Using Human Language Technology for Automatic Annotation and Indexing of Digital Library Content

Kalina Bontcheva, Diana Maynard, Hamish Cunningham, and Horacio Saggion

Dept of Computer Science, University of Sheffield,
211 Portobello St, Sheffield, UK S1 4DP
{kalina,diana,hamish,saggion}@dcs.shef.ac.uk

Abstract. In this paper we show how we used robust human language technology, such as our domain-independent and customisable named entity recogniser, for automatic content annotation and indexing in two digital library applications. Each of these applications posed a unique challenge: one required adapting the language processing components to the non-standard written conventions of Old English, while the other presented the challenge of indexing material in multiple modalities. This reusable technology could also form the basis for the creation of computational tools for the study of cultural heritage languages, such as Ancient Greek and Latin.

1 Introduction

As digital libraries grow in size and coverage, so does the need for automatic content annotation and indexing. Recent advances in human language technologies like named entity recognition, information extraction, and summarisation have made it possible to create automatically metadata (e.g., extract authors, titles) and document summaries, as well as annotate and index documents with information about persons, locations, dates, etc. These advances have been seen both in the quality of the results and in the robustness of the software solutions available. An increased acceptance of the importance of engineering to the successful application of HLT has led to more predictable systems that can realistically be technology providers for Digital Library systems (which have high reusability and portability requirements).

In this paper we show how we used such technologies for automatic content annotation and indexing in two digital library applications: eighteenth century court trials (OldBaileyIE) and a multilingual and multimodal collection on the Euro2000 football tournament (MUMIS). Each of these applications posed a unique challenge: the court trials required adapting the language processing components to the non-standard written conventions of Old English, while the football collection presented the challenge of indexing material in multiple modalities - video, audio, semi-structured documents, and free text (newspaper reports). In both cases, we used as a basis our robust and customisable named entity

recogniser, which comes as part of ANNIE- A Nearly New Information Extraction system, distributed with the GATE language technology architecture and development environment [1] [1, 3]. In OldBaileyIE we also used the graphical environment of GATE, which allows manual annotation verification and correction to be carried out on the processed texts.

2 Domain-Independent Named Entity Recognition

There is an increasing need for robust and general purpose tools capable of automatically annotating named entities in large volumes of text. The goal is to offer good performance without any tuning for the particular domain and/or text type and at the same time, to design the modules for easy customisation, if the user decides to do so. In response to this challenge, we developed a domain-independent Named Entity (NE) recognition system. It is capable of annotating entities such as persons, organisations (e.g., companies, government bodies), locations, dates, percents, money amounts, addresses.

The named entity recognition modules are easily customisable, because they are based on GATE's open architecture and consist of manually created sets of pattern-matching rules that can easily be extended to add new entity types, or modified for new domains. The rule patterns are based on information produced by earlier modules, which are responsible for segmenting the text into words (**tokenisation**) and sentences (**sentence splitting**), assignment of part-of-speech information to words (**POS tagging**), and annotations of specific named entity indicators (e.g., 'Ltd.', 'Mr.') (**gazetteer lists**). For example, the following rule specifies that one or more words, starting with an uppercase letter, followed by a company designator, should be annotated as an organisation.

```
Rule: OrgXKey
(
  ({Token.kind == word, Token.orth == upperInitial})+
  {Lookup.type == cdg}
) :orgName -->
  :orgName.Organization
```

The Lookup annotations are created by the gazetteer lookup module, which assigns pre-specified types to the given lists of strings (e.g., a company designator list, person title list). These can be edited within GATE's visual environment as part of the customisation process. Modifying these lists is an easy task and was successfully carried out by non-expert users.

Rule writing requires some knowledge of the JAPE pattern-matching language [2] and ANNIE annotations. The pattern-matching language is based on regular expressions over the annotations; when a sequence of annotations is

[1] GATE and ANNIE are available freely for download from `http://gate.ac.uk`. A demo of the named entity recogniser is available online at `http://gate.ac.uk/annie/index.jsp`.

