

Nested Named Entity Recognition in Historical Archive Text

Kate Byrne
School of Informatics
University of Edinburgh
k.byrne@ed.ac.uk

Abstract

This paper describes work on Named Entity Recognition (NER), in preparation for Relation Extraction (RE), on data from a historical archive organisation. As is often the case in the cultural heritage domain, the source text includes a high percentage of specialist terminology, and is of very variable quality in terms of grammaticality and completeness. The NER and RE tasks were carried out using a specially annotated corpus, and are themselves preliminary steps in a larger project whose aim is to transform discovered relations into a graph structure that can be queried using standard tools. Experimental results from the NER task are described, with emphasis on dealing with nested entities using a multi-word token method. The overall objective is to improve access by non-specialist users to a valuable cultural resource.

1. Introduction

Cultural heritage archives, which are typically publicly funded, are under considerable pressure nowadays to provide access to their textual material to a diverse audience, via the Internet. Governments everywhere have a social inclusion agenda, on which shared cultural experience rightly features prominently. Data collections that were put together over decades or even centuries, with highly variable recording standards and often with a specialist or scholarly audience in mind, are now being adapted hastily for users with low levels of subject knowledge but high expectations. It is a difficult task, but one in which NLP techniques are potentially valuable.

The first step of organising the textual material into a computerised database has already been completed by most heritage bodies. In very many cases the resulting databases are a mixture of sparsely populated fixed fields and associated free text. These databases are becoming available on the Web and finding effective mechanisms for querying them is a priority. One common solution is to limit

drastically the range of possible queries, making the interface simple but restricting query flexibility. Another approach that is gaining popularity involves treating the entire database as a text corpus over which “Google-style” string matching queries are allowed. This discards the structural information provided by the database framework and often gives the user too much freedom: without some guidance it is difficult for a non-specialist to frame a sensible query.

This paper describes work within a project whose aim is to combine the best of both these worlds, by retaining all available structural information and making query guidance possible, whilst transforming the data into a graph format that can be queried by simple string patterns, using standard tools. The graph data is held as RDF triples¹ and can be queried using SPARQL² or a similar graph query language.

The aim is to transform the archive’s free text material into such a graph structure (that can then be combined with a graph derived from the structured database fields). The approach taken is to extract binary relations between pairs of Named Entities in the text. Once found, the relations can readily be converted to RDF triples. This paper describes work on the first step, of Named Entity Recognition (NER), concentrating especially on dealing with nested entities.

A sample of text data from the RCAHMS³ historical archive database was annotated with entities and relations, as is described in Section 2. That section also explains some of the characteristics peculiar to the dataset. The handling of nested NEs (where one entity string contains others within it) is important, and Section 3 considers related work, looking particularly at approaches to nested entity discovery. A number of experiments have been carried out and the setup and results are described in Sections 4 and 5 respectively. Finally, Section 6 provides some discussion of the issues raised.

¹<http://www.w3.org/RDF/>

²<http://www.w3.org/TR/rdf-sparql-query/>

³The Royal Commission on the Ancient and Historical Monuments of Scotland, <http://www.rcahms.gov.uk/>.

2 Data and Annotation

The complete database from which the annotated corpus was drawn contains around 250,000 site-related records, with 750,000 records for associated archive items such as photographs, plans, drawings etc. Each of the site records has a text document attached. These vary greatly in length, from a couple of lines to several pages. The corpus for this NER and Relation Extraction (RE) work consists of 1,546 of these text documents, selected at random. Between them they contain 9,768 “sentences”, though only a minority — estimated at around 30% — are grammatical English sentences. The text is written in notes form, with a lot of bibliographic references and spatial location co-ordinates scattered throughout it. The corpus has been annotated for NEs and relations as described below.

Informal examination of the output suggests that Inter Annotator Agreement (IAA) figures⁴ may be relatively low, particularly for the relation annotation, but this has yet to be measured properly.

