a - CommonFactor)
    - T2R:

      (Factor - superClassOf - common_factor)

Both of the codes satisfy the target triple.

4. Factor - role - as what
    - FRED :

      (Not_emerge_1 - place - factor_1)
    - T2R:

      (factor - has-role - as-what)

FRED triple does not make sense. T2R provides exactly the target triple.

Overall Evaluation:

Table 15 Overall Evaluation; Case Study 4

	FRED		T2R	
Total number of Triples	13		4	
Qualified Triples (Yes)	2/13	15%	4/4	100%
Unqualified Triples (No)	-	-	-	-
Useless Triples (Not Useful)	9/13	69%	-	-
Vague Triples (?)	2/13	15%	-	-
Matched Triples	1/4	25%	4/4	100%

Summary:

1- FRED fails at connecting subject to the object with verb; and poor performance on identifying roles.
2- 69% of the triples generated by FRED are Part-of-Speech tags and useless.

Case Study 5: "Dr. Oakley said the disease was highly contagious and he knew of 3 people who had caught it."

Table 16 Case Study 5

	Triples	Evaluation
Human View	DrOakley - said - statement Statement - about - diseases Disease - was - contagious He - knew - people People - number - 3 People - had_caught - it	
FRED	Male_1 - a - Male	Yes
	Male_1 - = - Oakley	Yes
	Male_1 - = - Dr	Yes
	Event - POS - E	Not Useful
	Disease_1 - a - Disease	Yes
	Disease_1 - hasQuality - HighlyContagious	Yes
	People - POS - n	Not Useful
	Awareness - POS - F	Not Useful
	Male - POS - a	Not Useful
	Neuter_1 - a - Thing	Not Useful
	Know_1 - a - Event	Not Useful
	Know_1 - AwarenessOf - people_1	Yes
	Know_1 - cognizer - Oakley	Yes
	Thing - POS - a	Not Useful
	Catch - POS - v	Not Useful
	Disease - equivalentClass - Disease	Not Useful
	Catch_1 - a - Catch	Not Useful
	Catch - agent - people_1	Yes
	Catch - patient - neuter_1	No
	Say_1 - a - Event	?
	Say_1 - message - disease_1	Yes
	Say_1 - speaker - Oakley	Yes
	Oakley - a - person	Yes
	Oakley - hasRole - Doctor	Yes
	HighlyContagious - POS - a	Not Useful
	Statement - POS - F	Not Useful
	Doctor - a - Role	Yes
	People_1 - a - People	Yes
	People_1 - hasDataValue - 3	Yes
	Disease - POS - n	Not Useful
T2R	DrOakley - is - PER	Yes

	Disease - said - he	Yes
	Contagious - more-detail - highly	Yes
	Disease - said - people	No
	DrOakley - said - he	No
	contagious - has-role - Utterance	Yes
	DrOakley - has-role - Sayer	Yes
	DrOakley - said - people	Not Useful
	People - has-role - Utterance	Not Useful
	he - has-role - Utterance	Not Useful
	contagious - subject - disease	Yes
	contagious - more-detail - highly	Yes
	people - number - 3	Yes
	he - knew-of - people	Yes
	people - has-role - attribute-of-arg1	?
	he - has-role - knower	Yes
	people - number - 3	Yes
	it - has-role - thing-gotten	Not Useful
	people - has-role - receiver	Yes
	people - caught-direct-object - it	Yes

Discussion:

1. DrOakley - said - statement , Statement - about - diseases , Disease - was - contagious

   - FRED :

     (Say_1 - a - Event)
     (Say_1 - message - disease_1)
     (Say_1 - speaker - Oakley)
     (Disease_1 - hasQuality - HighlyContagious)

   - T2R:

     (Disease - said - he)
     (Contagious - more-detail - highly)
     (Contagious - subject - disease)

Both satisfy the target triple chain.

2. He - knew - people , People - number - 3 , People - had_caught - it

   - FRED:

(Know_1 - a - Event)
(Know_1 - AwarenessOf - people_1)
(Know_1 - cognizer - Oakley)
(People_1 - hasDataValue - 3)
(Catch - agent - people_1)
(Catch - patient - neuter_1)

- T2R:

(people - number - 3)
(he - knew-of - people)
(people - caught-direct-object - it)

FRED generates a complex set of events and gathers objects and subjects around those events. T2R has a simpler representation and equal to target triple chain.

Overall Evaluation:

Table 17 Overall Evaluation; Case Study 5

	FRED		T2R	
Total number of Triples	30		20	
Qualified Triples (Yes)	15/30	50%	13/20	65%
Unqualified Triples (No)	1/30	3.3%	2/20	10%
Useless Triples (Not Useful)	13/30	43%	4/20	20%
Vague Triples (?)	1/30	3.3%	1/20	5%
Matched Triples	2/2	100%	2/2	100%

Summary:

1- 43% of FRED triples are not useful while T2R only generated 20% useless triples.
2- 69% of the triples generated by FRED are Part-of-Speech tags and useless.

Case Study 6: “It has affected herds in the Pihama-Awatuna area in South Taranaki and the Stratford-Midhirst district.”

Table 18 Case Study 6

	Triples	Evaluation
Human View	It - affected - herds It - role - affecting Herds - role - affected Herds - are_in - Pihama-Awatuna area - is_in - SouthTaranaki area - is_in - Stratford_Midhirst Stratford_Midhirst - is_a – district	
FRED	Area_1 - a - Area	Yes
	Area_1 - areaIn - district_1	Yes
	Area_1 - areaOf - pihama-awatuna	Yes
	Area_1 - locatedIn - SouthTaranaki	Yes
	Herd - POS - n	Not Useful
	Event - POS - E	Not Useful
	SouthTaranaki - a - place	Yes
	SouthTaranki - = - South-Taranaki-District	No
	Stratford_Midhirst - POS - a	Not Useful
	Stratfod_midhirst_District - subclassOf - District	Yes
	District_1 - a - Stratford_midhirt_District	Not Useful
	Affect_1 - a - Event	Not Useful
	Affect - agent - neuter_1	?
	Affect - patient - herd_1	Yes
	Affect - affectIn_area_1	Yes
	Neuter_1 - a - Thing	?
	Pihama-Awatuna - a - Place	Yes
	Thing - POS - a	Not Useful
	Herd_1 - a - Herd	Yes
	District - POS - n	Not Useful
	Herd - equivalentClass - Herd	Not Useful
	Affect - POS - v	Not Useful
	Area - POS - n	Not Useful
T2R	SouthTaranaki - has-role - AM-LOC	Yes
	It - affected-in - area	Yes
	Stratford-Midhirst - is - LOC	Yes
	PihamaAwatuna - is - LOC	Yes
	district - superClassOf - Stratford_district	Yes
	it - affected-in - district	Yes
	district - has-role - AM-LOC	Yes

	it - affected-in - SouthTaranaki	Yes
	it - affected-in - Stratford_Midhirst	Yes
	herds - has-role - thing-affected	Yes
	it - has-role - thing-affecting	Yes
	SouthTaranaki - is - LOC	Yes
	Area - has-role - AM-LOC	Yes
	Area - superClassOf - PihamaAwatuna_area	Yes
	it - affected-direct-object - herds	Yes

Discussion:

1. It - affected - herds
   - FRED :

     (Affect_1 - a - Event)
     (Affect - agent - neuter_1)
     (Affect - patient - herd_1)

   - T2R:

     (it - affected-direct-object - herds)

Both interpretations are correct.

2. It – role - affecting
   - FRED :

     (Affect - agent - neuter_1)

   - T2R:

     (it - is - thing-affecting)

FRED replaced “it” with neuter, but it needs more description for user/agent to recognize that. T2R matches the target triple.

3. Herd – role - affected
   - FRED :

     (Affect - patient - herd)

- T2R:

  (herd - has-role - thing-affected)

Both interpretations are correct.

4. Herds - are_in - Pihama-Awatuna

   - FRED :

     (Affect - patient - herd_1)
     (Affect - affectIn_area_1)
     (Area_1 - areaOf - piahama-awatuna)

   - T2R:

     (It - affected-in - area)
     (Area - superClassOf - PihamaAwatuna_area)
     (it - affected-direct-object - herds)

Both interpretations are correct.

5. area - is_in - SouthTaranaki

   - FRED :

     (Area_1 - locatedIn - SouthTaranaki)

   - T2R:

     (it - affected-in - SouthTaranaki)
     (It - affected-in - area)

   Both interpretations are correct.

6. area - is_in - Stratford_Midhirst

   - FRED :

     None

   - T2R:

     (it - affected-in - Stratford_Midhirst)
     (It - affected-in - area)

   FRED doesn't have the triple, but T2R matched it.

7. Stratford_Midhirst - is_a – district

   - FRED :

     (Stratfod_midhirst_District - subclassOf - District)

   - T2R:

     (district - superClassOf - Stratford_district)

Both interpretations are correct.

Overall Evaluation:

Table 19 Overall Evaluation; Case Study 6

	FRED		T2R	
Total number of Triples	23		15	
Qualified Triples (Yes)	9/23	39%	15/15	100%
Unqualified Triples (No)	1/23	4%	-	-
Useless Triples (Not Useful)	10/23	43%	-	-
Vague Triples (?)	2/23	8.6%	-	-
Matched Triples	6/7	85.7%	7/7	100%

Summary:

1- FRED generated 43% useless triples while the number of correct triples is less than half. T2R completely matched all triples from human View.

2- Roles are better named in T2R than FRED.

Case Study 7: “Cows in at least 8 herds have caught salmonella this season, twice as many as last year [2010] .”

Table 20 Case Study 7

	Triples	Evaluation
Human View	Cows - caught - salmonella Cows - in - herds Herds - number - 8 Cows - role - receiver Salmonella - role - thing_caught Event1 - of - salmonella Event1 - time - This_Season Event2 - time - Last_Year Event1 - twice_bigger - Event2	
FRED	Season - equivalentClass - Season	Yes
	Season - POS - n	Not Useful
	Current - POS - a	Not Useful
	Event - POS - E	Not Useful
	Year - POS - n	Not Useful
	B_1 - a - Event	?
	B - agent - D_1	?
	Salmonela - equivalentClass - Salmonella	Yes
	Cow - POS - n	Not Useful
	Last - POS - a	Not Useful
	Herd - equivalentClass - Herd	Yes
	Year_1 - a - last year	Yes
	Catch_1 – a - Event	Yes
	Catch_1 - a - Catch	Not Useful
	Catch - agent - cow_1	Yes
	Catch - patient - salmonella_1	Yes
	Catch - assosiatedWith - season_1	Yes
	Season_1 - a - Season	Yes
	Season_1 - hasQuality - current	Yes
	Salmonella_1 - a - Salmonella	Not Useful
	lastyear - subClassOf - Year	Not Useful
	D_1 - a - D	?
	Herd - POS - n	Not Useful
	Cow_1 - a - Cow	Yes
	Cow_1 - cowIn - herd_1	No(explained*)
	Twice - POS - a	Not Useful
	D - POS - n	?
	B - POS - v	?

