A Suite of Tools for Marking Up Textual Data for Temporal Text Mining Scenarios

Argyrios Vasilakopoulos, Michele Bersani and William J. Black

UMIST, Dept. of Computation
P.O. Box 88, M60 1QD, Manchester, UK
{a.vassilakopoulos@postgrad.umist.ac.uk, Michele@sibilo.co.uk, wjb@co.umist.ac.uk}

Abstract

Text Mining is a relatively new area of research, very interesting for both computational linguists and data miners. It involves collecting and analyzing quantities of textual data by domain experts, whose main task is the manual revision of markup. We describe a suite of tools used to simplify the process: the Parmenides System that consists of data warehouse, ontology, semi-automatic information extraction and data mining tools. Here we focus on the Annotation Editor which incorporates linguistic tools that initialize the markup automatically.

Introduction

Knowledge Management (KM) is important for the smooth operation of modern organisations as it defines the way information is structured, stored, retrieved and shared. The information existent and currently produced in such environments consists mainly of textual (unstructured) data which, in order to be useful, needs to be analysed, structured and stored appropriately. Towards the management of textual data a number of tools are already created, offering the domain expert an easy way to deal with the management task. In this paper we present a suite of tools we currently develop and use in the Parmenides framework for annotating textual data. Our tool incorporates functionality comparable to that of the GATE Annotation Tool (Gaizauskas et al, 1996) (i.e. the ability to manually annotate semantic elements of interest) as well as an NLP and IE pipeline which helps the human annotator by providing useful clues (semi automatic annotation). In that sense, our tool is more likely to be compared to the Amilcare (Ciravegna, 2001; Ciravegna et al., 2002), Amilcare-based (MnM, Melita, OntoMat) and Visual Text (Deane et. Al, 2001) tools, which allow the annotation of textual data interactively by users. Regarding the link of our tool to the ontology, the most relevant work seems to be the Ontology Forge System described in (Collier, 2003). However, that environment is more an ontology creation and population (ontology expansion) tool focused mainly on named entities, while our Annotation Editor targets the domain expert's real needs in marking up basic semantic elements and events, using the ontology as a resource rather than an end-product.
In the next sections we briefly describe our knowledge-based framework, the way we structure our data (Common Annotation Scheme) and the actual annotation editor.

The Parmenides System

The Parmenides System is an implementation of an "Ontology Driven Text Mining Framework". It consists of a document warehouse, ontology acquisition and editing tools, a set of processing modules for automatic ontology-linked information extraction, annotation-editing tools, and data-mining tools (Spiliopoulou & Mueller, 2003).
The Document Warehouse (DW) is the module responsible for collecting, converting, annotating and storing documents for further analysis. The DW consists, thus, of the Document Collector and Converter (DCC) and the Annotation Editor. The link behind the modules that make up the DW, as well as the interface between any two modules of the whole system, is the Common Annotation Scheme (CAS). Every module (except the DCC) understands and extends the CAS.
The existence of the Annotation Editor in the DW is of vital importance: it is the tool that allows the domain expert to analyse semantically the data relevant to his/her domain. Additional reasons for using such an editor are the following:

- The editor helps in the production of data which will constitute our resources. With this data we develop and tune processing (modules) and language (lexicons, ontologies, etc.) resources for each specific domain.
- The annotated data will reveal the users' targets (events and concepts of interest in general).
- The data marked up by domain expert is assumed correct and thus, constitutes a gold standard against which we can safely measure system performance.

Parmenides System Merits

As mentioned in the introduction, Parmenides System targets the domain expert's real needs. It is designed as a real world Text Mining system that can deal with large amounts of textual data for various case studies. The design has benefited from experience using GATE which is oriented towards the developer of annotation based text analysis, and is instead focused on the needs of end users who need to annotate texts in large quantities, related to their own private ontologies. Our tool:

- Can deal with large amounts of textual data. Our system is designed to store and collectively analyse large amounts of textual data. The Resource Manager imposes less of an overhead than competing systems when a pipeline of text analysers is executed.

- Is focused on conceptual annotation. Conceptual annotations are non span-based (although they build on text span annotations) annotations that constitute the user's actual need. They are the only way to describe events and relations, so the system has to cope with representing them successfully, in a user friendly way.
- Can automatically produce conceptual annotations through the BSEE compiler, using ontology information. With the authoring of appropriate rules, which will be in accordance to the ontology structures, there is the possibility to automatically extract event and relation conceptual annotations.

Common Annotation Scheme

The Parmenides Common Annotation Scheme (CAS) is an XML representation which consists of three types of annotations as described in (Rinaldi et al., 2003):

Structural Annotations: They define the structure of the document (head, body and further sections, paragraphs, sentences and tokens). These annotations are in-line annotations i.e. they contain the text spans they label.