2.1 Named Entities

The NE annotation was designed around 11 classes: ORG, PERSNAME, ROLE, SITETYPE, ARTEFACT, PLACE, SITENAME, ADDRESS, PERIOD, DATE and EVENT. Three of these, SITETYPE, ARTEFACT and EVENT, are further divided into subclasses, but no experiments have yet been done with subclasses.

An earlier version of the same corpus had used fewer classes, but the classification scheme was changed because of considerations connected with the ultimate goal of producing a data querying application for non-specialists with a general interest in, say, local history. Locational terms were divided into PLACE (cities, towns, etc.), SITENAME (specific historical sites) and ADDRESS, and a distinction was made between SITENAMES (e.g. “Hill of Caldback”) and SITETYPES (“chambered round cairn”). The ADDRESS class is primarily intended to prevent its two more important siblings, PLACE and SITENAME, from being cluttered with locational terms that carry useful information but are too specific to be common query terms (such as “2, The High Street” or “HP 6079 0669”). The distinction between these different location classes is somewhat arbitrary at times. For example, “Hill of Caldback” is a SITENAME because it happens to be an archaeological site, but somewhere with no particular historical significance, like “Blackford Hill”, would be a PLACE. A location like “Braes of Doune” would be a PLACE, but “Liberton Brae”

⁴IAA is a measure of the degree to which independent annotation, by different human judges, coincides. If the figure is low it suggests that the correct classification of terms and relations is difficult to judge, and the performance of a machine learner is likely to be correspondingly low.

would be an ADDRESS in most contexts (as it is the name of a road in the city of Edinburgh). In the same way but of less significance because the classes are much smaller, there is overlap between DATE and PERIOD. The former is primarily for calendar dates, but may include expressions like “1870 – 1880”; whilst PERIOD would include terms like “late 19th Century”. It is a moot point which class fits “the 1870s” better, and the decision would be made in context.

On the face of it, a larger number of categories makes the NER stage harder, but it may assist in the RE stage. Some of the relations can be restricted to particular domain and range classes, so greater specificity amongst these classes helps limit the choices for relation categorisation. However, this argument is strongest where the classes are most orthogonal anyway; being able to distinguish, say, ORGs from PERSNAMEs and from locational terms is important for the RE stage and is also comparatively easy at the NER stage. The difference between the PLACE, ADDRESS and SITENAME classes is harder for the NE recogniser, but is also less important for relation classification.

2.2 Relations

Although not directly relevant to the NER task this paper is concerned with, the relation annotation is briefly described as background. The NE classes and relation predicates are closely linked and were designed to mesh with one another.

With one exception, the defined relations are binary ones of the form *predicate(subject, object)*. They are: *isA*, *sameAs*, *seeAlso*, *partOf*, *hasLocation* and *hasPeriod*. The exception is *eventRel*, which has higher arity: *eventRel(eventType, eventPatient, eventDate, eventAgent, eventRole, eventResult)*. This will subsequently be converted into binary relations that can be expressed in RDF. The *sameAs* relation is for co-reference, and takes the form of a collection of two-place relations between the members of a co-referential set.

The relation predicates were chosen for maximum generality within the corpus domain. There are a great many more specific categories that could have been included (e.g. *architectOf*, *excavatedBy*, and relationships between individual people) but the resulting class sets would have been too sparsely populated for successful classification.

The set of relation types includes the three (*isA*, *seeAlso* and *sameAs*) that are the core relations typically used in ontologies and thesauri for this domain. In the standard thesauri they are referred to respectively as *Hierarchical*, *Associative* and *Equivalence* relationships [3, 7, 10]. Thus, the extraction of relations can be viewed as an ontology building exercise in this domain.

2.3 Nested Entities

NER is generally treated as a classification task, where a sequence of tokens is tagged with labels by the classifier. Where entities overlap or are nested within one another, classification becomes difficult, because an individual token may require more than one label. In the corpus used here, up to three levels of nesting can occur. For example, in the string

```
[[[Edinburgh]PLACE University]ORG Library]ORG
```

the token “Edinburgh” is a PLACE entity on its own, and part of two distinct ORG entities.