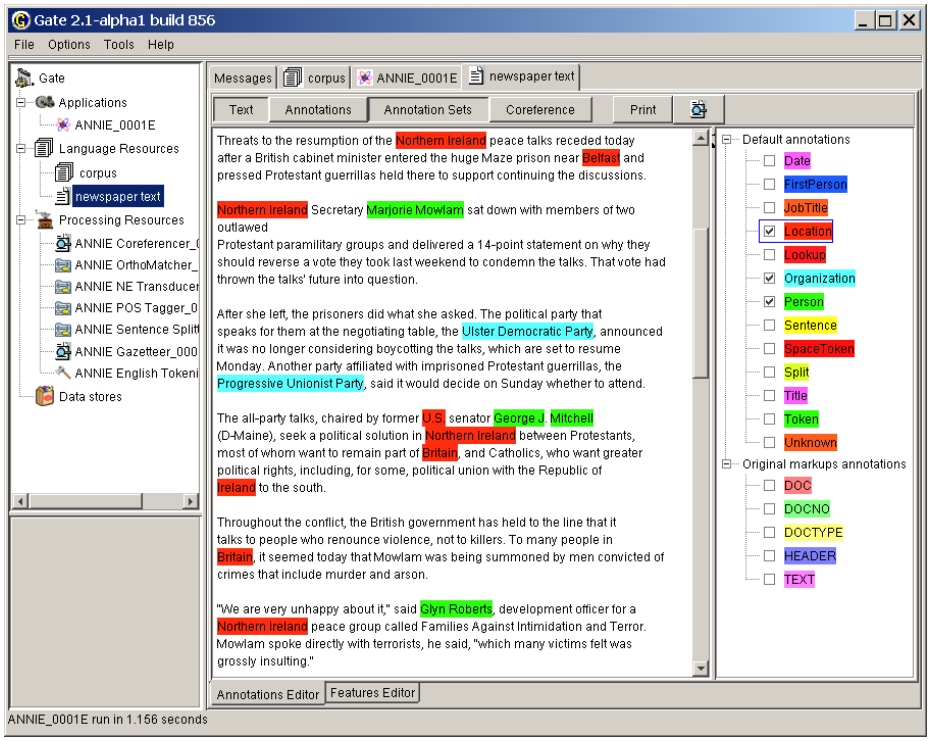

Fig. 1. Named Entities recognised by ANNIE

matched by the left-hand side pattern, then the right-hand side defines the type of annotation to be added (Organization in the example case above).

Our experiences with different systems and users indicate that writing rules to extract useful data from text is not conceptually difficult and can be learnt after some training. GATE's graphical environment allows visual inspection of the annotations and their features, which makes rule writing easier. Also, the types, attributes and values of the annotations produced by each module (e.g., tokeniser, part-of-speech tagger) are documented in GATE's User Guide [2].

In the following section we will discuss how the named-entity recogniser was easily adapted to deal with the different capitalisation and language conventions of 18*th* century English.

3 OldBailey - Semi-Automatic Annotation of a 18*th* Century English Collection

The application required the following entity types to be recognised: Person, Location, Occupation and Status.

3.1 Adapting the technology

There were two main ways in which the technology needed to be adapted. First, the application required some new entity types to be identified, and some modifications to the existing guidelines for annotation (for example, there was a new entity type "social status"). Second, the resources needed to be modified to take into account differences caused by the language used and the text type, such as different spelling, capitalisation, punctuation, and some noisy input.

First, new gazetteer lists had to be added for status information (e.g. "wife", "spinster", "Lord", "Governor"), since this entity type is not recognised in the default gazetteer.