Lexical Annotations: They identify lexical units of interest (entity instances), such as person's names, organizations, drug names, time expressions, etc. and are token-reference annotations, i.e. they do not contain textual spans but refer to unique token IDs instead.

Semantic/Conceptual Annotations: They are also token-reference annotations referring to specific (already marked up as lexical annotations) entities via co-referential IDs. They are used to mark entities, relationships and events.

An Editor for the CAS

Our analysis tools having been prototyped in the GATE environment, we evaluated the latter's suitability for annotation editing, and concluded that it did not enable users to fully benefit from the way that knowledge encoded in the ontology importantly constraints the slots and fillers that users may choose when correcting or creating fact annotations. The CAFETIERE (Conceptual Annotation of Facts, Events, Terms, Individual Entities and Relations) Annotation Editor offers constrained instance creation and slot filling. The editor allows users to manipulate documents converted to a skeletal CAS representation or a previously saved analysis. From a scratch document, users can annotate manually, or run a pipeline of NLP and IE analysers, and add, modify or delete annotations.

Users can add new annotations strictly in order, lexical first, then conceptual. Different instances of the same entities can be marked as co-referent, and complex representations of structured objects and events built up with all slots as constrained by the ontology. Figure 1 shows examples of annotations in the CAFETIERE Editor.

Automatic Analysis Pipeline

The automatic analysis pipeline aims at performing some basic natural language analysis and information extraction processes. Currently the pipeline is four level: tokenization, part-of-speech tagging, gazetteer and ontology lookup and semantic element end event extraction. More specifically the aforementioned steps collectively contribute the following information:

- *Tokenization:* Breaks texts in tokens and assigns a unique token id and orthographic classification tag to each of them. The orthographic tags are fully configurable and the ones currently used are the same as the ones used in GATE.
- *Part-of-Speech Tagging:* Assigns a POS classification label to each of the tokens of the text. We currently use a java implementation of the popular Brill Tagger (Vasilakopoulos, 2003), trained on texts tagged with the Penn TreeBank corpus tagset.
- *Gazetteer lookup:* lookup of single and multi word tokens in flat lists of names (companies, people, locations, etc…) suitable to specific domains. This kind of lookup returns only the conceptual class of the single/multi words matched, paying no attention to the specific attributes of the entities they represent.
- *Ontology lookup:* lookup of single or multi word tokens in ontologies like Protégé. The information acquired by ontology lookup is more meaningful compared to information produced by gazetteer lookup, and provides better input for the BSEE Compiler module, as attributes of concepts, as well as inter-conceptual relations are now accessible.
- *BSEE Compiler:* A rule compiler that elaborates the output of the 3 previous steps. BSEE Compiler applies sets of context sensitive rules on pre-processed texts and extracts basic semantic elements (named entities, temporal expressions), events and relations (by means of filling predefined template slots). We currently elaborate hand-crafted rules for specific case studies. However, work is in progress towards exploitation of "rule induction" via Machine Learning on already annotated corpora.

All information added by all these four steps of analysis is shown to the domain expert through the Editor's window and is fully modifiable (editable, extendable) according to the case study needs.

Ontology Integration

Regarding the Annotation Editor, our system interacts with the ontology in two ways. The first way is through the Ontology Lookup phase, and the second is through the compiler phase.
The Lookup essentially links surface forms from the current text to existent ontology instances. The user can

browse and manipulate the instances found according to his/her needs and reflect the changes to the ontology. A point worth to be mentioned here is the fact that the user can provide information about attributes of annotations (concepts-instances), which are constrained according to the same constraints defined in the Ontology (constraints on slots).
In a similar way, BSEE rules can use the structure of the concepts they look for and can impose constraints on concept attributes according to their type. However, in such cases, the rule writer has to know exactly the way the ontology is structured and the exact identity of the legal objects for each of the slots referred to.
It is, also, worth mentioning, the ability of the editor to use any ontology manager API available (in our case we have both integrated the Protégé and Wordmap ontology managers (Wordmap, 2002)). The only prerequisite is the fact that the integrated ontology has to implement a basic interface according to which the Annotation Editor can communicate with the resource.

Event (Multi-span) Annotation

The target of the Annotation Manager is the mark up of events (with their appropriate attributes) and temporal information. Currently, events are represented as templates with appropriate slots according to user defined, domain-specific ontologies. Annotation itself is very simple: the human annotator highlights manually the span of text that denotes the event instance (that is the core of the event, the verb or nominalization that indicates the event instance), associates it with a concept in the ontology used, (getting in this way the template for the event) and finally filling out the template slots. Additionally, at this point, the Annotation Editor proposes all possible candidate fillers for each of the slots of the event template: the human annotator can safely choose among the proposed legal filler objects for every attribute of the event instance he annotates. Event instances are then saved as PEvent elements according to the common annotation scheme. Instead of doing the event annotation manually, BSEE rules can me written in order to discover events and their appropriate fillers according to the core event phrase and its context.

