

IeXML: towards an annotation framework for biomedical semantic types enabling interoperability of text processing modules

Dietrich Rebolz-Schuhmann¹, Harald Kirsch¹, Goran Nenadic²

¹ European Bioinformatics Institute (EBI), Wellcome Trust Genome Campus, Hinxton, Cambridge, UK

² School of Informatics, University of Manchester, Manchester, UK
rebholz@ebi.ac.uk, kirsch@ebi.ac.uk, g.nenadic@manchester.ac.uk

Abstract

Identification of biological, medical, chemical and other terms in the biomedical literature is the key to successful information extraction. Annotation of these semantic types is subject to ongoing research and new solutions are constantly being proposed. Unfortunately, the corresponding text processing modules usually fail to be easily incorporated into a different text mining infrastructure, because more often than not they follow special format requirements leading to substantial integration overheads. Furthermore, they are rarely ready to incorporate annotations of other modules (e.g. POS taggers or other named entity recognisers) to enable additional complex annotations (e.g. event identification). We have carefully analysed the requirements for modules in an information extraction pipeline to be interoperable. We propose a generalised framework that defines an annotation and data exchange format for a set of components (namely tokeniser, POS tagger, NER, etc.) that facilitate annotation of semantic types. The suggested annotation types include tokens, terms, chunks and sentences, and semantic categories are attributed to terms. In our annotation schema, the annotations follow a hierarchical order, whereas the order of text processing modules is not predefined. We define a basic but extendable set of tags and attributes that the modules need to provide in order to facilitate interoperability for annotation of semantic types. In the case of ambiguities, alternative annotations are gathered and kept in a list. We also propose an XML-based implementation (called IeXML) of the framework, which serves as a general exchange format between text mining modules. Exchange and composition of such modules will increase collaborative progress of research work done in the bio-text mining community.

1. Background

Information extraction (IE) and text mining (TM) from the scientific literature are complex tasks that require a number of processing resources (e.g. POS taggers, gazetteers, named entity recognisers, parsers, etc.). Identification of named entities and terms (such as names of genes, proteins, gene products, organisms, drugs, chemical compounds, etc.), in particular, is the key factor for accessing and integrating the information stored in the literature [10]. Although techniques for term identification are becoming more widely-used in biomedical and bioinformatics research [6, 8, 9], there are very few community-wide efforts to facilitate their interoperability [1, 3]. Interoperability is here regarded as the ability to combine modules and exchange data, meta-data and other resources to maximise their re-use [5, 7]. This is in contrast to widespread attempts to standardise exchange and semantic descriptions of non-textual biomedical data (e.g. through SBML [15]), ontological descriptions (e.g. through the OBO format [12]) and specifications of bioinformatics services (e.g. through the Taverna/myGrid framework [16]).

Still, at present, many researchers in bio-text mining openly contribute and disseminate data (e.g. annotated corpora such as Genia¹ or BioCreActive²) or modules (such as POS taggers, parsers and specific named entity recognisers). Unfortunately, adaptation of available resources to a convenient format usually requires substantial efforts, since they have been designed to meet a particular aim and tend not to comply with any common data format, which leads to overheads of wrapping modules and adapting input and output formats. This has significantly reduced the ability of many users as well as researchers to reuse available modules.

Remote procedure calls, Web Services, common APIs or platforms are usual ways to achieve service-oriented

interoperability of software modules. An example for an open source JAVA-based platform is IBM's UIMA (Unstructured Information Management Architecture³) that has recently been introduced to the bio-text mining community for creating and composing text processing modules and integrating their results. The UIMA platform primarily targets *solution development and integration* rather than interoperability of individual text processing components. It addresses the challenge of "standardizing and improving the process of development, deploying and integrating the operation of multiple text-analysis methods" [11], within the given platform. For example, UIMA provides various ready-to-use JAVA-based methods for reading, consuming or creating annotations, but their semantics is defined by and may not be shared between different applications.