Second, because the texts dealt with some very specific locations in London, such as "Addington Basin", "Aldermanbury Postern", "Cripplegate", additions had to be made to the lists of locations, and in particular, to the location keywords. For example, words such as "Fink", "Chain", "Ax", "Key", "Workhouse", "Rents" etc. would not normally be associated with locations, but these were commonly used in the $18th$ century (e.g. "Bennet Fink". We also needed to recognise facilities such as pubs as Locations, so new lists had to be added to take these into consideration.

As far as people's names were concerned, fewer changes needed to be made, but one particular feature of the reporting style was that many first names were abbreviated, e.g. "Benj." for "Benjamin", "Edw." for "Edward", etc. New entries had to be made in the gazetteer lists for people's first names to take this into account.

Although we already had a list of occupations, many additions had to be made to this, because there were some typical occupations of the $18th$ century which are rare these days, such as "Black-shoe-boy", "axle tree maker", "bottled porter dealer".

Finally, we had to deal with orthographic variations, since the punctuation, spelling and capitalisation were very inconsistent. For example, many proper nouns were spelt both with initial capital letters and without, and sometimes a hyphen was followed by a capital and sometimes by a lowercase letter. For example, "Bread Street" was also found in the text as "Bread-Street", "Bread-street" and "Bread street". These non-standard written conventions were dealt with both in the gazetteer lists (e.g. by adding variations of words with and without capital letters) and also in the grammar rules (for example, by looking for strings ending in particular patterns such as "-maker", "-draper", "-keeper", "-broker", with and without capital letters and hyphens.

Changing the gazetteer lists also meant making some changes to the grammar rules to accommodate these. For example, new rules had to be written for the new entity type "status", and further rules had to be written to solve conflicts with existing entity types. For example, "Baker" could be either an occupation or a person's name, so rules had to be written incorporating contextual information in order to disambiguate the entities.

Below we show an example of a typical rule added to the grammar to take account of the new gazetteer lists, which creates an annotation of type "Occu-

pation". The rule matches a pattern consisting of any kind of word (recognised by the tokeniser), followed by one of the entries in the gazetteer list for job_keys (words which typically indicate jobs, such as "-seller"). It then annotates this pattern with the entity type "Occupation", and gives it a feature "rule" with value "OccupationKey". The rule feature is simply used for debugging purposes, so it is clear which particular rule has fired to create the annotation. This rule would annotate a string such as "book-seller" or "potato-merchant".

```
Rule: OccupationKey
Priority: 50
(
 {Token.kind == word}
 {Lookup.type == job_key}
):jobtitle
-->
:jobtitle.Occupation =
   {rule = "OccupationKey"}
```

Most of the adaptation to the system lay in updating the gazetteer lists and checking for conflicts between the new rules and lists, and existing ones. This latter was mainly carried out by running the system on test texts and correcting errors where they arose. The total adaptation time for the grammars and gazetteer lists (by an experienced system developer) was approximately one person-week. Further minor changes to the system were then made by the non-expert system users from the Sheffield University's Humanities Research Institute, who are creating the digital collection.

3.2 The Annotation Correction Environment

Automatic named entity annotation systems are typically capable of capturing between 85% and 95% of the entities in the texts (although these numbers are sometimes closer to 100% for entities like dates and money amounts). Nevertheless, in certain applications, as was the case with the Old Bailey collection, it is important to have all entities annotated correctly. This process involves two tasks: deleting wrong annotations and adding new annotations for the entities that have been missed. Still, because the named entity recogniser has already annotated at least 85% of all entities, this task is much less time-consuming than fully manual annotation.

Our users from the Humanities Institute used the visual annotation environment that comes with the GATE system. This environment makes the annotation process as simple and quick as possible and yet allows the flexibility to add new annotation types.