	Quantity_1 - a - Quantity	Not Useful
	Catch - POS - v	Not Useful
	Herd_1 - a - Herd	Yes
	Herd_1 - hasDataValue - 8	Yes
	Salmonella - POS - n	Not Useful
	Quantity - POS - n	Not Useful
	2010 - a - date	Yes
T2R	cows - has-role - receiver	Yes
	cows - caught - salmonella	Yes
	salmonella - more_detail - twice	Not Useful
	herds - number - 8	Yes
	2010 - has-role - AM-TMP	Yes
	Salmonella - has-role - thing-gotten	Yes
	Salmonella - superClassOf - year-salmonella	No
	Cows - caught-direct-object - 2010	No
	Cows - in - herds	Yes
	Year - superClassOf - many_last_year	Not Useful

Discussion:

1. Cows - caught - salmonella

   - FRED :

     (Catch_1 – a - Event)
     (Catch_1 - a - Catch)
     (Catch - agent - cow_1)
     (Catch - patient - salmonella_1)

   - T2R:

     (Cows - caught_to - salmonella)

Both interpretations are correct.

2. Cows - in - herd

   - FRED :

     (Cow_1 - cowIn - herd_1)

   - T2R:

(cows - in - herds)

FRED connects these entities with cowIn relation, which is not popular/general for naming a relation. T2R matches the target triple.

3. Herd – number - 8

   - FRED :

     (Herd_1 - hasDataValue - 8)

   - T2R:

     (Herds - number - 8)

Both interpretations are correct.

4. Cows - role - receiver

   - FRED:

     (Catch_1 – a - Event)
     (Catch_1 - a - Catch)
     (Catch - agent - cow_1)

   - T2R:

     (Cows - has-role - receiver)

Both interpretations are correct.

5. Salmonella - role - thing_caught

   - FRED:

     (Catch_1 – a - Event)
     (Catch_1 - a - Catch)
     (Catch - patient - salmonella_1)

   - T2R:

     (Salmonella - has-role - thing_gotten)

Both interpretations are correct.

6. Event1 - of - salmonella , Event1 - time - This_Season , Event2 - time - Last_Year , Event1 - twice_bigger - Event2

- FRED:

  (Catch - assosiatedWith - season_1)
  (Season_1 - a - Season)
  (Season_1 - hasQuality - current)
  ?

- T2R:

  (salmonella - more_detail - twice)
  (Cows - caught-direct-object - 2010)
  ?

Both of these tools fail at finding proper relations, most probably because of grammatical structure of the sentence.

Overall Evaluation:

Table 21 Overall Evaluation; Case Study 7

	FRED		T2R	
Total number of Triples	35		10	
Qualified Triples (Yes)	14/35	40%	6/10	60%
Unqualified Triples (No)	1/35	2.8%	2/10	20%
Useless Triples (Not Useful)	15/35	42.8%	2/10	20%
Vague Triples (?)	5/35	14.3%	-	-
Matched Triples	4*/6	66%	5/6	83%

Summary:

1- Total amount of triples generated by FRED is more than twice of T2R triples, while T2R shows a higher percentage of qualified triples than FRED.
2- Explanation for * and triple (Cow_1 - cowIn - herd_1) : CowIn relation might be meaningful for human user but for software agents it needs more processing; Moreover CowIn is not a general term and adding it to the system vocabulary is an extra work, while it might not be useful in other datasets .

Case Study 8: "Another possible case had cleared without treatment."

Table 22 Case Study 8

	Triples	Evaluation
Human View	Case - cleared_without - treatment Treatment -role- cleaner Case -role- thing_made_clean	
FRED	PossibleCase - subclassOf - Case	Yes
	Emptying - POS - F	Not Useful
	Possible - POS - a	Not Useful
	NotTreatment - POS - n	Not Useful
	notTreatment - disjointWith - Treatment	?
	Case - POS - n	Not Useful
	Event - POS - E	Not Useful
	Not_treatment_1 - a - NotTreatment	Not Useful
	Case_1 - a - PossibleCase	Not Useful
	Case_1 - = - case_1	Not Useful
	Clear_1 - a - Emptying	Yes
	Clear_1 - theme - not_treatment_1	?
	Clear_1 - source - case_1	?
T2R	Case - has-role - thing-made-clean	Yes
	Treatment - has-role - AM-MNR	Not Useful
	Case - superClassOf - possible_case	Yes
	Case - cleared-without - treatment	Yes

Discussion:

1. Case - cleared_without - treatment

   - FRED :

     (Clear_1 - a - Emptying)
     (Clear_1 - theme - not_treatment_1)
     (Clear_1 - source - case_1)

   - T2R:

     (Case - cleared-without - treatment)

Both interpretations are correct.

2. Treatment – role - cleaner

   - FRED :

(Clear_1 - theme - not_treatment_1)

- T2R:

(Treatment - has-role - AM_MNR)

FRED doesn't refer to exact information and leaves it ambiguous. T2R is also giving some information that is not useful.

3. Case – role – thing_made_clean

- FRED :

(Clear_1 - source - case_1)

- T2R:

(Case - has-role - thing-made-clean)

Overall Evaluation:

Table 23 Overall Evaluation; Case Study 8

	FRED		T2R	
Total number of Triples	13		4	
Qualified Triples (Yes)	2/13	15%	3/4	75%
Unqualified Triples (No)	-	-	-	-
Useless Triples (Not Useful)	8/13	61.5%	1/4	25%
Vague Triples (?)	3/13	23.07%	-	-
Matched Triples	2/3	66.6%	2/3	66.6%

Summary:

1- FRED generated 61% useless triples while the number of correct triples is less than half.
2- Roles are better named in T2R than FRED.

Case Study 9: "The disease was devastating for farmers who had to treat affected cows with antibiotics, withdraw them from milking during and after treatment, and vaccinate the rest of the herd."

Table 24 Case Study 9

	Triples	Evaluation
Human View	Disease - was_devastating_for - farmers Farmers - treat - cows Farmer – role – who_treats Cows - role - to_be_treated Treatment - is - antibiotic Farmers - withdraw - cows Cows - withdrawn_from - milking Milking - withdrawal_duriation - druingTreatment Milking - withdrawal_duration - afterTreatment Farmers - vaccinate - restOfherd	
FRED	Milk - POS - verb	NotUsful
	Withdraw_1 - a - Removing	NotUsful
	Withdraw - type - Event	NotUsful
	withdraw - agent - farmer_1	Yes
	withdraw - theme - thing_1	No
	Disease_1 - a - Disease	Yes
	Cow - POS - n	NotUsful
	Rest - POS - n	NotUsful
	Affected - POS - a	NotUsful
	Antibiotioc_1 - a - antibiotic	Yes
	Herd - equivalentClass - Herd	Yes
	ExperienceObj - POS - F	NotUsful
	Milk_1 - a - Milk	Yes
	Milk_1 - during - treatment_1	Yes
	Treatment_1 - a - Treatment	Yes
	Vaccinate - POS - v	NotUsful
	Thing_1 - a - Thing	Not Useful
	Possession - POS - F	NotUsful
	CommunicateCategorization - POS - F	NotUsful
	Farmer_1 - a - Farmer	NotUsful
	Treatment - POS - n	Yes
	Farmer - equivalentClass - Farmer	NotUsful
	Rest_1 - a - Rest	NotUsful
	Rest_1 - restOf - herd_1	NotUsful
	Herd - POS - n	Yes
	Antibiotic - POS - n	NotUsful

	Devastate_1 - a - Event	NotUsful
	Devastate_1 - type - ExperiencerObj	?
	Devastate_1 - for - farmer_1	Yes
	Devastate_1 - stimulus - disease_1	Yes
	Cow_1 - a - AffectedCow	Yes
	Farmer - POS - n	Yes
	Treat_1 - a - Event	NotUsful
	Treat_1 - type - CommunicateCategorzation	Yes
	Treat_1 - item - cow_1	Yes
	Treat_1 - medium - Farmer_1	Yes
	Treat_1 - with - antibiotic	No
	Removing - POS - F	Yes
	AffectedCow - subClassOf - Cow	NotUsful
	Thing - POS - n	Yes
	Herd_1 - a - Herd	NotUsful
	Disease - euivalentClass - Disease	NotUsful
	Vaccinate_1 - a - Event	Yes
	Vaccinate - agent - farmer_1	?
	Vaccinate - patient - rest_1	Yes
	Milk_1 - a - Event	Yes
	Milk_2 - after - treatment_1	?
	Have_1 - a - Event	Yes
	Have_1 - type - possession	?
	Have_1 - theme - treat_1	NotUsful
	Have_1 - owner - farmer_1	?
	Disease - POS - n	NotUsful
	Antibiotic - equivalentClass - Antibiotic	NotUsful
	Milk - POS - verb	Yes
T2R	Disease - superClassOf - disease_devastating	Not Useful
	Disease - has-role - destroyer	Yes
	Devastating - for - farmers	Yes
	farmers - has-role - assumer-of-attribute	?
	cows - has-role - thing	Yes
	farmers - treat-with - antibiotics	Yes
	farmers - treat-direct-object - cows	Yes
	antibiotics - has-role - attribute	Yes
	cows - superClassOf - affected_cows	Yes
	farmers - withdraw-direct-object - them	Yes
	milking - has-role - removed-from	?
	farmers - has-role - entity-removing	No
	cows - withdraw-direct-object - them	No

	farmers - vaccinate-direct-object - rest	Yes
	rest - of - herd	Yes
	cows - vaccinate-direct-object - rest	No
	farmers - has-role - Vaccinator	Yes
	cows - has-role - Vaccinator	No

Discussion:

1. Disease - was_devastating_for - farmers

   - FRED:

     (Devastate_1 - a - Event)
     (Devastate_1 - type - ExperiencerObj)
     (Devastate_1 - for - farmer_1)
     (Devastate_1 - stimulus - disease_1)

   - T2R:

     (Devastating - for - farmers)
     (Disease - superClassOf - disease_devastating)

FRED introduces “devastate” as an event and “disease” is the stimulus for this event; in T2R grammar parser recognizes “devastating” as a modifier for “disease”. Both interpretations are correct but represent two different views of this phrase

2. Farmers - treat - cows

   - FRED:

     (Treat_1 - a - Event)
     (Treat_1 - type - CommunicateCategorzation)
     (Treat_1 - item - cow_1)
     (Treat_1 - medium - Farmer_1)

   - T2R:

     (farmers - treat-direct-object - cows)

Both satisfy the target triple.

3. Farmer – role – who_treats

   - FRED:

(Treat_1 - medium - Farmer_1)

- T2R:

  (farmers - has-role - assumer-of-attribute)

FRED assigns a role “medium” to farmer, depending on the definition of this term , this relation can be considered correct. According to PropBank frameset for verb “treat” farmer role is ARG0 or subject and “assumer_of_attribute”.

4. Cows - role - to_be_treated

   - FRED:

     (Treat_1 - item - cow_1)

   - T2R:

     (cow - has-role - thing)

   Both of the interpretations are correct.

5. Treatment - is - antibiotic

   - FRED:

     (Treat_1 - with - antibiotic)

   - T2R:

     (Farmer - treat-with – Antibiotics)

   Both interpretations are correct.