Commonly, in NER tasks, only a single level of nesting is dealt with — generally the longest string, or outermost entity. However, the subject and object of each relation will be a Named Entity, and if nested entities are omitted then relations using them will also be lost.

For example, in the kind of text we are dealing with, the nesting of a PLACE entity within a longer entity string is quite common (as in [[Aberdeen] School of Architecture], [Earl of [Argyll]], and so forth). Although the resulting *hasLocation* relations will probably be important in query applications, they are likely to be missed unless we can deal with nested NEs. Similarly, bibliographic references (which are classed as EVENTS with subclass DESCRIPTION) typically contain PERSNAME and DATE entities which may participate in separate relations. A user who is interested in historical sites mentioned in a given bibliographic work (such as a paper by a particular archaeologist), is very likely also to be interested in sites associated with the author of that work (which will probably date from the same period, or be similar in some other way). Thus the discovery of *all* entities, regardless of level, is important. Various methods documented in the literature are examined below, and Section 4 describes experiments using a novel approach to the problem.

3 Related Work

Interest in nested NE detection has increased in recent years, though it is still the case that most NER work deals with only one level at a time. One way of dealing with nested entities [12] is to detect one level (the innermost in this case) and then derive rules with which to find other NEs containing these as substrings. The dataset used was the GENIA corpus [2]. The authors report an improvement of around 3% in the F-score under certain conditions.

A different approach [6] uses structured multilabel classification to deal with overlapping and discontinuous entities. (The corpus of MEDLINE abstracts used did not contain nested entities.) In multilabel classification, each ex-

ample is associated with a set of labels instead of just one. Here, the labels are structured in the sense that they do not come from a pre-defined set but are built for the instance in hand. Theoretically, the number of different labels is exponential on the length of the instance, but the set for consideration can be limited using the structure of the labels. The method was successful when compared with standard sequential tagging, such as is used by the CandC classifier [4] employed in this work.

In another study [5], the problem is cast as a binary classification task, using a one-*vs*-rest scheme, to get round the difficulty of individual tokens requiring more than one label if they are part of a nested entity. In this work just two entity classes (proteins and DNA in the GENIA corpus) were used, in separate experimental runs, with two levels of nesting: outermost and any one inner. The study found that their “outmost labeling” method recognised outermost entities better, and “inner labeling” was better for inner NEs (as one might expect).

The idea of using “joined label tagging” [1], in which the number of entity classes is expanded to include concatenations of overlapping class labels, is discussed in Section 4 below. It is contrasted with the method proposed here, which concatenates tokens instead, so that each nested entity string has its own separate label. Alex et al. also use cascading and layering methods in a pipeline combining taggers, to achieve good results for nested entity discovery.

4 Experimental Setup

The 1,546 text documents comprising the corpus were tokenised, split into sentences and POS tagged, before being formatted for annotation using the MMAX2 annotation tool.⁵ The data was then reformatted for the CandC maximum entropy classifier [4], using the BIO notation. The beginning of an entity string is given a “B-” prefix before its label, tokens within the entity string have an “I-” prefix, and tokens that are not entities are labelled “O”. Thus, a phrase like “...in the National Monuments Record” becomes “in_O the_O National_B-ORG Monuments_I-ORG Record_I-ORG”. Tagging was done using 10-fold cross-validation.

The distribution of NE types is summarised in Table 1. In total, the annotated corpus contains 28,272 entity strings, if all levels of nesting are included, and all lengths of entity string. For reasons explained below, NE strings consisting of seven or more tokens were excluded, bringing the total down to 27,453. The proportion of NEs having other entities nested within them is 9.4%, whilst 18.7% are nested within longer NEs. Each containing entity typically has either one, two or three shorter NEs within, with two being

⁵<http://www.eml-research.de/english/research/nlp/download/mmax.php>

the commonest. The corpus contains no entities with more than three levels of nesting, i.e. at most we have outer, inner and innermost levels. No disjoint entities (where the tokens comprising the NE string are non-adjacent) were included.