Temporal Annotation

Through the Annotation Manager GUI three kinds of temporal annotations can be made:

Annotations of Temporal Expressions: Temporal expressions (TIMEX objects) are annotated by simply annotating the relevant textual span and assigning the appropriate features. The features available are currently the ones used in the TimeML specification (Ingria & Pustejovsky, 2002).

Timestamping of Event Instances: The timestamping of event instances is done indirectly, by selecting an appropriate TIMEX for the appropriate temporal slot of the event template. As every TIMEX has the value attribute, the filling of the appropriate event template slot with a TIMEX essentially indicates the temporal anchoring of the event instance in question.

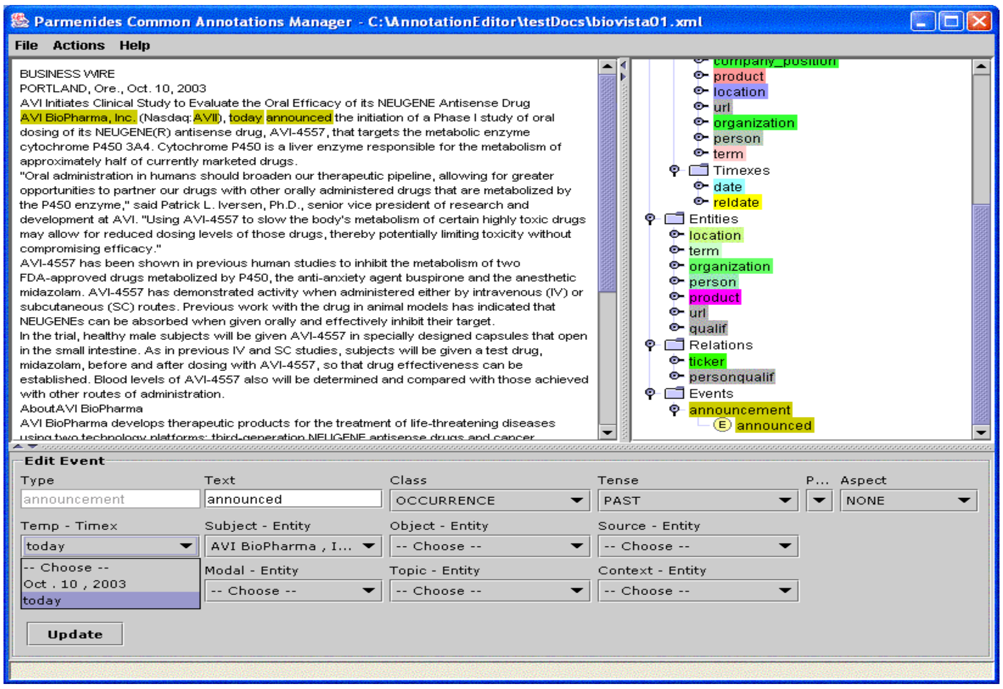
Event Temporal Ordering: This is done by creating special *Temporal Relations* between already marked up event instances. The possible temporal relations that can hold between any two events are again the ones proposed by the TimeML specification (under the TLINK element tag). However, this is totally modifiable and any set of temporal relations can be used, according to the user's own preference.
Again, with the authoring of domain and ontology specific rules all three kinds of temporal annotation can be performed automatically by the system.

Conclusions and Future Work

In this paper we described the Annotation Editor used in the Parmenides Text Mining Framework. We presented the CAFETIERE Annotator Manager which is designed to semi-automatically extract and annotate basic semantic elements, facts, and events in natural language texts. The CAFETIERE editor uses domain expert ontologies in order to help the annotator in performing its task, according to the specific application domain structure and rules.
In order to have a system that is both robust and really easily adaptable, there are still some things to be done. First, we need to further develop the ontology front-end. The fact that we can easily adapt *any* ontology manager the user wishes makes our tool usable in different environments. However, work still has to be done on consistency when new items are discovered in texts and update the ontology.
A second issue is linked to the authoring of BSEE rules. At present, rules are hand-crafted, by linguist experts who, collectively with domain experts, analyse specific domains. One of our future plans is the use of machine learning techniques in order to induce rule sets from already annotated data by domain experts. Having fulfilled these extensions, it will be easier for our tool to be adapted with the least effort to completely new domains, as well as the annotation task which is the basis for successful text mining scenarios, will be considerably facilitated.