Another approach to interoperability of bio-text mining processing modules is to rely on common data exchange formats that allow for all components to communicate with other components if they comply with the defined format(s). Several groups and projects have suggested specific processing formats (e.g. [4] or SciXML⁴), but there is not yet a common initiative to provide a community-wide solution that would improve interoperability and co-operation within the community. Previous efforts to standardise generic annotation of general-language corpora (e.g. Text Encoding Initiative (TEI)⁵, Corpus Encoding System (CES)⁶) have not been widely accepted in domain-specific text mining applications, as they have been mainly focused on resources that are manually developed.

In this paper we propose a task-oriented data exchange framework that facilitates interoperability for annotation of semantic types, and provides a basis for other more complex annotations (e.g. event annotations). It includes

³ <http://www.research.ibm.com/UIMA/>

⁴ <http://www.tsujii.is.s.u-tokyo.ac.jp/jw-tmnlpo/simone.pdf>

⁵ <http://www.tei-c.org/>

⁶ <http://www.cs.vassar.edu/CES/>

¹ <http://www.tsujii.is.s.u-tokyo.ac.jp/GENIA/>

² <http://biocreative.sourceforge.net/>

the basic principles for marking up biomedical text to enable interoperability through a 'common exchange format', where various levels of mark-up are gathered within the document. We have identified the major tasks (modules) and defined a basic set of mark-up tags and attributes that enables interoperability. These are briefly discussed below (Section 2). We also propose an inline XML-based implementation (called IeXML) of the framework and illustrate it via a processing pipeline example (Section 3).

2. Towards a common annotation framework: requirements and principles

A common data exchange format should be general enough to allow all software modules to be interoperable, and – on the other hand – it has to be specific enough to gather all information provided by all modules contributing annotations to the text. Also, a common annotation schema has to account for the fact that selected modules rely on input from other text mining components (e.g. a shallow parser might expect POS information). These dependencies have to be kept in mind when building a framework for the annotation of natural language text.

It is the main aim of the framework presented here to facilitate the replacement of text processing modules that perform the same task and comply with the common exchange format. For a general text processing pipeline, this format has to serve all components in a pipeline, with minimal pre-assumptions on their dependences and ordering of their application. For example, tokenisation and separation of text into sentences are the basic text processing steps, and typically do not rely on any previous pre-processing, whereas chunk or dependency parsing as well as named-entity recognition (NER) could be based on POS annotation, but need not. Consequently, we can assume that tokenisation and sentence splitting are typically performed first, followed by POS tagging and NER in this or the reverse order, and that any further processing is completed afterwards. Further, we do not exclude that the annotations are modified at a later stage, as long as the common data format is preserved.

2.1. Basic principles for annotations provided by TM/IE modules

We have identified a set of tasks and corresponding modules that are typically used to support annotation of semantic types. These include: sentence finder, tokeniser, POS tagger and term tagger. The modularisation used here may be more fine-grained than encountered in some TM/IE systems today. This supports a precise specification of tasks as well as input/output behaviours, and should not hinder merging modules (e.g. tokenisation and POS tagging) as long as they match the requirements.

Our framework is based on a basic (minimal) set of conceptual tags and attributes that each component should provide. However, this should not preclude any module from producing other tags or attributes, nor should the tags represent any rigid meaning. As an example, we propose how a token should be annotated to allow for the reuse of tokenisers, but we do not attempt to define what a token is, so not to restrict the choice in available tokenisers. Furthermore, in addition to identifying tokens, a tokeniser may supply additional attributes, not included in the basic set of attributes.

Any IE/TM module adds or changes tags in an input document to produce a result document by generating at least a set of basic elements for that type (as defined below). We distinguish between the following basic types of annotation elements:

- **tokens** (w elements) are minimal individual constituents of text; apart from words, numbers etc., these include punctuation, references, links, inline formulas, dates, measurements, etc. They do not contain any of the other three elements, and have c attributes that specify their category (e.g. POS information).

- **terms** (e elements) comprise one or more tokens denoting concepts, objects or entities, with possibly ambiguous semantics. This includes names, terminology and nomenclature strings or sequence mutations. Attributes are used to attach semantics (sem attribute) and POS information (c attribute). They can contain token and term elements only.