As discussed above, the problem with nested entities is that individual tokens require multiple labels if they participate in more than one entity string. One possibility is to concatenate labels; for the “Edinburgh University Library” example cited above, the first token might be labelled B-PLACE_B-ORG_B-ORG, the second O_I-ORG_I-ORG and the third O_O-I-ORG. This makes the task more difficult because the number of categories to choose from is larger, while the number of training instances remains the same, but it enables all levels of entity to be recognised. This joined label tagging technique has been successfully used in the biomedical domain [1] and that work may be extended in future projects to cover the RCAHMS dataset, which would enable a comparison between joined label tagging and the technique used here.

The alternative tried here is to concatenate the tokens instead, so that each entity string becomes a single token and can be given its own correct label. To achieve this, a maximum entity string length must be determined in advance, and then every token in the corpus is concatenated with those following, up to the chosen length. For example, if the maximum entity length to search for were 3, then the phrase “when Edinburgh University Library was built” would be tokenised and labelled as follows:

```
when O
when.Edinburgh O
when.Edinburgh.University O
Edinburgh PLACE
Edinburgh.University ORG
Edinburgh.University.Library ORG
University O
University.Library O
University.Library.was O
Library O
Library.was O
Library.was.built O
... and so on.
```

An analysis of the distribution of entity string lengths showed that 97.10% were of length 6 tokens or fewer, though the longest was 25 tokens. The maximum length figure was therefore set to 6 for this experiment, which excluded 819 NEs. This is consistent with the overall goal of the project, because long strings are very unlikely as user query terms. On the face of it, the advantage of this method is that the number of categories is unchanged, while the amount of training data is increased — though the big increase is in the number of negative examples. The obvious drawback is the increased time taken for training the classifier, and also that (depending on how the tokens are

Entity Type	Raw Count	≥ 7 tokens	Kept
SITETYPE	5,675	7	5,668
ADDRESS	3,558	100	3,458
EVENT	3,843	667	3,176
DATE	3,520	1	3,519
ORG	2,737	7	2,730
SITENAME	2,737	25	2,712
PLACE	2,509	6	2,503
PERSNAME	2,318	0	2,318
ARTEFACT	879	0	879
PERIOD	406	6	400
ROLE	90	0	90
Total:	28,272	819	27,453

Table 1. NE types distribution

arranged) the sentence length is increased 6-fold.

The training data was presented to the classifier in the format shown above, with features added. For each individual token a set of 6 “multi-tokens” was generated, with one token in the first, two in the second, and so on. The test data was in exactly the same form, without the tag labels. The classifier output a single tag per token received: from the set of 11 class labels plus “O”.

The final token of the concatenated string was made salient to the classifier by passing it the POS tag of the last token in the string. The final token was picked each time for consistency and because it is most likely to be the head word of any entity string. The following unigram features were also experimented with, but produced only a marginal improvement in overall performance (see Table 2, run 10):

- position within set of 6 multi-word tokens: p1 to p6
- contains “_”: yUnd or nUnd
- capital following “_”: ic0 (none), ic1 (some) or ic2 (all)
- last token type: letters converted to “A” or “a”, digits to “0”, punctuation unchanged (so “Shetland” becomes “Aaaaaaaa”).

The CandC classifier is, naturally enough, optimised for standard sentences. It has a number of built-in features based on the bigram and trigram context in which each token appears, and it also makes use of gazetteers of personal names and place-names. The multi-word tokenisation may well confuse the classifier and, to explore the multi-word method thoroughly, a classifier would have to be configured with it in mind. Therefore some experiments were run with “previous word” and “previous POS tag” bigram and trigram features, where “previous” refers to the preceding