6. Farmers - withdraw - cows , Cows - withdrawn_from - milking

   - FRED:

     (Withdraw_1 - a - Removing)
     (Withdraw_1 - type - Event)
     (Withdraw_1 - agent - farmer_1)
     (Withdraw_1 - theme - thing_1)
     ? no connection to milking

   - T2R:

     (farmers - withdraw-direct-object - them)

(milking - has-role - removed-from)
?

FRED fails at making a connection to “milking”, and T2R cannot figure out the structural grammar behind the sentence.

7. Milking - withdrawal_duriation - druingTreatment , Milking - withdrawal_duration - afterTreatment

   - FRED:

     (Milk_1 - a - Milk)
     (Milk_1 - during - treatment_1)
     (Treatment_1 - a - Treatment)
     (Milk_2 - after - treatment_1)
     ? Milk_1 , Milk_2

   - T2R:

     ?

Grammatical structure of this phrase bans both tools from proper information mining.

8. Farmers - vaccinate - herd

   - FRED:

     (Vaccinate_1 - a - Event)
     (Vaccinate - agent - farmer_1)
     (Vaccinate - patient - rest_1)
     (Rest_1 - restOf - herd_1)

   - T2R:

     (farmers - vaccinate-direct-object - rest)
     (rest - of - herd)

Both Interpretations are correct.

Overall Evaluation:

Table 25 Overall Evaluation; Case Study 9

	FRED		T2R	
Total number of Triples	53		18	
Qualified Triples (Yes)	23/53	43.4%	11/18	61.1%
Unqualified Triples (No)	1/53	1.8%	4/18	22.2%
Useless Triples (Not Useful)	24/53	45.3%	1/18	5.5%
Vague Triples (?)	4/53	7.5%	2/18	11.1%
Matched Triples	7/8	87.5%	6/8	75%

Summary:

FRED generated 53 triples, four times more than T2R, and still less than 50% are correct; while in T2R 61% of triples are qualified.

Case Study 10: "Those cows that recovered came back into milk and resumed full production with no apparent ill effects, although that could take time."

Table 26 Case Study 10

	Triples	Evaluation
Human View	Cows - came_back_to - milk Cows - role - recovered Cows - resumed - full_production	
FRED	Milk - POS - n	Not Useful
	Recover_2 - a - time	No
	Recover_2 - agent - cow_1	No
	ApparentIll - POS - a	Not Useful
	Time - POS - n	Not Useful
	Event - POS - E	Not Useful
	Effect - POS - n	Not Useful
	Cow - POS - n	Not Useful
	Not_take_1 - a - Event	?
	Not_take_1 - entity - thing_1	?
	Not_take_1 - value - recover2	?
	Capacity_1 - a - Event	?
	Capacity_1 - entity - Thing_1	?
	Capacity_1 - value - recover_2	?
	Milk_1 - a - Milk	Yes
	Arriving - POS - F	Not Useful
	Recover - POS - v	Not Useful
	Thing_1 - a - Thing	Not Useful
	Milk - equivalentClass - Milk	Yes
	IllEffect - subClassOf - Effect	Not Useful
	Full - POS - a	Not Useful
	Production - POS - n	Not Useful
	Come_1 - a - Capacity	?
	Come_1 - type - Arriving	Not Useful
	Come_1 - theme - cow_1	?
	Come_1 - although - Capacity_1	?
	Cow_1 - a - Cow	Yes
	Cow_1 - backInto - milk_1	Yes
	Production_1 - a - FulllProduction	Yes
	Capacity - POS - F	?
	Capacity - disjointWith - Capacity	?
	Recover_1 - a - Event	Yes
	Recover - type - recover	?

	Recover_1 - agent - cow_1	No
	Thing - POS - n	Yes
	FullProduction - subClassOf - Production	Not Useful
	Back - POS - a	Not Useful
	ProcessResume - POS - F	Not Useful
	Resume_1 - a - Event	Yes
	Resume_1 - a - ProcessResume	Yes
	Resume_1 - agent - cow_1	No
	Resume_1 - although - not_take_1	?
	Resume_1 - instrument - effect_1	?
	Resume_1 - process - production_1	Yes
	ApparentIllEffect - subClassOf - IlllEffect	Not Useful
	Effect_1 - a - apparentIllEffect	Not Useful
T2R	Cows - has-role - that-which-was-sick	Yes
	cows - came-direct-object - milk	Yes
	cows - has-role - entity-in-motion-/-'comer'	Yes
	production - superClassOf - full_production	Yes
	effects - has-role - AM-MNR	Not Usefull
	cows - resumed-direct-object - production	Yes
	cows - resumed-with - effects	Yes
	effects - superClassOf - effects_apparent_ill	Yes
	time - has-role - thing-taken	Yes
	that - has-role - thing-taken	Not Useful
	could - has-role - AM-MOD	Not Useful
	that - take-direct-object - time	Not Useful

Discussion:

1. Cows - came_back_to- milk

   - FRED

     (Come_1 - a - Capacity)
     (Come_1 - type - Arriving)
     (Come_1 - theme - cow_1)
     (Cow_1 - a - Cow)
     (Cow_1 - backInto - milk_1)

   - T2R

     (cows - came-direct-object - milk)

FRED assigns “theme” as a role to “cow” which doesn’t make sense. T2R has proper translation.

2. Cows - role - recovered

   - FRED

     (Recover_1 - a - Event)
     (Recover - type - recover)
     (Recover_1 - agent - cow_1)

   - T2R

     Cows - is - that-which-was-sick

FRED considers recovery as an “Event” so “cow” should be “patient” of this recovery not the “agent”. T2R has correct translation.

3. Cows - resumed - full_production

   - FRED:

     (Resume_1 - a - Event)
     (Resume_1 - a - ProcessResume)
     (Resume_1 - agent - cow_1)
     (Resume_1 - process - production_1)

   - T2R:

     (cows - resumed-direct-object - production)
     (production - superClassOf - full_production)

Both interpretations are correct.

Overall Evaluation:

Table 27 Overall Evaluation; Case Study 10

	FRED		T2R	
Total number of Triples	46		14	
Qualified Triples (Yes)	10/46	21.7%	8/12	66.6%
Unqualified Triples (No)	4/46	8.6%	2/12	16.6%
Useless Triples (Not Useful)	18/46	39.1%	1/12	8.3%
Vague Triples (?)	14/46	30.4%	3/12	25%
Matched Triples	1/3	33.3%	3/3	100%

Case Study 11: "Farmers with salmonella in the herd should advise Fonterra [multinational dairy company], he said."

Table 28 Case Study 11

	Triples	Evaluation
Human View	Farmers - should_advise - Fonterra Farmer - role - giving_advice Fonterra – role – receiving_advice Farmers - own - affectedHerd Herd - affected_ with - salmonella	
FRED	Farmer - equivalentClass - Farmer	Yes
	Male_1 - a - Male	Not Useful
	Advise_1 - a - Telling	Yes
	Advise_1 - a - Event	?
	Advise_1 - hasQuality - %5D	?
	Advise_1 - message_B	?
	Advise_1 - speaker - farmer_1	Yes
	Herd - POS - n	Not Useful
	Event - POS - E	Not Useful
	Salmonella - equivalentClass - salmonella	Yes
	Multination - POS - a	Not Useful
	Male - POS - a	Not Useful
	Fonterra%5b - a - Organization	Yes
	%5D - POS - a	No
	Company - POS - n	Not Useful
	Telling - POS - F	Not Useful
	Herd_1 - a - Herd	Yes
	MultinationaDiaryCompany - subClassOF - DairyCompany	Yes
	Salmonella - POS - n	Not Useful
	Herd - equivalnetClass - Herd	Yes
	Salmonella_1 - a - Salmonella	Yes
	Salmonella_1 - salmonellaIn - herd_1	No(explained in summary)
	Say_1 - a - Event	?
	Say_1 - a - statement	Yes
	Say_1 - message - farmer_1	No
	Say_1 - speaker - male_1	Yes
	Statement - POS - F	Not Useful
	Dairy_company_1 - a - MultinationalDairyCompany	Yes
	Dairy_compnay_1 - = - Fonterra%5B	Yes

	Dairy - POS - n	Not Useful
	DairyCompany - subClassOf - Company	Yes
	DairyCompany - POS - n	Not Useful
	Farmer_1 - a - Farmer	Yes
	Farmer_1 - with - salmonella_1	No
T2R	salmonella -in- herd	Yes
	Farmers -with- salmonella	Yes
	Farmers - has-role - entity-giving-advice	Yes
	Fonterra_Multinational_Dairy_Company -is- advice	No
	should - has-role - AM-MOD	Not Useful
	Farmers -advise-direct-object- Fonterra_Multinational_Dairy_Company	Yes
	Statement -begins_with- Farmers	Yes
	he - has-role - Sayer	Yes
	he -said- Statement	Yes
	Fonterra_Multinational_Dairy_Company -is- MISC	Yes

Discussion:

1. Farmers - should_advise - Fonterra

   - FRED:

     (Advise_1 - a - Telling)
     (Advise_1 - a - Event)
     (Advise_1 - hasQuality - %5D)
     (Advise_1 - message_B)
     (Advise_1 - speaker - farmer_1)
     ? no connection to Fonterra

   - T2R:

     (Farmers -advise-direct-object- Fonterra_Multinational_Dairy_Company)

FRED partially represents the triple but fails to build a connection with Fonterra. T2R interpretation is correct.

2. Farmer - role - giving_advice

   - FRED:

     (Advise_1 - speaker - farmer_1)

- T2R:

  (Farmers -is- entity-giving-advice)

Both interpretations are correct.

3. Fonterra – role – receiving_advice
   - FRED:

     None

   - T2R:

     (Fonterra_Multinational_Dairy_Company -is- advice)

Both of these tools fail at generating triples.

4. Farmers - own - affectedHerd
   - FRED:

     (Farmer_1 - with - salmonella_1)
     (Salmonella_1 - salmonellaIn - herd_1)

   - T2R

     (Farmers - with – Salmonella)
     (Salmonella – in – Herd)

FRED creates a very specific type of relation between Salmonella and herd, “salmonellaIn”; which need extra translation for software agents to fully capture the meaning of it. T2R interpretation is correct.

5. Herd - affected_ with - salmonella
   - FRED:

     (Salmonella_1 - salmonellaIn - herd_1)

   - T2R:

     (salmonella -in- herd)

FRED has a bad choice for naming the relation as mentioned in previous statement. T2R interpretation is correct.

Overall Evaluation:

Table 29 Overall Evaluation; Case Study 11

	FRED		T2R	
Total number of Triples	34		10	
Qualified Triples (Yes)	15/34	44.1%	7/10	70%
Unqualified Triples (No)	4/34	11.7%	2/10	20%
Useless Triples (Not Useful)	11/34	32.3%	1/10	10%
Vague Triples (?)	4/34	11.7%	-	-
Matched Triples	2*/5	40%	4/5	80%

Summary:

1- Explanation for * : in statements of 4 and 5, poor choice for naming the relation "salmonellaIn", requires software agent extra processing and translation; otherwise this term is not general or trivial.

