A Web Service Annotation Framework for CTD Using the UIMA Concept Mapper

Andrew MacKinlay and Karin Verspoor

National ICT Australia, Victoria Research Laboratory
Department of Computing and Information Systems, The University of Melbourne
Melbourne 3010 VIC, Australia

Introduction
We develop a set of simple web service annotators to address the named entity categories of chemical, disease, gene, and action term as defined by a set of controlled vocabularies in use at the Comparative Toxicogenomics Database [1]. It has been observed that when a large target vocabulary is defined, as is the case for gene normalization tasks and at least three of the tasks we tackle here, there may not be a significant advantage for methods that perform a separate mention detection step (e.g., using a classifier derived using machine learning) prior to mapping mentions to the target vocabulary, over strict dictionary-based methods [2]. This observation results from the requirement that in this scenario, all entity mentions must eventually be associated to a controlled vocabulary term, and we expect that association to be done by matching (possibly using “fuzzy” matching) mentions to the name or synonyms of the terms in the dictionary. Hence, we have implemented dictionary-based annotators.

Dictionary methods
Our first suite of annotators is a dictionary-lookup system based on ConceptMapper\(^1\) [3], a tool for finding dictionary matches in the UIMA framework [4, 5]. At initialisation, it is supplied with a prepared dictionary of terms. It keeps this potentially large collection of strings (containing single or multiple tokens) in memory in an efficient data structure and then searches text supplied to it for token sequences which match those in the dictionary. It has the ability to store synonyms for terms and map them back to a canonical form.

It has several parameters which control how the matching occurs, making the tool quite flexible. These include parameters relating to case matching, as well as flags for whether overlapping spans should all be annotated. It has been observed that the parameters and their values can significantly influence the performance of dictionary-based matching tools and appropriate values can vary with the specific type of entity to be recognized [6]. Therefore, some parameter settings are specific to particular target dictionary of the annotator, and are noted below. Of the parameters we set in all cases, the following are the most notable:

\(^1\) http://uima.apache.org/d/uima-addons-current/ConceptMapper/ConceptMapperAnnotatorUserGuide.html
• OrderIndependentLookup = false; we only match if the order of the contained tokens is the same as those in the dictionary.
• FindAllMatches = false; we only find the longest match within a given span of text, ignoring any shorter spans contained in it.
• SearchStrategy = ContiguousMatch; we require that matched tokens be adjacent to each other.

It is also important that ConceptMapper is supplied with a high-quality dictionary, or the matches will be of poor quality. The dictionaries in all cases are derived from the appropriate provided CTD dictionaries, although the specifics of constructing them differ for each annotator, and are discussed in more detail below.

Dictionary-based Disease and Chemical Annotators
While these annotators are applied to fairly different tasks, the techniques we settled on were identical for each. We created a ConceptMapper dictionary entry for each item in the corresponding CTD data set. We also created a synonym for each synonym listed in the supplied data, but excluded any synonyms with a length of one character or less, or which occurred in a list of common English words (derived from the Snowball project\(^2\)). These heuristics for dictionary construction increased the precision over a subset of the learning corpus without harming recall. The CaseMatch flag of ConceptMapper was set to ignoreall, meaning that the dictionary match was case insensitive.

Dictionary-based Gene Annotator
Using the same parameters as above for gene annotation led to a very high number of false positives, so in this case we added more restrictions during dictionary construction and annotation to increase precision. When creating the gene dictionary for ConceptMapper, as well as applying the same filter as for chemicals and diseases, we also exclude anything which is a sequence of 5 or fewer digits, anything which is a sequence of 1 or 2 known words, and anything which has fewer than 3 characters in total. We also change the ConceptMapper CaseMatch flag to digitfold, meaning that the case is ignored for entries containing one or more digits, but we require an exact match otherwise.

Dictionary-based Action Term Annotator
The dictionary for action-term mapping was far smaller, and is also somewhat different in nature from the other annotators since derivational and inflectional variants of the source terms are far more likely to be important. We manually created a dictionary of morphological variants of the

\(^2\) [http://snowball.tartarus.org/algorithms/english/stop.txt](http://snowball.tartarus.org/algorithms/english/stop.txt)
action terms based on those in the source CTD data, giving a dictionary of 273 synonyms in total. The CaseMatch flag was again set to ignoreall.

MetaMap-based methods
We built an additional set of annotators for the disease and chemical categories, by taking advantage of the well-known MetaMap tool [7, 8]. We observed that the CTD dictionary for these two categories included MeSH (Medical Subject Headings) identifiers as an identifier or alternative identifier. Since the MetaMap tool includes the capability of recognizing MeSH terms, this seemed to be a logical strategy for recognizing these terms. We substituted our ConceptMapper-based annotators in our UIMA pipeline with the MetaMap UIMA Annotator, limited MetaMap to recognizing terms from the MeSH vocabulary, and then matched the MeSH IDs to the appropriate CTD controlled vocabulary terms. As the performance of the MetaMap annotators was inferior to the dictionary-based lookup over a subset of the learning corpus, these were not submitted for the official evaluation.

Server implementation details
We implemented each of our annotators as a service that calls a UIMA instance. Each annotator described above is associated with a UIMA pipeline. Each pipeline is instantiated once (initialisation of the ConceptMapper pipelines, in particular, is time-consuming, since all dictionary entries must be loaded into memory), and then called repeatedly from the web server as REST requests are received.

Our use of the UIMA framework meant that we only had to implement the web service connection to the underlying annotation system once; the communication between web service and a UIMA pipeline with a given type system is generic, while the internal characteristics of the UIMA pipeline itself can change. In addition, adapting to a new type system (for example the MetaMap type system rather than the types produced by ConceptMapper) is also straightforward.

Conclusions
We were able to build simple annotators for the Comparative Toxigenomics Database by focusing on the dictionaries that define the target vocabulary of the database, and by building on existing tools for dictionary term matching – one generic, ConceptMapper, for which we only had to provide the appropriate dictionary specifications and select meaningful parameter settings, and one tailored to biomedical vocabulary, MetaMap, for which we had to subselect from its output the vocabulary relevant to the CTD categories. Our servers are intended to provide a simple annotation baseline for comparison with more sophisticated named entity recognition servers.

---

References