- **chunks** (c elements) denote syntax units, such as noun, verb or prepositional phrases. They can contain token, term and chunk elements, and their type is assigned via a t attribute.

- **sentences** (s elements) refer to sentence elements. They can contain any of the above types.

Here, the annotation types follow a hierarchical order, whereas the order of text processing modules (tasks) is not predefined. Given the above requirements and tag types, a brief summary of the descriptions of the identified tasks is as follows:

- 1) A **sentence finder** provides s elements to an input document.

- 2) A **tokeniser** is responsible for providing w elements to an input document.

- 3) **NER and term taggers** annotate documents with e elements and specify a mandatory sem attribute that points to semantic types (see also below).

- 4) A **POS-tagger** generates c attributes for any w and e elements found in a document.

Each component can amend the annotations existing in documents as long as the resulting annotation is in line with the relevant principles (e.g. a POS tagger may either merge multiple tokens and assign a joint POS tag (c attribute), or split a token into multiple units and assign POS tags to each of them). Also, a component may gather various alternatives using an *alt* (alternatives) element, which lists all alternatives as an *ali* (alternative item) element each (see also Section 3).

Furthermore, we demand that the original document should be completely recoverable from any result document. This enables client applications to display IE/TM annotations as semantic enrichment of documents.

In the following subsection we give more details about the requirements for term and POS taggers.

2.2. An annotation task example: term tagging

Although there are notable differences between terms and named entities [10], they are typically used uniformly in further processing steps and are referred here as terms. Therefore, we suggest a common element (e) to denote terms, named entities and any other domain-specific sequences (such as mutation notations) that need a semantic attribute. Similarly, we refer to ATR (automatic term recognition) and NER modules as term taggers.

If we apply the above mentioned principles to a software component that is classified as a term tagger,

then the software component has to adhere to the following requirements:

(1) Domain-specific terms are marked up as *e* elements with a mandatory attribute *sem* to specify the semantics of the term. (It is our intention to allow a yet unspecified set of possible values for the *sem* attribute.) The term tagger may assign POS-information as a *c* attribute to a term.

(2) If the term tagger is applied to a tokenised document (text containing *w* elements), each term must contain at least one token. If it is applied to an untokenised document, each term must contain tokenisable text that would result in at least one token if tokenised.

(3) The input document may already contain *e* elements. Furthermore, terms can be nested.

2.3. An annotation task example: POS tagging

In the case of a POS-tagger, the input to a POS-tagger is a tokenised document, and POS information is encoded with the attribute *c* assigned to each *w* element. This attribute is also added to every term (*e*) element. The POS-tagger may add POS information to tokens that are inside terms. Also, the POS-tagger may change POS information in tokens or terms that have already POS information assigned. The attribute value may be an empty string to signal that the POS tagger does not know which POS applies.

In general, this framework allows for modification of the order of applied text processing modules; only in the case of the POS-tagger we require that it receives a tokenised input document. This does not exclude modules that merge a tokenizer with a POS-tagger at the same time. Furthermore, our framework leaves enough flexibility to allow, for example, the introduction and specification of *c* attributes before the POS tagger is applied, or having tokens with or without POS information inside of terms, or transformation of multiple tokens into a single token ("multi-word tokens").

3. IeXML: towards an implementation of the framework

A variety of document formats are currently in use that have to be considered as possible inputs for text mining systems. Some of them do not follow an open standard initiative (e.g. commercial formats), while others are well established in both scientific and commercial domains (e.g. XML and HTML). Although XML is well suited for interoperability of electronic data and although scientific literature is increasingly available in XML data formats, there is not yet a standard exchange format defined that harmonises available document formats and that considers semantic annotation and enrichment of text that is produced automatically by real-world applications [13].

We have developed a straightforward XML-based implementation (called IeXML) of the above annotation schema for biomedical text mining (see Figure 1 for an example). In our proposal, we have used the inline annotation approach where XML elements are integrated into the text, as opposed to stand-off annotations where references to text components are kept separated from the text itself (either within or outside the document).