Wrong annotations are detected by the annotator visually, by inspecting the highlighted text strings; the GATE environment uses different colours for the different types of annotations (see Figure 2). To delete a wrong annotation, the user right-clicks on the highlighted text and selects the type of annotation

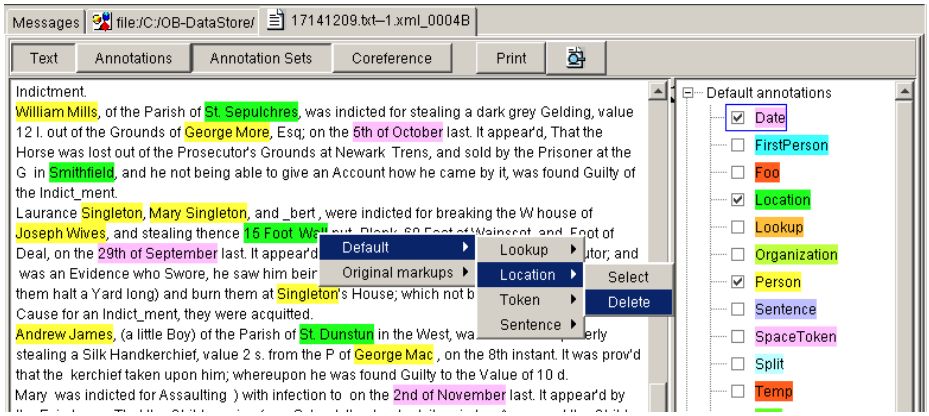

Fig. 2. An Old Bailey example text from 1714

they want deleted (there could be more than one annotations of different kinds associated with the same string). For example, in Figure 2, the system wrongly recognised '15 Foot Wall' as a location, because of a space in the digitised text, which changed the original phrase '15 Foot Wallnut Plank' to become '15 Foot Wall nut Plank'[2]. GATE also offers a facility to delete all occurences of wrong annotations, when they refer to the same entity. For example, if the House has wrongly been identified as a location, all occurrences of this entity in the text can be viewed and deleted by pressing the Delete key, when the entity is selected in the Coreference data panel on the right (see Figure 3). In this way, the user removes all occurrences with one action, rather than having to delete all these annotations individually.

New annotations are added by selecting the text with the mouse (e.g., "William Mills") and then clicking on the desired annotation type (e.g., Person), which is shown in the list of types on the right-hand-side (see Figure 2).

The annotation environment comes as standard with a number of pre-defined annotation types (e.g., Person, Organization). New types can be defined and made available in the annotation environment by specifying them in the XML Schema language supported by W3C. For example, the schema for defining an Occupation annotation type looks like:

```
// Occupation schema
<?xml version="1.0"?>
<schema
xmlns="http://www.w3.org/2000/10/XMLSchema">
 <!-- XSchema deffinition for Occupation-->
```

[2] In 18*th* century English nouns were frequently capitalised. This example also demonstrates that the digitised content contained noisy data, such as underscores, spaces, etc.