Case study 12: “"We don't know what's behind the outbreak," he said.”

Table 30 Case Study 12

	Triples	Evaluation
Human View	We - don’t_know - reason Reason - behind - outbreak	
FRED	Outbreak_1 - a - outbreak	Yes
	What_1 - a - What	Not Useful
	Male_1 - a - Male	Not Useful
	Event - POS - E	Not Useful
	Person - POS - n	Not Useful
	Awareness - POS - F	Not Useful
	Male - POS - a	Not Useful
	What - POS - n	Not Useful
	Know_1 - a - Event	Yes
	Know1 - a - Awareness	Yes
	Know_1 - behnid - outbreak_1	Yes
	Know_1 - cognizer - person_1	Yes
	Know_1 - content - what_1	Yes
	Outbreak - POS - n	Not Useful
	Person_1 - a - Person	Not Useful
	Say_1 - a - Event	Yes
	Say_1 - a - Statement	Yes
	Say_1 - message - know_1	Yes
	Say1_ - speaker - male_1	Yes
	Outbreak - equivalentClass - Outbreak	Not Useful
	Statement - POS - F	Not Useful
T2R	We - has-role - knower	Yes
	We -not-know-direct-object- reason	Yes
	reason - has-role - thing-known-or-thought	Yes
	he - has-role - Sayer	Yes
	not - has-role - AM-NEG	Not Useful
	he -said- Statement	Yes
	Statement -begins_with- We	Yes
	reason -behind- outbreak	Yes

Discussion:

1. We – don’t know – Reason , Reason - behind - outbreak
    - FRED

(Know_1 - a - Event)
(Know1 - a - Awareness)
(Know_1 - behnid - outbreak_1)
(Know_1 - cognizer - person_1)
(Know_1 - content - what_1)

- T2R:

(We -not-know-direct-object- reason)
(reason -behind- outbreak)

FRED refers to "We" as "person" without mentioning it in any other triple. T2R has correct interpretation.

Overall Evaluation:

Table 31 Overall Evaluation; Case Study 12

	FRED		T2R	
Total number of Triples	21		10	
Qualified Triples (Yes)	10/21	47.6%	7/8	87.5%
Unqualified Triples (No)	-	-	-	-
Useless Triples (Not Useful)	11/21	52.3%	1/8	12.5%
Vague Triples (?)	-	-	-	-
Matched Triples	1*/1	100%	1/1	100

Summary:

1- T2R detects "what's" in middle of any sentences and replaces it with "the reason", otherwise grammar parser cannot provide good syntactic information.

Case Study 13: "The article does not tell us specifically what genus and species of salmonella is affecting these animals."

Table 32 Case Study 13

	Triples	Evaluation
Human View	Article - doesn't_tell - specification Specification - of - salmonella Specification - such_as - genus Specification - such_as - species Salmonella - affected - animals Salmonella – role – affecting Animals – role – affected	
FRED	Request - POS - F	?
	Animal_1 - a - Animal	Yes
	NotGenus - POS - n	?
	NotGenus - disjoingWith - Genus	?
	Event - POS - E	Not Useful
	Animal_1 - equivalentClass - Animal	Not Useful
	Person - POS - n	Not Useful
	Tell_1 - a - Request	No
	Tell_1 - A - Event	Yes
	Tell_1 - addressee - person_1	Yes
	Tell_1 - message - not_genus_1	?
	Tell_1 - speaker - article_1	Yes
	Species - POS - n	Not Useful
	Salmonella_1 - equivalentClass - Salmonella	Not Useful
	Article - POS - n	Not Useful
	Affect_2 - a - Event	Yes
	Affect_2 - agent - species_1	Yes
	Affect_2 - patient - animal_1	Yes
	Animal - POS - n	Not Useful
	Affect_1 - a - Event	Yes
	Affect_1 - a - Affect	Not Useful
	Affect_1 - agent - not_genus_1	Not Useful
	Affect_1 - patient - animal_1	Yes
	Article_1 - a - article	Yes
	Species_1 - a - Species	Yes
	Species_1 - species_of - salmonella_1	Yes
	Specifically - POS - a	Not Useful
	Person_1 - a - Person	Not Useful
	Salmonella - POS - n	Not Useful
	Salmonella_1 - a - Salmonella	Yes

	Affect - POS - v	Not Useful
	Species - equivalentClass - Species	Not Useful
	Notgenus_1 - a - NotGenus	?
	Specifically_1 - a - Specifically	Not Useful
T2R	genus - has-role - thing-affecting--animate-only!	Yes
	species -affecting-direct-object- animals	Yes
	us - has-role - Hearer	Yes
	article -not-tell-more-detail- specifically	Yes
	not - has-role - AM-NEG	Not Useful
	specifically - has-role - AM-MNR	Not Useful
	what - has-role - Utterance	Yes
	article -tell- species	Not Useful
	genus -affecting-direct-object- animals	Yes
	genus -superClassOf- what-genus	Not Useful
	animals - has-role - Utterance	Yes
	genus - has-role - Utterance	Yes
	article -tell- genus	Not Useful
	article - has-role - Speaker	Yes
	animals - has-role - thing-affected	Yes
	species - has-role - thing-affecting--animate-only!	Yes
	species -of- salmonella	Yes
	species - has-role - Utterance	Yes
	article -tell- what	Not Useful
	article -tell- animals	Not Useful
	article -not-tell-direct-object- us	Yes

Discussion:

1. Article - doesn't_tell - specification ,
   Specification - of - salmonella,
   Specification - such_as - genus,
   Specification - such_as - species

   - FRED:

     (Tell_1 - a - Request)
     (Tell_1 - A - Event)
     (Tell_1 - addressee - person_1)
     (Tell_1 - message - not_genus_1)
     (Tell_1 - speaker - article_1)

(Affect_2 - a - Event)
(Affect_2 - agent - species_1)
(Affect_2 - patient - animal_1)
(Animal - POS - n)
(Affect_1 - a - Event)
(Affect_1 - a - Affect)
(Affect_1 - agent - not_genus_1)
(Affect_1 - patient - animal_1)
(Article_1 - a - article)
(Species_1 - a - Species)
(Species_1 - species_of - salmonella_1)

- T2R:

(article - not-tell-more-detail - specifically)
(article - not-tell-direct-object - us)
(genus - has-role - Utterance)
(species - has-role - Utterance)

Both tools have similar representation for the set of target triples.

2. Salmonella - affected - animals

- FRED

(Affect_1 - a - Event)
(Affect_1 - a - Affect)
(Affect_1 - agent - not_genus_1)
(Affect_1 - patient - animal_1)
(Article_1 - a - article)
(Species_1 - a - Species)
(Species_1 - species_of - salmonella_1)

- T2R

(species -of- salmonella)
(genus -affecting-direct-object- animals)
(species -affecting-direct-object- animals)

Both interpretations are correct.

3. Species_of_Salmonella – role – affecting

- FRED

  (Affect_2 - agent - species_1)

- T2R

  (species - has-role - thing-affecting--animate-only!)
  (species -of- salmonella)

Both interpretations are correct.

4. Animals – role – affected

   - FRED:

     (Affect_2 - patient - animal_1)

   - T2R:

     (animals - has-role - thing-affected)

Both Interpretations are correct.

Overall Evaluation:

Table 33 Overall Evaluation; Case Study 13

	FRED		T2R	
Total number of Triples	34		21	
Qualified Triples (Yes)	13/34	38.23%	14/21	66.7%
Unqualified Triples (No)	1/34	2.9%	-	-
Useless Triples (Not Useful)	15/34	44.1%	7/21	33.3%
Vague Triples (?)	5/34	14.7	-	-
Matched Triples	4/4	100%	4/4	100%

Case Study 14: “Neither does it tell us what antibiotics the organism is susceptible too.”

Table 34 Case Study 14

	Triples	Evaluation
Human View	It - doesn’t_tell_ about - antibiotics Organism - sensetive_to - antibiotics It – role – speaker us – role - listener	
FRED	Request - POS - F	Not Useful
	NotThing - POS - n	Not Useful
	NotThing - disjointWith - Thing	?
	Antibiotics - POS - v	No
	Event - POS - E	Not Useful
	Person - POS - E	Not Useful
	Tell_1 - a - Request	?
	Tell_1 - a - Event	Yes
	Tell_1 - addressee - person_1	Yes
	Tell_1 - message - thing_1	?
	Tell_1 - speaker - not_Thing_1	?
	Organism_1 - a - Organism	Yes
	Organism - hasQuality - suscpetible	Not Useful
	Too_1 - a - too	Not Useful
	Do - POS - v	Not Useful
	Neuter_1 - a - Thing	?
	Too - POS - a	Not Useful
	Not_thing_1 - a - NotThing	?
	Do_1 - a - Event	Not Useful
	Do_1 - a - Do	?
	Do_1 - agent - Not_Thing_1	?
	Do_1 - patient - netuer_1	?
	Thing - POS - n	Not Useful
	Thing - POS - a	Not Useful
	Antibiotic_1 - a - Antibiotic	Yes
	Antibiotic_1 - agent - thing_1	No
	Antibiotic_1 - patient - organism_1	Yes
	Antibiotic - hasQuality - Susceptible	Not Useful
	Person_1 - a - Person	Not Useful
	Susceptible - POS - a	Not Useful
	Organism - POS - n	Not Useful
	Thing_1 - a - thing	Not Useful
T2R	It - has-role - Speaker	Yes

	us - antibiotics-direct-object - us	No
	organism - susceptible-direct-object - us	No
	Neither - tell-direct-object - us	Yes
	Susceptible - more-detail - too	Not Useful
	It - tell-direct-object - us	Yes
	us - has-role - Hearer	Yes
	Neither - has-role - Speaker	No

Discussion:

1. It - doesn't_tell_ about - antibiotics
   - FRED

     (Tell_1 - a - Request)
     (Tell_1 - a - Event)
     (Tell_1 - addressee - person_1)
     (Tell_1 - message - thing_1)
     (Tell_1 - speaker - not_Thing_1)
     (Antibiotic_1 - a - Antibiotic)
     (Antibiotic_1 - agent - thing_1)

   - T2R

     None

2. Organism - sensetive_to - antibiotics
   - FRED:

     (Antibiotic_1 - patient - organism_1)
     (Antibiotic - hasQuality - Susceptible)

   - T2R

     None

3. It – role – speaker
   - FRED:

     (Tell_1 - speaker - not_Thing_1)

   - T2R

(It - has-role - Speaker)

FRED detected a wrong role for speaker, as no_thing is the object of sentence. T2R matches the target role.

4. us – role - listener

   - FRED:

     (Tell_1 - addressee - person_1)

   - T2R

     (us - has-role - Hearer)

FRED doesn't include what person_1 is referring to, but T2R has a correct interpretation.