Acknowledgements

This work has been partly funded by the European Union and the project partners in Parmenides Project (IST-2001-39023).


Parmenides Common Annotations Manager - C:\AnnotationEditor\testDocs\biovista01.xml

File Actions Help

BUSINESS WIRE
PORTLAND, Ore., Oct. 10, 2003
AVI Initiates Clinical Study to Evaluate the Oral Efficacy of its NEUGENE Antisense Drug
AVI BioPharma, Inc. (Nasdaq:AVII), today announced the initiation of a Phase I study of oral dosing of its NEUGENE(R) antisense drug, AVI-4557, that targets the metabolic enzyme cytochrome P450 3A4. Cytochrome P450 is a liver enzyme responsible for the metabolism of approximately half of currently marketed drugs.
"Oral administration in humans should broaden our therapeutic pipeline, allowing for greater opportunities to partner our drugs with other orally administered drugs that are metabolized by the P450 enzyme," said Patrick L. Iversen, Ph.D., senior vice president of research and development at AVI. "Using AVI-4557 to slow the body's metabolism of certain highly toxic drugs may allow for reduced dosing levels of those drugs, thereby potentially limiting toxicity without compromising efficacy."
AVI-4557 has been shown in previous human studies to inhibit the metabolism of two FDA-approved drugs metabolized by P450, the anti-anxiety agent buspirone and the anesthetic midazolam. AVI-4557 has demonstrated activity when administered either by intravenous (IV) or subcutaneous (SC) routes. Previous work with the drug in animal models has indicated that NEUGENEs can be absorbed when given orally and effectively inhibit their target.
In the trial, healthy male subjects will be given AVI-4557 in specially designed capsules that open in the small intestine. As in previous IV and SC studies, subjects will be given a test drug, midazolam, before and after dosing with AVI-4557, so that drug effectiveness can be established. Blood levels of AVI-4557 also will be determined and compared with those achieved with other routes of administration.
AboutAVI BioPharma
AVI BioPharma develops therapeutic products for the treatment of life-threatening diseases using two technology platforms: third-generation NEUGENE antisense drugs and cancer

Edit Event

Type: announcement
Text: announced
Class: OCCURRENCE
Tense: PAST
P...:
Aspect: NONE
Temp - Timex: today (-- Choose --, Oct . 10 , 2003, today)
Subject - Entity: AVI BioPharma , I...
Object - Entity: -- Choose --
Source - Entity: -- Choose --
Modal - Entity: -- Choose --
Topic - Entity: -- Choose --
Context - Entity: -- Choose --

Update

Figure 1: Annotations in CAFETIERE

References

Ciravegna F., Dingli A., Petrelli D., Wilks Y. (2002) Document Annotation via Adaptive Information Extraction. Poster at the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval August 11-15, 2002, in Tampere, Finland.

Ciravegna F. (2001) Adaptive Information Extraction from Text by Rule Induction and Generalisation. In Proceedings of 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle, August 2001.

Collier N., Takeuchi K., Kawazoe A. (2003) A Framework for integrating Deep and Shallow Semantic Structures in Text Mining.

Deane P., Hilster D., Meyers A. (2001) Text Processing in an Integrated Development Environment (IDE): Integrating Natural Language Processing (NLP) Techniques. In PC AI, Volume 15, Issue 5, Sept/Oct 2001.

Gaizauskas R., Rodgers P., Cunningham H., Humphreys H. (1996) GATE User Guide. Department of Computer Science, University of Sheffield, Department of Computer Science, University of Sheffield. 1996. http://gate.ac.uk/.

Ingria B., Pustejovsky J. (2002) TimeML Specification. Available at http://www.cs.brandeis.edu/~jamesp/arda/time/documentation/TimeML-Draft3.0.9.html

Rinaldi F., Dowdall J., Hess M., Ellman J., Zarri G.P., Bernard L., Karanikas H. (2003) Multilayer Annotations in Parmenides. Second International Conference on Knowledge Capture 2003, Florida, USA.

Spiliopoulou M., Mueller R.M. (2003) PARMENIDES: Ontology Driven Temporal Text Mining on Organisational Data for Extracting Temporal Valid Knowledge. In the 14th on Machine Learning and the 7th on Principles and Practice of Knowledge Discovery in Databases Joint European Conferences (ECML/PKDD 2003), September 22-26, 2003, Cavtat, Croatia.

Vasilakopoulos A. (2003) Improved Unknown Word Guessing By Decision Tree Induction for POS Tagging with TBL. In Proceedings of CLUK Conference, Edinburgh, 2003.

Wordmap (2002) Wordmap Introduction. Found at http://www.wordmap.com/downloads/white_papers.html