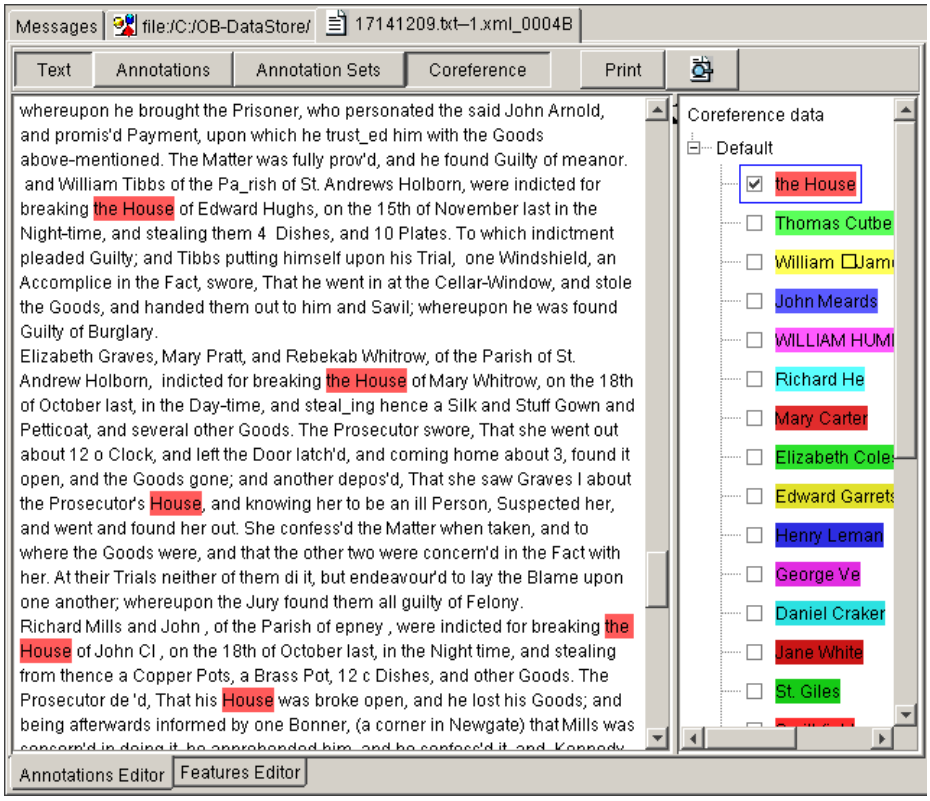

Fig. 3. Viewing and deleting all annotations for an entity

```
  <element name="Occupation">
 </element>
</schema>
```

The GATE visual annotation environment can also be used independently from the language processing tools. It offers the advantage of making the annotation process independent from the particular document format used, e.g., XML. Our experience with users from the humanities has shown that they find the annotation task easier when it involves selecting text and associating types in a visual environment, than when it requires writing XML markup. The colour coding scheme which associates different types of annotations with different colours makes it easier for users to find and correct the annotations. Once the annotation process is completed, the user only needs to save the document, which automatically inserts the newly added markup in the appropriate format (e.g., HTML, XML).

4 MUMIS - Indexing and Search in Multiple Media

The MUMIS (MUltiMedia Indexing and Searching environment) system uses Information Extraction (IE) components developed within GATE to produce formal annotations about essential events in football video programme material.[3] The textual sources used for this project are taken from reports of the Euro2000 Championships: semi-structured ticker reports that give a minute by minute objective account of the match; match reports that also give a full account of the match but may be subjective; and comments that give general information such as player profiles. These reports are drawn from a variety of online media sources (BBC-online, Press Association, The Guardian, etc.). The video material was indexed by running the IE system on the output of an automatic speech recognition system. Information about the same event in multiple sources is merged into a common index, which also contains references to the video material, thus allowing direct access to the relevant part of the video.

The indexed multimedia information is accessed via a menu-based user interface (in Dutch, English, and German) (see Figure 4) which allows professional users to query the indexed information and play video fragments matching the query (e.g., "all fouls on Beckham").

The domain-independent named entity recogniser was extended to recognise domain-specific entities such as teams, players, and referee. Our philosophy of clean separation between linguistic data and the algorithms that process it resulted in a substantial saving of effort, since only new domain-specific data (i.e., names and grammar rules) had to be created, while the algorithms using them remained unchanged. Further details on the adaptation process and on the use of information extraction for multimedia indexing and search can found in [4].

[3] `http://mumis.vda.nl/`, funded by the EC's 5th Framework HLT programme under grant number IST-1999-10651. Project partners: Universities of Twente, Nijmegen and Sheffield, Esteam AB, VDA Ltd., DFKI.