Overall Evaluation:

Table 35 Overall Evaluation; Case Study 14

	FRED		T2R	
Total number of Triples	32		8	
Qualified Triples (Yes)	5/32	15.6%	4/8	50%
Unqualified Triples (No)	2/32	6.2%	3/8	37.5%
Useless Triples (Not Useful)	16/32	50%	1/8	12.5%
Vague Triples (?)	9/32	28.1%	-	-
Matched Triples	2*/4		2/4	50%

Summary:

1- This sentence has a poor grammar which blocks T2R grammar_parser to do correct parsing.
2- Statement 3 and 4 are not accepted from FRED because of poor choices of naming.
3- Solution: by changing some part of the sentence, more triples can be generated.

Case Study 15: “So if there is a vaccine, they must know the name of the organism responsible for the outbreak.”

Table 36 Case Study 15

	Triples	Evaluation
Human View	organism - responsiblefor - outbreak Vaccine - names - organism They – role – knower Organism –role – thing_to_know	
FRED	So - POS - a	Not Useful
	Responsible - POS - a	Not Useful
	Outbreak_1 - a - Outbreak	Yes
	There - POS - n	Not Useful
	There_1 - a - There	Not Useful
	There_1 - = - vaccine_1	No
	Event - POS - E	Not Useful
	Vaccine - POS - n	Not Useful
	Organism_1 - a - Organism	Yes
	Organism_responsiblefor - outbreak_1	Yes
	Awareness - POS - F	Not Useful
	Name - POS - n	Not Useful
	Name_1 - a - Name	Yes
	Name_1 - nameOf - organims_1	Yes
	Know_1 - a - Event	Yes
	Know_1 - a - Awareness	Yes
	Know_1 - hasQuality - so	No
	Know_1 - organizer - thing_1	Yes
	Know_1 - content - name_1	Not Useful
	Thing - POS - n	Not Useful
	Outbreak - POS - n	Not Useful
	Vaccine_1 - equivalentClass - vaccine	Not Useful
	Organism - POS - n	Not Useful
	Outbreak - equivalentClass - outbreak	Yes
	Vaccine_1 - a - vaccine	Not Useful
T2R	Is - subject - vaccine	No
	so - superClassOf - is-so	No
	so– has-role - AM-DIS	Not Useful
	must - has-role - AM-MOD	Not Useful
	they - has-role - knower	Yes
	responsible - for - outbreak	Yes
	responsible - has-role - thing-known-or-thought	Yes
	vaccine - has-role - AM-ADV	Not Useful

	if - has-role - AM-ADV	Not Useful
	there - has-role - AM-ADV	Not Useful
	name - of - organism	Yes

Discussion:

1. organism - responsiblefor - outbreak
   - FRED:

     (Organism_responsiblefor - outbreak_1)
   - T2R:

     (name - of - organism)
     (responsible - for - outbreak)
     ?

FRED has the triple but the name for relation is not a general term. T2R fails at connecting organism and responsible.

2. Vaccine - names - organism
   - FRED:

     (Organism_responsiblefor - outbreak_1)
   - T2R:

     (name - of - organism)
     ?

FRED has the triple but the name for relation is not a general term. T2R fails at connecting organism to vaccine.

3. They – role – knower
   - FRED:

     None
   - T2R:

     (they - has-role - knower)

FRED doesn't assign any roles to "They", but T2R matches the target triple.

4. Organism –role – thing_to_know

   - FRED:

     (Know_1 - content - name_1)
     (Name_1 - nameOf - organims_1)

   - T2R:

     (responsible - has-role - thing-known-or-thought)
     (name - of - organism)
     ?

FRED reaches the target triple chain by two triples, but T2R cannot connect organism and responsible.

Overall Evaluation:

Table 37 Overall Evaluation; Case Study 15

	FRED		T2R	
Total number of Triples	25		11	
Qualified Triples (Yes)	9/25	36%	4/11	36.4%
Unqualified Triples (No)	2/25	8%	2/11	18.2%
Useless Triples (Not Useful)	14/25	56%	5/11	45.4%
Vague Triples (?)	-	-	-	-
Matched Triples	3/4	75%	1/4	25%

Summary:

1- This sentence has a poor grammar which blocks T2R grammar_parser to do correct parsing.
2- Solution: by changing some part of the sentence, more triples can be generated.

4.2 Summary of Evaluations

In the previous section, both tools have been applied on a set of Case Studies and the results have been compared with Human-View triples. Among all triples in Human-View, (subject-verb-object) holds a high degree of importance as it reveals the core of each sentence. FRED attempts to cover this triple with a set of four or more triples. It refers to “verb” as an “Event”, “subject” as an “agent” and “object” as a “patient”. A good illustration of such representation is shown in Case Study 1:

Human-View triple is (Farmers - Lost - cows)

- FRED :

  (Lose_1 - a - Lose)
  (Lose_1 - a - Event)
  (Lose_1 - agent - farmer)
  (Lose_1 - patient - cow_1)

- T2R:

  (farmers - lost - cows)

FRED repeats this pattern for any terms in the sentence that it considers as an "Event". Whereas a simpler form of this representation can be obtained when merging or more analysis is applied. Such a simple form would move it closer to Human-View triple. On the other hand, T2R captures the equal amount of information in the form of (subject-verb-object) in just one triple, by using grammar parser.

Having followed the same approach, FRED produces a large set of triples as output, and in most of the case studies around 50% of these triples are rated as "Not Useful". Contrarily, the size of outputs generated by T2R is a half of FRED's. Moreover, large number of triples makes it more difficult for a human or software agent to follow/understand the relations.

Role labeling in FRED has a repetitive routine for all sentences. For instance, in each sentence we have Agent as subject, patient as object, etc. But T2R, describes each role based on the verb used in that sentence. These examples are taken from Case Studies 1 and 3:

> **Sentence:** "To date farmers have lost more than 20 cows, and hundreds have been infected."
>
> - FRED : (Lose_1 - agent - farmer)
> - T2R: (Farmers - has-role - entity-losing-thing)
>
> **Sentence:** "Information was being shared among clinics in an effort to find a common thread."
>
> - FRED : (Share_1 - patient - information_1)
> - T2R: (information - has-role - thing-shared)

As shown above, FRED does not rely on the verb meaning for labeling the terms while T2R creates a better presentation according to the verb of sentence.

The vocabulary for naming properties in subject-property-object triple is best to be as general as possible; In Case Studies 7 and 11, FRED uses different specific names for naming properties where the concept of these two relations are equal; T2R names both relations the same:

> **Sentence**: “Cows in at least 8 herds have caught salmonella this season, twice as many as last year [2010] .”
>
> - FRED : (Cow_1 - cowIn - herd_1)
> - T2R:(cows - in - herds)
>
> **Sentence**: “Farmers with salmonella in the herd should advise Fonterra [multinational dairy company], he said.”
>
> - FRED: (Salmonella_1 - salmonellaIn - herd_1)
> - T2R : (Salmonella – in – Herd)

Although, T2R representation format is closer to Human-View comparing to FRED and also the size of its output is smaller, both tools are highly dependent to their parsers. And when a parser is unable to fully parse the grammatical structure of a sentence, tools cannot generate correct/explicit information. The main reason for losing information in such cases is because of human writing style. Considering that parsers are being trained to process different sentences, they still cannot cover all writing styles 100%. However, by providing sentences with not complicated grammatical structure, tools are able to mine more information.

5. Conclusion and Future Work

5.1. Summary and Conclusion

Semantic web marks a new era of information technology, in which all pieces of data are connected together as a huge linked dataset of entities and relations. This constitutes a network of interconnected RDF triples. More importantly, all of this information is machine-readable. However, not all formats of data can be easily converted into RDF triple, especially when they do not follow a specific structure like tables or databases. Reports, articles, emails and any other textual documents contain a lot of information as well, but they represent an extensive challenge – they need to be translated into triples of information.

In this project, the main goal is to take advantage of text processing algorithms and convert any kind of text into a graph of data. This is done not only by considering syntactical structure of a text, but also covering semantic side of it. This is in contract to other tools in this field that merely look for certain patterns in the text or rely on a specific knowledgebase.

In the first step of the proposed approach, we pass text document to a grammar parser to identify all dependencies that exist among its terms. Then, in the semantic (second) step, the terms are labeled with roles of "who does/did what to whom, when, where and how". The results of both steps are analyzed and fused to obtain a set of RDF triples that represent all aspects – grammar and semantic based – of the processed text.

For evaluation, we compare the results obtained with our approach with the one obtained using FRED (a text to RDF convertor tool), and also with a set of triples created by a human. Our results expressed a better representation of texts in most of the case studies and they are fairly close to human-generated triples. While the FRED generate twice as many as triples as the proposed method, they are more complicated and they do not resemble the human-generated triples to such degree us the triples created by our method.

All in all, considering that parsers are not yet able to fully cover all writing styles, our code attempts to produce as many as triples possible from a text without relying on a knowledgebase or predefined patterns.

5.2 Future Works

Directions for potential future works include:

1- Event-Detection: These systems can benefit from this work by creating knowledge base of historical data and track/identify similar events from current data.

2- Query answering engines: Once, textual documents are translated into data-graph and are connected to other data sets, query engines are able to follow links based on related terms from a query and mine information.

3- Text Summarization: Another interesting and challenging area of further activities is related to summarization of a text into a smaller yet meaningful description. Such a process would involve ranking the relations extracted from that text and keeping the most valued information while omitting unnecessary ones.

Bibliography

Abulaish, M., & Dey, L. (2006). Biological relation extraction and query answering from MEDLINE abstracts using ontology-based text mining. *Data & Knowledge Engineering, ELSEVIER*.

Antoniou, G. Grigoris & Van Harmelen, Frank 2008, *A semantic Web primer*, 2nd ed, MIT Press, Cambridge, MA

Babko-Malaya, O. (2005). *Guidelines for PropBank framers.* Unpiblished Manual.

Bastian, M., Heymann, S., & Jacomy, M. (2009). *Gephi*: An Open Source Software for Exploring and Manipulating Networks. *Third International AAAI Conference on Weblogs and Social Media* (p. 2). AAAI.

Biemann, C. (2005). Ontology learning from text: A survey of methods. *LDV Forum*.

Bonial, C., Babko-Malaya, O., Choi, J. D., Hwang, J., & Palmer, M. (2010). *PropBank* Annotation Guidelines. *Center for Computational Language and Education Research, CU-Boulder*.

Cimiano, P., & Völker, J. (2005). Text2Onto, A Framework for Ontology Learning and Data-driven Change Discovery. *10th International Conference on Applications of Natural Language to Information Systems* (pp. 227-238). Alicante: Springer Berlin Heidelberg.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*.

Cunningham, e. a. (15 April 2011). *Text Processing with GATE (Version 6).* University of Sheffield Department of Computer Science.

Cunningham, H. (2002). GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, *36* (2), 223-254.

Dahab, M. Y., Hassan, H. A., & Rafea, A. (2007). TextOntoEx: Automatic ontology construction from natural English text. *Expert Systems with Applications, ELSEVIER*.

DuCharme, B. (2011). *Learning Sparql.* O'Reilly Media, Inc.

Gansner, E. R., Koutsofios, E., & North, S. (2010). *Drawing graphs with dot; User Manual.*

Gildea, D., & Jurafsky, D. (2002). Automatic Labeling of Semantic Roles. *Computational Linguistics*, 245-288.