Run	Description	Precision %	Recall %	F-score %	Correct NEs
Single-token: average 24,448 NEs (varies because of random selection)					
1	Single token, all lengths, random-1	70.47	69.73	70.10	16,923
2	Single token, all lengths, random-2	71.05	69.75	70.39	16,918
3	Single token, outermost NEs	74.77	72.42	73.58	16,671
4	Single token, max len 6, random-1	74.64	72.67	73.64	17,931
5	Single token, max len 6, random-2	74.57	72.65	73.60	17,920
6	Single token, outermost, max len 6	77.06	75.09	76.06	18,359
7	As run 6 + domain gazetteers	76.98	75.18	76.07	18,379
Multi-token: 27,453 NEs available					
8	Multi-token, basic	80.75	65.24	72.17	17,899
9	Multi-token, domain gazetteers	80.52	65.67	72.34	18,015
10	Multi-token, unigram features	82.14	66.79	73.67	18,322
11	Multi-token, all POS and word trigrams	81.84	66.26	73.23	18,178
12	Multi-token, w_{i-1} alone	87.70	66.79	75.83	18,322
13	As run 12 with weights adjusted	84.79	70.81	77.17	19,426
14	As run 12 with weights adjusted	82.96	73.39	77.32	19,860
15	As run 12 with weights adjusted	78.43	75.91	77.15	20,825

Table 2. Summary of NER results

single token from the original corpus text, and not the immediately preceding multi-word token. Similarly, forward bigrams and trigrams were also tried. These features were chosen as they are known to be particularly useful to a standard NE classifier, but there is scope for further exploration.

The experiments were run almost exclusively with the CandC experimental NE tagger, which is a state of the art system, tuned for NER over English text. Since it was realised that the multi-token method would be penalised by not in fact being very like normal English text, a simple experiment was done with another, general purpose tagger, Zhang Le’s Maximum Entropy Toolkit, or ZLMaxent [11]. The aim was not to tune this tagger properly for NER, but merely to compare the single and multi-token methods on a level playing field, where neither had any advantage.

5 Results

Results so far indicate that the multi-word tokenisation technique can improve the tagger’s performance, when combined with the extra word and POS features mentioned above.

Table 2 summarises the overall NER results, comparing a number of runs of the standard single-token method with multi-word tokenisation, and showing Precision, Recall and F-score (harmonic mean) figures, calculated using the CONLL scorer. For the single token runs, only one level of entities is available each time. The table shows what the total number of available entities was for each experiment, and the scores are percentages against this total. The scores for the multi-token runs are percentages of the full set of

27,453 NEs, as this method is capable of training on, and outputting, all the NEs at once.

The first run included all NEs, selecting a single one at random wherever there was a nested set. Run 2 is exactly the same, with a different random selection. It was unclear whether this would be a fairer baseline than using only the outermost entities, as is commonly done, so run 3 was included. The significant difference in performance was unexpected: consistently using outer entities produced a 3% improvement in the overall F-score, whereas one might expect such entities to be much harder to recognise, as they are sparser in the corpus. Runs 4 and 5 (which are the same except for using a different random selection in each family of nested entities) excluded the longer entities — those consisting of strings of more than 6 tokens — in order to match the multi-word method, which also excludes them. The results are not significantly different from the previous run, which supports a conclusion that the classifier is sensitive to the length of the entity strings, but that it is a mixture of lengths (as in runs 1 and 2) that causes problems, rather than their absolute length. To test this, run 6 used only outermost NEs (as in run 3) and excluded the long ones. This produced another significant performance gain, to a F-score of 76.06%. This suggests the possibility of improving the tagger’s model by training it separately on entities of each different string length.