We have decided to start with simple inline annotations for various reasons. Firstly, according to a recent survey in the biomedical field [2], 50% of developers rely on inline annotation in their text mining

solutions, showing that there are no preferences for any of the two annotation approaches. It is well known that inline annotation suits different text encodings (e.g. ASCII, UTF-8, Unicode, etc.), whereas stand-off annotations better overcome XML's limitations to describe ambiguous and overlapping elements [4, 13]. However, a vast majority of results of basic processing steps (such as sentence boundaries, tokenisation, POS) as well as annotation of semantic types can be represented with a limited level of ambiguities. Also, most pragmatic solutions in real-world applications resolve and scarcely represent (or use) complex ambiguities if they are not resolved at an early processing stage already. Furthermore, standard software tools are widely available to handle and visualise inline annotation (e.g. XML Document Object Model, XML DOM). Such tools are currently not or rarely available for stand-off annotation, since offset referencing is not yet standardised (there are some attempts to use token-based offsetting as opposed to character-based [4]). Lastly, stand-off annotations might raise substantial conflicts in a schema if document formats defined by the publishers of biomedical journals (e.g. Medline abstracts, Pubmed Central and Biomed Central documents) already contain stand-off annotations (e.g. MeSH terms, journal information, authors, date of publication, etc).

```
. . .
<s>
<e c="n" sem="uniprot:P50144,uniprot:P50145">
  <w c="n">Cholecystokinin</w>
</e>
<w c="cnj">and</w>
<e c="n" sem="uniprot:002686,uniprot:P01350">
  <w c="n">gastrin</w>
</e>s
<w c="v:V:P">differed</w>
<w c="prep">in</w>
<w c="n">stimulating</w>
<e c="n" sem="uniprot:002686,uniprot:P01350">
  <w c="n">gastrin</w>
</e>
<e c="n" sem="GO:0046903" onto="BP">
  <w c="n">secretion</w>
</e>
<w c="prep">in</w>
<e c="n" sem="species:9986">
  <w c="n">rabbit</w>
</e>
<w c="adj">gastric</w>
<w c="n:p">glands</w>
<w c="pun">.</w>
</s>
. . .
```

Figure 1: An excerpt tagged using the IeXML schema, after applying a sentence splitter, tokeniser, POS tagger and various term taggers

We have applied the IeXML schema to the problem of the annotation and disambiguation of semantic types using a cascaded approach [13]. A set of tools from different sites (EBI and the University of Manchester) is being ported to comply with IeXML in order to assess benefits of combining various modules for annotation of semantic types. The existing modules (including separate term taggers for various semantic types (proteins, GO terms, drugs, species etc.) that are embedded in EBIMed [14]) are being ported to comply with IeXML. Our methodology is based on the idea of separating the process into clearly defined tasks applied one after another in a

pipeline (workflow), where each module operates continuously on an input *stream* and performs its function on stretches or windows of text that are usually much smaller than the whole input. In this case, inline annotations are much more suitable than the stand-off approach. Communication of information between the modules is strictly downstream and all meta-information is contained in the data stream itself in the form of inline XML mark-up. Figure 1 presents an excerpt of a tagged document.

In the case of ambiguity, we suggest using a generic *alt* element. The *alt* element should span the minimal number of elements necessary to cover all alternatives. The modules may provide weight attributes for individual alternatives. More precisely, alternatives are represented inline as follows:

```
<alt>
  <ali>first alternative</ali>
  <ali>second alternative</ali>
  <ali>...</ali>
</alt>
```

For example, alternative term structuring (nesting) can be represented as in the following example:

```
<e c="n" sem="...">leukaemic
<alt>
  <ali>T <e c="n" sem="...">cell line</e></ali>
  <ali><e c="n" sem="...">T cell</e>
line</ali>
</alt>
</e>
```

4. Conclusions

Despite the huge efforts in developing and providing tools for the community, there are no common processing formats for the biomedical domain that could lead to wider availability and interoperability. In this paper we have suggested a framework to improve combining modules and results of annotation of semantic types, and to maximise their re-use. The suggested approach is task-oriented, as it involves the identification of basic modules that perform well-defined tasks in text processing and that can act as a part of a basic text processing workflow. For each such module, basic but extendable tag sets and basic attributes that the modules need to provide are defined.