Hratch Mangassarian, Hassan Artail, A general framework for subjective information extraction from unstructured English text, Data & Knowledge Engineering, Volume 62, Issue 2, August 2007, Pages 352-367, ISSN 0169-023X, 10.1016/j.datak.2006.10.001.

Heath, T., & Bizer, C. (2011). *Linked Data: Evolving the Web into a Global Data Space.*

Hebeler, J., Fisher, M., Perez- Lopez, A., & Dean, M. (2009). *Semantic Web Programming.* Wiley.

Marneffe, M.-C. d., & Manning, C. D. (2008). The *Stanford* typed dependencies representation. *URL http://nlp. Stanford. edu/software/dependencies_manual. pdf*.

Marneffe, M.-C. D., MacCartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. *Proceedings of LREC*, 449-454.

Novalija, I., Mladenic, D., & Bradesko, L. (2011). OntoPlus: Text-driven ontology extension using ontology content, structure and co-occurrence information. *Knowledge-Based Systems*, *24* (8), 1261–1276.

Peter Pin-Shan Chen. 1976. The entity-relationship model\—toward a unified view of data. *ACM Trans. Database Syst.* 1, 1 (March 1976), 9-36. DOI=10.1145/320434.320440 http://doi.acm.org/10.1145/320434.320440

Pradhan, S., Ward, W., Hacioglu, K., Martin, J., & Jurafsky, D. (2004). Shallow semantic parsing using support vector machines. *Proceedings of HLT/NAACL*, 233.

Presutti, V., Draicchio, F., & Gangemi, A. (2012). Knowledge Extraction Based on Discourse Representation Theory and Linguistic Frames. *18th International Conference, EKAW 2012* (pp. 114-129). Galway City: Springer Berlin Heidelberg.

Appendices

Appendix I: Definitions of the *Stanford* typed dependencies

[Marneffe & Manning, 2008]

abbrev: abbreviation modifier

> An abbreviation modifier of an NP is a parenthesized NP that serves to abbreviate the NP (or to define an abbreviation).
>
> "The Australian Broadcasting Corporation (ABC)" abbrev(Corporation, ABC)

acomp: adjectival complement

> An adjectival complement of a verb is an adjectival phrase which functions as the complement (like an object of the verb).
>
> "She looks very beautiful" acomp(looks, beautiful)

advcl: adverbial clause modifier

> An adverbial clause modifier of a VP or S is a clause modifying the verb (temporal clause, consequence, conditional clause, etc.).
>
> "The accident happened as the night was falling" advcl(happened, falling)
>
> "If you know who did it, you should tell the teacher" advcl(tell, know)

advmod: adverbial modifier

> An adverbial modifier of a word is a (non-clausal) adverb or adverbial phrase (ADVP) that serves to modify the meaning of the word.
>
> "Genetically modified food" advmod(modified, genetically)
>
> "less often" advmod(often, less)

agent: agent

> An agent is the complement of a passive verb which is introduced by the preposition "by" and does the action.
>
> "The man has been killed by the police" agent(killed, police)

"Effects caused by the protein are important" agent(caused, protein)

amod: adjectival modifier

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP.

"Sam eats red meat" amod(meat, red)

appos: appositional modifier

An appositional modifier of an NP is an NP immediately to the right of the first NP that serves to define or modify that NP. It includes parenthesized examples.

"Sam, my brother" appos(Sam, brother)

"Bill (John's cousin)" appos(Bill, cousin)

attr: attributive

An attributive is a complement of a copular verb such as "to be", "to seem", "to appear".

Currently, the converter only recognizes WHNP complements.

"What is that?" attr(is, What)

aux: auxiliary

An auxiliary of a clause is a non-main verb of the clause, e.g. modal auxiliary, "be" and "have" in a composed tense.

"Reagan has died" aux(died, has)

"He should leave" aux(leave, should)

auxpass: passive auxiliary

A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information.

"Kennedy has been killed" auxpass(killed, been)

aux(killed,has)

"Kennedy was/got killed" auxpass(killed, was/got)

cc: coordination

Coordination is the relation between an element of a conjunct and the coordinating conjunction word of the conjunct. (Note: different dependency grammars have different treatments of coordination. We take one conjunct of a conjunction (normally the first) as the head of the conjunction.)

"Bill is big and honest" cc(big, and)

"They either ski or snowboard" cc(ski, or)

ccomp: clausal complement

A clausal complement of a verb or adjective is a dependent clause with an internal subject which functions like an object of the verb, or adjective. Clausal complements for nouns are limited to complement clauses with a subset of nouns like "fact" or "report". We analyze them the same (parallel to the analysis of this class as "content clauses" in Huddleston and Pullum2002). Such clausal complements are usually finite (though there are occasional remnant English subjunctives).

"He says that you like to swim" ccomp(says, like)

"I am certain that he did it" ccomp(certain, did)

"I admire the fact that you are honest" ccomp(fact, honest)

complm: complementizer

A complementizer of a clausal complement (ccomp) is the word introducing it. It will be the subordinating conjunction "that" or "whether".

"He says that you like to swim" complm(like, that)

conj: conjunct

A conjunct is the relation between two elements connected by a coordinating conjunction, such as "and", "or", etc. We treat conjunctions asymmetrically: The head of the relation is the first conjunct and other conjunctions depend on it via the conj relation.

"Bill is big and honest" conj(big, honest)

"They either ski or snowboard" conj(ski, snowboard)

cop: copula

A copula is the relation between the complement of a copular verb and the copular verb. (We normally take a copula as a dependent of its complement; see the discussion in section 4.)

"Bill is big" cop(big, is)

"Bill is an honest man" cop(man, is)

csubj: clausal subject

A clausal subject is a clausal syntactic subject of a clause, i.e., the subject is itself a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clause is the complement of the copular verb. In the two following examples, "what she said" is the subject.

"What she said makes sense" csubj(makes, said)

"What she said is not true" csubj(true, said)

csubjpass: clausal passive subject

A clausal passive subject is a clausal syntactic subject of a passive clause. In the example below, "that she lied" is the subject.

"That she lied was suspected by everyone" csubjpass(suspected, lied)

dep: dependent

A dependency is labeled as dep when the system is unable to determine a more precise dependency relation between two words. This may be because of a weird grammatical construction, a limitation in the *Stanford* Dependency conversion software, a parser error, or because of an unresolved long distance dependency.

"Then, as if to show that he could, . . . " dep(show, if)

det: determiner

A determiner is the relation between the head of an NP and its determiner.

"The man is here" det(man, the)

"Which book do you prefer?" det(book, which)

dobj: direct object

The direct object of a VP is the noun phrase which is the (accusative) object of the verb.

"She gave me a raise" dobj(gave, raise)

"They win the lottery" dobj(win, lottery)

expl: expletive

This relation captures an existential "there". The main verb of the clause is the governor.

"There is a ghost in the room" expl(is, There)

infmod: infinitival modifier

An infinitival modifier of an NP is an infinitive that serves to modify the meaning of the NP.

"Points to establish are . . . " infmod(points, establish)

"I don't have anything to say" infmod(anything, say)

iobj: indirect object

The indirect object of a VP is the noun phrase which is the (dative) object of the verb.

"She gave me a raise" iobj(gave, me)

mark: marker

A marker of an adverbial clausal complement (advcl) is the word introducing it. It will be a subordinating conjunction different from "that" or "whether": e.g. "because", "when", "although", etc.

"Forces engaged in fighting after insurgents attacked" mark(attacked, after)

mwe: multi-word expression

The multi-word expression (modifier) relation is used for certain multi-word idioms that behave like a single function word. It is used for a closed set of dependencies between words in common multi-word expressions for which it seems difficult or unclear to assign any other relationships. At present, this relation is used inside the following expressions: rather than, as well as, instead of, such as, because of, instead of, in addition to, all but, such as, because of, instead of, due to. The boundaries of this class are unclear; it could grow or shrink a little over time.

"I like dogs as well as cats" mwe(well, as)

mwe(well, as)

"He cried because of you" mwe(of, because)

neg: negation modifier

The negation modifier is the relation between a negation word and the word it modifies.

"Bill is not a scientist" neg(scientist, not)

"Bill doesn't drive" neg(drive, n't)

nn: noun compound modifier

A noun compound modifier of an NP is any noun that serves to modify the head noun. (Note that in the current system for dependency extraction, all nouns modify the rightmost noun of the NP – there is no intelligent noun compound analysis. This is likely to be fixed once the Penn Treebank represents the branching structure of NPs.)

"Oil price futures" nn(futures, oil)

nn(futures, price)

npadvmod: noun phrase as adverbial modifier

This relation captures various places where something syntactically a noun phrase (NP) is used as an adverbial modifier in a sentence. These usages include: (i) a measure phrase, which is the relation between the head of an ADJP/ADVP/PP and the head of a measure phrase modifying the ADJP/ADVP; (ii) noun phrases giving an extent inside a VP which are not objects; (iii) financial constructions involving an adverbial or PP-like NP, notably the following construction $5 a share, where the second NP means "per share"; (iv) floating reflexives; and (v) certain other absolutive NP constructions. A temporal modifier (tmod) is a subclass of npadvmod which is distinguished as a separate relation.

"The director is 65 years old" npadvmod(old, years)

"6 feet long" npadvmod(long, feet)

"Shares eased a fraction" npadvmod(eased, fraction)

"IBM earned $ 5 a share" npadvmod($, share)

"The silence is itself significant" npadvmod(significant, itself)

“90% of Australians like him, the most of any country” npadvmod(like, most)

nsubj: nominal subject

A nominal subject is a noun phrase which is the syntactic subject of a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clauseis the complement of the copular verb, which can be an adjective or noun.

“Clinton defeated Dole” nsubj(defeated, Clinton)

“The baby is cute” nsubj(cute, baby)

nsubjpass: passive nominal subject

A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause.

“Dole was defeated by Clinton” nsubjpass(defeated, Dole)

num: numeric modifier

A numeric modifier of a noun is any number phrase that serves to modify the meaning of the noun.

“Sam eats 3 sheep” num(sheep, 3)

number: element of compound number

An element of compound number is a part of a number phrase or currency amount.

“I lost $ 3.2 billion” number($, billion)

parataxis: parataxis

The parataxis relation (from Greek for “place side by side”) is a relation between the main verb of a clause and other sentential elements, such as a sentential parenthetical, or a clause after a “:” or a “;”.

“The guy, John said, left early in the morning” parataxis(left, said)

partmod: participial modifier

A participial modifier of an NP or VP or sentence is a participial verb form that serves to modify the meaning of a noun phrase or sentence.

"Truffles picked during the spring are tasty" partmod(truffles, picked)

"Bill tried to shoot demonstrating his incompetence" partmod(shoot, demonstrating)

pcomp: prepositional complement

This is used when the complement of a preposition is a clause or prepositional phrase (or occasionally, an adverbial phrase). The prepositional complement of a preposition is the head of a clause following the preposition, or the preposition head of the following PP.