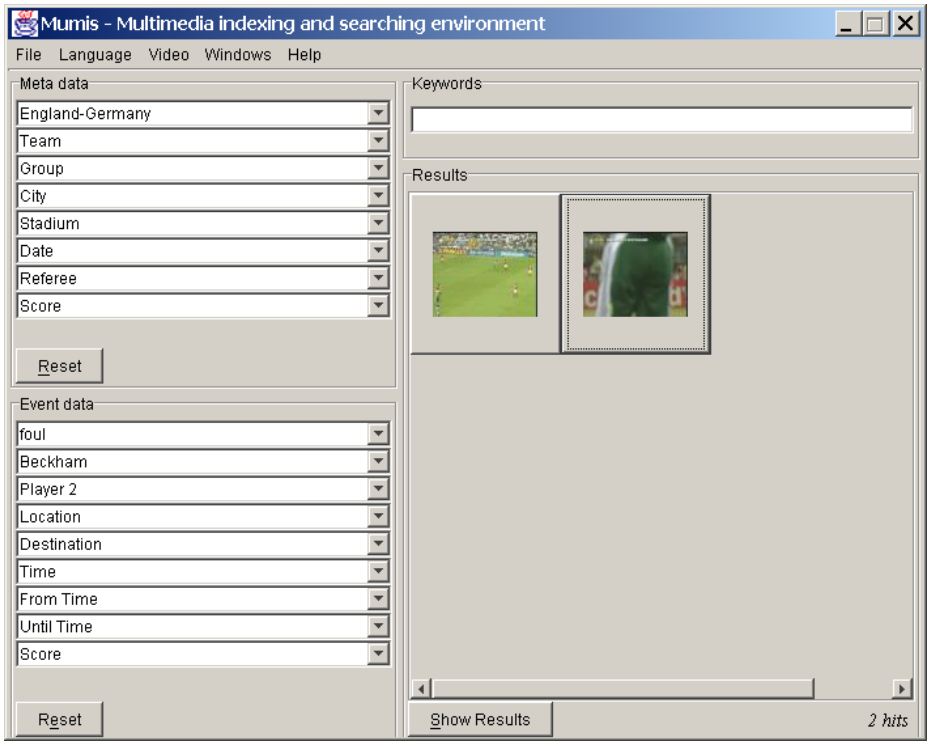

Fig. 4. The MUMIS interface

5 Conclusion

In this paper we presented briefly the domain-independent portable named entity recogniser developed as part of the ANNIE Information Extraction system. ANNIE is intended to be useable in many different applications, on many different kinds of text and for many different purposes. Here we showed how it can be applied for indexing and annotating digital library content. The system's portability offers a comprehensive support for documents in many different formats, from badly-spelled lower case email messages to structured XML or HTML pages to newswires. It is capable of dealing with noisy data and processing of large data volumes. Finally, ANNIE is also capable of processing and visualising multilingual documents, based on Unicode [5], which makes it particularly suitable for digital library applications which need to deal with content in languages other than English. For example, it could be a basis for creation of computational tools for the study of cultural heritage languages, such as Ancient Greek and Latin.

References

1. H. Cunningham. GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, 36:223–254, 2002.
2. H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. *The GATE User Guide*. `http://gate.ac.uk/`, 2002.
3. Diana Maynard, Valentin Tablan, Hamish Cunningham, Cristian Ursu, Horacio Saggion, Kalina Bontcheva, and Yorick Wilks. Architectural elements of language engineering robustness. *Journal of Natural Language Engineering – Special Issue on Robust Methods in Analysis of Natural Language Data*, 2002. forthcoming.
4. H. Saggion, H. Cunningham, D. Maynard, K. Bontcheva, O. Hamza, C. Ursu, and Y. Wilks. Extracting Information for Automatic Indexing of Multimedia Material. In *3rd International Conference on Language Resources and Evaluation (LREC 2002)*, page xxx, Las Palmas, Gran Canaria, Spain, 2002.
5. V. Tablan, C. Ursu, K. Bontcheva, H. Cunningham, D. Maynard, O. Hamza, Tony McEnery, Paul Baker, and Mark Leisher. A unicode-based environment for creation and use of language resources. In *Proceedings of 3rd Language Resources and Evaluation Conference*, 2002. forthcoming.