We are currently developing an inline XML implementation of the framework. Note that an equivalent implementation can be done with the stand-off approach. Since some phenomena (e.g. sentences, tokens) can be represented more naturally using inline markup, while others are better modelled through stand-off annotations, we expect a combination of the two to be best suited for the annotation of biomedical texts.

The principles presented here allow natural extension to other module types such as parsers, anaphora resolution modules, etc. We believe that this approach could be used as a basic step for a community-wide discussion on developing common exchange formats for biomedical text processing, which will improve interoperability, tool availability and reusability in the domain. Of course, after providing interoperability on the tag level, the next challenge for the community is to define the minimal common values for tags in order to enable semantic interoperability (i.e. to specify the values for the *sem*, *c*, *t* and other attributes).

Acknowledgements

The authors express special thanks to Prof. Jun'ichi Tsujii (NaCTeM/U-Tokyo), Dr. Sophia Ananiadou (NaCTeM) and Dr. Jin-Dong Kim (U-Tokyo) for the fruitful discussions on interoperability challenges and potential solutions. This work has been partially supported by the Network of Excellence "Semantic Interoperability and Data Mining in Biomedicine" (NoE 507505), and BBSRC grant "Mining Term Associations from Literature to Support Knowledge Discovery in Biology" (BB/C007360/1).

References

- Carroll, J., Evans R, Klein E: **Supporting text mining for e-Science: the challenges for Grid-enabled natural language processing.** *UK e-Science Programme All Hands Meeting 2005*
- Corpora Survey, http://compbio.uchsc.edu/corpora/Survey_Summary.htm
- Grover C, Halpin H, Klein E, Leidner JL, Potter S, Riedel S, Scrutchin S, Tobin R: **A Framework for Text Mining Services.** *UK e-Science Programme All Hands Meeting 2004*
- Grover C, Matthews M, Tobin R: **Tools to Address the Interdependence between Tokenisation and Standoff Annotation,** *Workshop on Multidimensional markup with XML (XMLNLP), EACL 2006.*
- Interoperability Focus, www.ukoln.ac.uk/interop-focus/about/leaflet.html
- Jensen LJ, Saric J, Bork P: **Literature mining for the biologist: from information retrieval to biological discovery.** *Nat Rev Genet.* 2006 Feb;7(2):119-29
- Kirsch H, Gaudan S, Rebholz-Schuhmann D: **Distributed modules for text annotation and IE applied to the biomedical domain.** *International Journal of Medical Informatics.* (doi:10.1016/j.ijmedinf.2005.06.011), 2005
- Krallinger M, Alonso-Allende Erhardt R, Valencia A: **Text-mining approaches in molecular biology and biomedicine.** *Drug Discovery Today* 10, 439-445 (2005).
- Krallinger M, Valencia A: **Text mining and information retrieval services for Molecular Biology.** *Genome Biology,* 6 (7), 224 (2005).
- Krauthammer M, Nenadic G: **Term identification in the biomedical literature.** *Journal Biomedical Informatics,* 37(6):512-26. 2004.
- Mack R et al.: **Text analytics for life science using the Unstructured Information Management Architecture.** *IBM Systems Journal, Vol 43, No 3, 2004*
- Open Biological Ontologies, <http://obo.sourceforge.net/>
- Rebholz-Schuhmann D, Kirsch H, Gaudan S, Arregui M, Nenadic G: **Annotation and Disambiguation of Semantic Types in Biomedical Text: a Cascaded Approach to Named Entity Recognition.** *Workshop on Multidimensional markup with XML (XMLNLP), EACL 2006.*
- Rebholz-Schuhmann D, Kirsch H, Arregui M, Gaudan S, Riethoven M, Stoehr P: **EBIMed – Text crunching to gather facts for proteins from Medline.** *Bioinformatics* (accepted)
- Systems Biology Markup Language (SBML), <http://sbml.org/index.psp>
- Taverna project, <http://taverna.sourceforge.net/>