"We have no information on whether users are at risk" pcomp(on, are)

"They heard about you missing classes" pcomp(about, missing)

pobj: object of a preposition

The object of a preposition is the head of a noun phrase following the preposition, or the adverbs "here" and "there". (The preposition in turn may be modifying a noun, verb, etc.) Unlike the Penn Treebank, we here define cases of VBG quasi-prepositions like "including", "concerning", etc. as instances of pobj. (The preposition can be called a FW for "pace", "versus", etc. It can also be called a CC – but we don't currently handle that and would need to distinguish from conjoined prepositions.) In the case of preposition stranding, the object can precede the preposition (e.g., "What does CPR stand for?").

"I sat on the chair" pobj(on, chair)

poss: possession modifier

The possession modifier relation holds between the head of an NP and its possessive determiner, or a genitive 's complement.

"their offices" poss(offices, their)

"Bill's clothes" poss(clothes, Bill)

possessive: possessive modifier

The possessive modifier relation appears between the head of an NP and the genitive 's.

"Bill's clothes" possessive(John, 's)

preconj: preconjunct

A preconjunct is the relation between the head of an NP and a word that appears at the beginning bracketing a conjunction (and puts emphasis on it), such as “either”, “both”, “neither”).

“Both the boys and the girls are here” preconj(boys, both)

predet: predeterminer

A predeterminer is the relation between the head of an NP and a word that precedes and modifies the meaning of the NP determiner.

“All the boys are here” predet(boys, all)

prep: prepositional modifier

A prepositional modifier of a verb, adjective, or noun is any prepositional phrase that serves to modify the meaning of the verb, adjective, noun, or even another prepositon. In the collapsed representation, this is used only for prepositions with NP complements.

“I saw a cat in a hat” prep(cat, in)

“I saw a cat with a telescope” prep(saw, with)

“He is responsible for meals” prep(responsible, for)

prepc: prepositional clausal modifier

In the collapsed representation (see section 4), a prepositional clausal modifier of a verb, adjective, or noun is a clause introduced by a preposition which serves to modify the meaning of the verb, adjective, or noun.

“He purchased it without paying a premium” prepc without(purchased, paying)

prt: phrasal verb particle

The phrasal verb particle relation identifies a phrasal verb, and holds between the verb and its particle.

“They shut down the station” prt(shut, down)

punct: punctuation

This is used for any piece of punctuation in a clause, if punctuation is being retained in the typed dependencies. By default, punctuation is not retained in the output.

"Go home!" punct(Go, !)

purpcl: purpose clause modifier

A purpose clause modifier of a VP is a clause headed by "(in order) to" specifying a purpose. At present the system only recognizes ones that have "in order to" as otherwise the system is unable to distinguish from the surface representations between these and open clausal complements (xcomp). It can also recognize fronted "to" purpose clauses in sentences.

"He talked to him in order to secure the account" purpcl(talked, secure)

quantmod: quantifier phrase modifier

A quantifier modifier is an element modifying the head of a QP constituent. (These are modifiers in complex numeric quantifiers, not other types of "quantification". Quantifiers like "all" become det.)

"About 200 people came to the party" quantmod(200, About)

rcmod: relative clause modifier

A relative clause modifier of an NP is a relative clause modifying the NP. The relation points from the head noun of the NP to the head of the relative clause, normally a verb.

"I saw the man you love" rcmod(man, love)

"I saw the book which you bought" rcmod(book,bought)

ref : referent

A referent of the head of an NP is the relative word introducing the relative clause modifying the NP.

"I saw the book which you bought" ref (book, which)

rel: relative

A relative of a relative clause is the head word of the WH-phrase introducing it.

"I saw the man whose wife you love" rel(love, wife)

This analysis is used only for relative words which are not the subject of the relative clause. Relative words which act as the subject of a relative clause are analyzed as a nsubj.

root: root

The root grammatical relation points to the root of the sentence. A fake node “ROOT” is used as the governor. The ROOT node is indexed with “0”, since the indexation of real words in the sentence starts at 1.

“I love French fries.” root(ROOT, love)

“Bill is an honest man” root(ROOT, man)

tmod: temporal modifier

A temporal modifier (of a VP, NP, or an ADJP is a bare noun phrase constituent that serves to modify the meaning of the constituent by specifying a time. (Other temporal modifiers are prepositional phrases and are introduced as prep.)

“Last night, I swam in the pool” tmod(swam, night)

xcomp: open clausal complement

An open clausal complement (xcomp) of a VP or an ADJP is a clausal complement without its own subject, whose reference is determined by an external subject. These complements are always non-finite. The name xcomp is borrowed from Lexical-Functional Grammar.

“He says that you like to swim” xcomp(like, swim)

“I am ready to leave” xcomp(ready, leave)

xsubj: controlling subject

A controlling subject is the relation between the head of a open clausal complement (xcomp) and the external subject of that clause.

“Tom likes to eat fish” xsubj(eat, Tom)

Appendix II: *Stanford* Dependency Hierarchy

[Marneffe & Manning, 2008]

- root - root
    - dep - dependent
        - aux - auxiliary
            - auxpass - passive auxiliary
            - cop - copula
        - arg - argument
            - agent - agent
            - comp - complement
                - acomp - adjectival complement
                - attr - attributive
                - ccomp - clausal complement with internal subject
                - xcomp - clausal complement with external subject
                - complm - complementizer
                - obj - object
                    - dobj - direct object
                    - iobj - indirect object
                    - pobj - object of preposition
                - mark - marker (word introducing an advcl)
                - rel - relative (word introducing a rcmod)
            - subj - subject
                - nsubj - nominal subject
                    - nsubjpass - passive nominal subject
                - csubj - clausal subject
                    - csubjpass - passive clausal subject
            - cc - coordination
            - conj - conjunct
            - expl - expletive (expletive “there”)
            - mod - modifier
                - abbrev - abbreviation modifier
                - amod - adjectival modifier
                - ap*POS* - appositional modifier
                - advcl - adverbial clause modifier
                - purpcl - purpose clause modifier
                - det - determiner
                - predet - predeterminer

    - preconj - preconjunct
    - infmod - infinitival modifier
    - mwe - multi-word expression modifier
    - partmod - participial modifier
    - advmod - adverbial modifier
    - neg - negation modifier
    - rcmod - relative clause modifier
    - quantmod - quantifier modifier
    - nn - noun compound modifier
    - npadvmod - noun phrase adverbial modifier
    - tmod - temporal modifier
    - num - numeric modifier
    - number - element of compound number
    - prep - prepositional modifier
    - poss - possession modifier
    - possessive - possessive modifier (’s)
    - prt - phrasal verb particle
- parataxis - parataxis
- punct - punctuation
- ref - referent
- sdep - semantic dependent
    - xsubj - controlling subject

Appendix III: Alphabetical list of *POS* tags used in the Penn Treebank Project

- CC Coordinating conjunction
- CD Cardinal number
- DT Determiner
- EX Existential there
- FW Foreign word
- IN Preposition or subordinating conjunction
- JJ Adjective
- JJR Adjective, comparative
- JJS Adjective, superlative
- LS List item marker
- MD Modal
- NN Noun, singular or mass
- NNS Noun, plural
- NNP Proper noun, singular
- NNPS Proper noun, plural
- PDT Predeterminer
- POS Possessive ending
- PRP Personal pronoun
- PRP$ Possessive pronoun
- RB Adverb
- RBR Adverb, comparative
- RBS Adverb, superlative
- RP Particle
- SYM Symbol
- TO to
- UH Interjection
- VB Verb, base form
- VBD Verb, past tense
- VBG Verb, gerund or present participle
- VBN Verb, past participle
- VBP Verb, non-3rd person singular present
- VBZ Verb, 3rd person singular present
- WDT Wh-determiner
- WP Wh-pronoun
- WP$ Possessive wh-pronoun
- WRB Wh-adverb

Appendix IV: ProMED report

Link: **http://www.ProMEDmail.org/direct.php?id=20111214.3589**

Published Date: 2011-12-14 10:35:05
Subject: PRO/AH/EDR> Undiagnosed disease, bovine - New Zealand: (TK), salmonella, RFI
Archive Number: 20111214.3589
UNDIAGNOSED DISEASE, BOVINE - NEW ZEALAND: (TARANAKI), SALMONELLA,
REQUEST FOR INFORMATION
****

A ProMED-mail post
http://www.ProMEDmail.org
ProMED-mail is a program of the
International Society for Infectious Diseases
http://www.isid.org

Date: Tue 13 Dec 2011
Source: Stuff.co.NZ [edited]
http://www.stuff.co.nz/taranaki-daily-news/news/6129273/Cattle-disease-puzzle-for-vets

Taranaki dairy farmers are losing stock -- and thousands of dollars -- as vets tackle a severe outbreak of salmonella. The outbreak has stumped the vets, and they may bring in an epidemiologist to investigate it. To date farmers have lost more than 20 cows, and hundreds have been infected.

Kaponga vet Guy Oakley said nothing had emerged as a common factor. "We can't reject anything. It's incredibly frustrating."

Information was being shared among clinics in an effort to find a common thread. "We're trying to find funding for someone to look at the information we have and make sense of it. Nothing has emerged as a common factor." Dr Oakley said the disease was highly contagious and he knew of 3 people who had caught it.

It has affected herds in the Pihama-Awatuna area in South Taranaki and the Stratford-Midhirst district. Cows in at least 8 herds have caught salmonella this season, twice as many as last year [2010].

In the Stratford-Midhirst area there have been 3 confirmed cases. Another possible case had cleared without treatment. In one herd 200 animals were affected and in the other 2 herds, 60-100 animals caught the disease.

In the Pihama area, between 80 and 90 animals in one 500-cow herd have

caught salmonella and animals in 3 other herds have also had it this season [2011].

In the past, the disease had occurred in only 1 or 2 animals in a herd but the current outbreak had affected many, Dr Oakley, of Coastal Veterinary Services, said.

The disease was devastating for farmers who had to treat affected cows with antibiotics, withdraw them from milking during and after treatment, and vaccinate the rest of the herd.

Those cows that recovered came back into milk and resumed full production with no apparent ill effects, although that could take time.

One farmer last year [2010] had spent NZS 10 000 [about USD 7500] on drugs for his herd, and also faced substantial production losses, he said.

Like Dr Oakley, Stratford vet Craig Hassell is puzzled by the outbreak. The 2 vets are recommending vaccination, even though one farmer whose herd had been vaccinated had subsequently had 90 cows with salmonella. Dr Hassell said while vaccination might not be 100 percent successful, it would reduce the severity of the disease in any animals that caught it. Farmers with salmonella in the herd should advise Fonterra [multinational dairy company], he said. Dr Oakley said vets did not know what the risk factors of the current outbreak were.

"So to be absolutely safe, vaccination is the only tool. We don't know what's behind the outbreak," he said.

"Our conjectures have crumbled after investigations."

[Byline: Sue O'Dowd]

--
Communicated by:
ProMED-mail
<ProMED@ProMEDmail.org>

[The article does not tell us specifically what genus and species of salmonella is affecting these animals. Neither does it tell us what antibiotics the organism is susceptible too. We do know there is a vaccine. So if there is a vaccine, they must know the name of the organism responsible for the outbreak. The article mentions people affected but provides no details regarding recuperation, or hospitalization.

This article is very odd in that it leaves many questions. While

vaccinations are rarely 100 percent effective, it is still a valuable
too in saving individuals.

Clearly more information would be helpful on this case. - Mod.TG]

[A HealthMap/ProMED-mail interactive map of New Zealand can be seen at
http://healthmap.org/r/1xir. - Sr.Tech.Ed.MJ]

.....sb/tg/mj/dk

Appendix V: Complete outputs for examples from Senna and Stanfrod

Example: Capital Names in *Senna* and *Stanford*

Input: "Yet the Missouri based maker of Diamond, Premium Edge, Kirkland Signature, and other pet food brands has not called special attention to the expansion of the recall to cat food beyond amending a statement on the company Internet recall site.

***Stanford* Output:**

```
(ROOT
  (S (CC Yet)
    (NP
      (NP (DT the) (NNP Missouri))
      (VP (VBN based)
        (NP
          (NP (NN maker))
          (PP (IN of)
            (NP (NNP Diamond) (, ,) (NNP PremiumEdge) (, ,) (NNP
KirklandSignature)
              (CC and)
              (JJ other) (NN pet) (NN food) (NNS brands))))))
    (VP (VBZ has) (RB not)
      (VP (VBN called)
        (NP (JJ special) (NN attention))
        (PP (TO to)
          (NP
            (NP (DT the) (NN expansion))
            (PP (IN of)
              (NP (DT the) (NN recall)))))
        (PP (TO to)
          (NP (NN cat) (NN food)))
        (PP (IN beyond)
          (S
            (VP (VBG amending)
              (NP (DT a) (NN statement))
              (PP (IN on)
                (NP (DT the) (NN company) (NN Internet) (NN recall) (NN
site))))))))
    (. .)))

cc(called-19, Yet-1)
det(Missouri-3, the-2)
nsubj(called-19, Missouri-3)
partmod(Missouri-3, based-4)
dobj(based-4, maker-5)
nn(KirklandSignature-11, Diamond-7)
conj_and(KirklandSignature-11, PremiumEdge-9)
nn(brands-16, PremiumEdge-9)
nn(brands-16, KirklandSignature-11)
conj_and(KirklandSignature-11, other-13)
nn(brands-16, other-13)
```

```
nn(brands-16, pet-14)
nn(brands-16, food-15)
prep_of(maker-5, brands-16)
aux(called-19, has-17)
neg(called-19, not-18)
root(ROOT-0, called-19)
amod(attention-21, special-20)
dobj(called-19, attention-21)
det(expansion-24, the-23)
prep_to(called-19, expansion-24)
det(recall-27, the-26)
prep_of(expansion-24, recall-27)
nn(food-30, cat-29)
prep_to(called-19, food-30)
prepc_beyond(called-19, amending-32)
det(statement-34, a-33)
dobj(amending-32, statement-34)
det(site-40, the-36)
nn(site-40, company-37)
nn(site-40, Internet-38)
nn(site-40, recall-39)
prep_on(amending-32, site-40)
```

***Senna* Output:**

Yet	CC	O	O	-	O	O	O	(S1(S*
the	DT	B-NP	O	-	B-A0	O	O	(NP(NP*
Missouri	NNP	I-NP	S-LOC	-	E-A0	O	O	*
based	VBN	I-NP	O	based	S-V	O	O	*
maker	NN	E-NP	O	-	B-A1	O	O	*)
of	IN	S-PP	O	-	I-A1	O	O	(PP*
Diamond	NNP	B-NP	S-PER	-	I-A1	O	O	(NP(NP*
,	,	I-NP	O	-	I-A1	O	O	*
PremiumEdge	NNP	I-NP	S-PER	-	I-A1	B-A0	O	*
,	,	I-NP	O	-	I-A1	I-A0	O	*
KirklandSignature	NNP	I-NP	S-PER	-	I-A1	I-A0	O	*)
and	CC	I-NP	O	-	I-A1	I-A0	O	*
other	JJ	I-NP	O	-	I-A1	I-A0	O	(NP*

pet	JJ	I-NP	O	-	I-A1	I-A0	O	*
food	NN	I-NP	O	-	I-A1	I-A0	O	*
brands	NNS	E-NP	O	-	E-A1	E-A0	O	*))))
has	VBZ	B-VP	O	-	O	O	O	(VP*
not	RB	I-VP	O	-	O	S-AM-NEG	O	*
called	VBN	E-VP	O	called	O	S-V	O	(VP*
special	JJ	B-NP	O	-	O	B-A1	O	(NP(NP*
attention	NN	E-NP	O	-	O	E-A1	O	*)
to	TO	S-PP	O	-	O	B-AM-TMP	O	(PP*
the	DT	B-NP	O	-	O	I-AM-TMP	O	(NP(NP*
expansion	NN	E-NP	O	-	O	I-AM-TMP	O	*)
of	IN	S-PP	O	-	O	I-AM-TMP	O	(PP*
the	DT	B-NP	O	-	O	I-AM-TMP	O	(NP*
recall	NN	E-NP	O	-	O	I-AM-TMP	O	*))
to	TO	B-VP	O	-	O	I-AM-TMP	O	(S(VP*
cat	VB	E-VP	O	-	O	I-AM-TMP	O	(VP*
food	NN	S-NP	O	-	O	I-AM-TMP	O	(NP*)
beyond	IN	S-PP	O	-	O	I-AM-TMP	O	(PP*
amending	VBG	S-VP	O	amending	O	I-AM-TMP	S-V	(S(VP*
a	DT	B-NP	O	-	O	I-AM-TMP	B-A1	(NP(NP*

statement	NN	E-NP	O	-	O	I-AM-TMP	E-A1	*)
on	IN	S-PP	O	-	O	I-AM-TMP	B-A2	(PP*
the	DT	B-NP	O	-	O	I-AM-TMP	I-A2	(NP*
company	NN	I-NP	O	-	O	I-AM-TMP	I-A2	*
Internet	NNP	I-NP	S-MISC	-	O	I-AM-TMP	I-A2	*
recall	NN	I-NP	O	-	O	I-AM-TMP	I-A2	*
site	NN	E-NP	O	-	O	E-AM-TMP	E-A2	*))))))))))))))
.	.	O	O	-	O	O	O	*))

Example: Args Vs. Roles

Input: "Close cooperation between medical and veterinary services leads to good surveillance of zoonotic diseases."

***Senna* Output:**

Close	JJ	B-NP	O	-	B-A0	(S1(S(NP(NP*
cooperation	NN	E-NP	O	-	I-A0	*)
between	IN	S-PP	O	-	I-A0	(PP*
medical	JJ	B-NP	O	-	I-A0	(NP(ADJP*
and	CC	I-NP	O	-	I-A0	*
veterinary	JJ	I-NP	O	-	I-A0	*)
services	NNS	E-NP	O	-	E-A0	*)))
leads	VBZ	S-VP	O	leads	S-V	(VP*
to	TO	S-PP	O	-	B-A2	(PP*
good	JJ	B-NP	O	-	I-A2	(NP(NP*
surveillance	NN	E-NP	O	-	I-A2	*)
of	IN	S-PP	O	-	I-A2	(PP*
zoonotic	JJ	B-NP	O	-	I-A2	(NP*
diseases	NNS	E-NP	O	-	E-A2	*))))
.	.	O	O	-	O	*))

Example: Trimming Graph

Input: “36 of more than 200 workers have fallen ill in 2 weeks since the outbreak, their union said.”

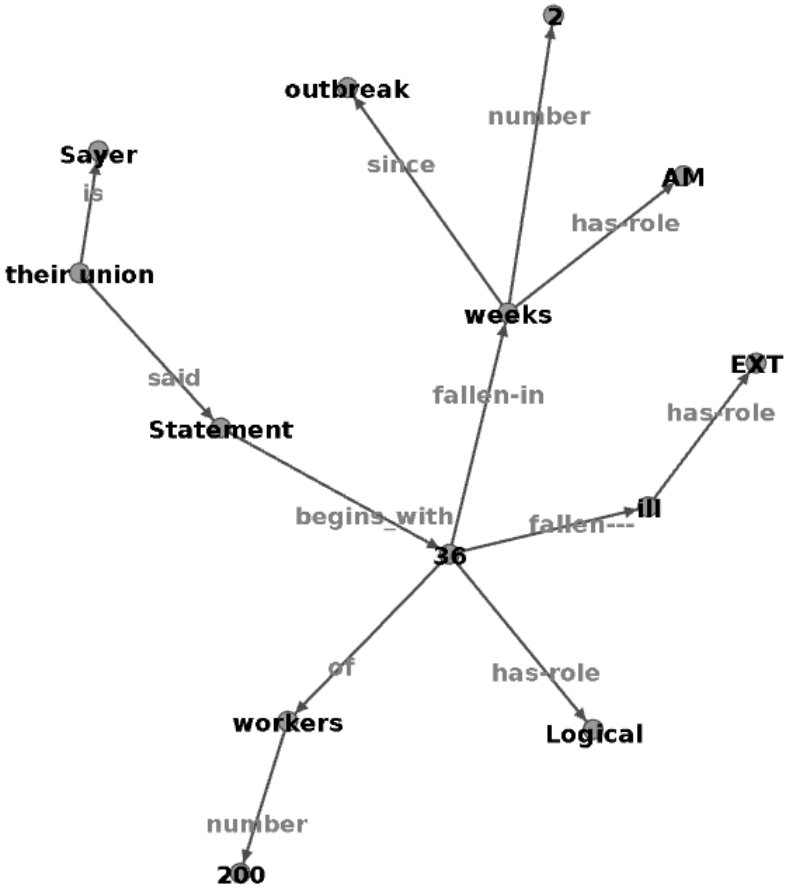

Figure 22 Complete graph

Example: Location

Input: “laboratory testing that conducted by state public health laboratories in Connecticut, Maryland, Pennsylvania and Wisconsin.”

laboratory	NN	B-NP	O	-	B-A1	(S1(NP(NP*
testing	NN	E-NP	O	-	E-A1	*)
that	WDT	S-NP	O	-	S-R-A1	(SBAR(WHNP*)
conducted	VBN	S-VP	O	conducted	S-V	(S(VP*
by	IN	S-PP	O	-	B-A0	(PP*
state	NN	B-NP	O	-	I-A0	(NP*
public	JJ	I-NP	O	-	I-A0	*
health	NN	I-NP	O	-	I-A0	*
laboratories	NNS	E-NP	O	-	I-A0	*))
in	IN	S-PP	O	-	I-A0	(PP*
Connecticut	NNP	S-NP	S-LOC	-	I-A0	(NP(NP*)
,	,	O	O	-	I-A0	*
Maryland	NNP	S-NP	S-LOC	-	I-A0	(NP*)
,	,	O	O	-	I-A0	*
Pennsylvania						
	and					
Wisconsin	NNP	S-NP	S-LOC	-	E-A0	(NP*))))))
.	.	O	O	-	O	*))