The final run of the single-token set used extra gazetteers, in addition to those built-in to the classifier (for personal names and place names). The main ingredient was the Thesaurus of Monument Types (TMT), which is based on MIDAS [7]. Terms from the Thesaurus of Object Types,

NE class	Run 7 (<i>best single</i>)			Run 12 (<i>unsmoothed multi</i>)			Run 15 (<i>smoothed multi</i>)		
	P %	R %	F %	P %	R %	F %	P %	R %	F %
ADDRESS	79.99	82.03	81.00	82.32	83.42	82.87	70.97	85.50	77.56
ARTEFACT	53.41	32.49	40.40	71.14	16.29	26.51	56.62	29.73	38.98
DATE	86.38	92.04	89.12	95.09	82.01	88.06	88.22	90.68	89.43
EVENT	85.24	73.43	78.89	94.81	64.47	76.75	87.74	76.26	81.60
ORG	91.23	90.98	91.11	99.27	89.22	93.97	98.30	91.24	94.64
PERIOD	64.59	56.61	60.34	83.26	44.75	58.21	72.29	63.25	67.47
PERSNAME	83.49	84.69	84.08	96.86	77.13	85.87	91.26	85.15	88.10
PLACE	83.34	78.15	80.66	94.88	66.69	78.33	91.17	69.77	79.05
ROLE	93.10	60.67	73.47	98.00	54.44	70.00	96.15	55.56	70.42
SITENAME	62.53	71.04	66.51	65.98	62.60	64.24	53.80	69.46	60.63
SITETYPE	67.32	64.88	66.08	82.17	45.07	58.21	71.52	63.70	67.38
	76.98	75.18	76.07	87.70	66.79	75.83	78.43	75.91	77.15

Table 3. Detailed NER results

maintained by MDA (which formerly stood for “Museum Documentation Association”) and available from the same source as TMT, were also added. These were intended to help the SITETYPE and ARTEFACT classes respectively. This produced no improvement, which is not altogether surprising, as work in the NER field has shown that gazetteers tend not to enhance performance significantly if the training data is sufficiently representative [8]. The main reason for including them here was in order to test whether having multi-word terms helped. With a multi-word tokenisation it was clear that longer strings could be included in the gazetteer list, by concatenating them (with underscore characters) exactly as the corpus tokens were concatenated. This is a simpler approach than those needed to handle multi-word gazetteer entries in a standard single token setup.

Runs 8 to 15 used the multi-word tokenisation. The first, the simplest for this approach, used no additional features and no gazetteers apart from the classifier’s own. It performed worse overall than the single-word method (though on a larger number of NERs), but it seems likely there was a mixture of positive and negative effects (see below). Adding the domain gazetteers (run 9) had no impact, answering the question raised about multi-word gazetteer terms. This is not especially unexpected: gazetteers tend only to include terms the classifier sees plenty of examples of anyway, and has little trouble with. It may be possible to improve the contribution made by gazetteers, by training a separate model for them, as recent work has shown [9].

The unigram features discussed in Section 4 were tested in run 10, and produced a small improvement. It might be worth further experimentation to find better features characterising the multi-word token in a useful way.

A series of runs was made, of which run 11 and run 12 are shown as being the most noteworthy, using various combinations of previous and next POS tags and words:

$pos_{i-2}, w_{i-2}, pos_{i-1}, w_{i-1}, pos_{i+1}, w_{i+1}, pos_{i+2}, w_{i+2}$.

This was an attempt to reproduce some of the normal features built-in to an NE classifier. Surprisingly, w_{i-1} (previous word) was not only by far the most useful (run 12), but was much better on its own than when combined with the others (run 11 used a combination of all the POS and word features just listed).

It is noticeable that precision is improved at the expense of recall in the multi-word experiments. For this domain, that is arguably a benefit, as is discussed below. However, in order to try to balance precision and recall better, a series of trials was carried out with varying weights for each class applied to the maximum entropy model that CandC uses. This adds constraints that the model must fit, and biases it towards or against selecting a particular NE class tag. Runs 13, 14 and 15 are examples of slightly different weightings that each improved recall without losing too much precision, and improved the overall F-score. In run 13 the weighting favours SITETYPE and EVENT classes, which are large and important classes having poor recall (see Table 3). In Run 14 the weighting is uniform across all the NE classes but biased against the “O” class, to improve recall across the whole set by having a less cautious model. Run 15 is similar to 14 but with an even more smoothing, and almost balances precision and recall. The same trials were made for the single word models where, as expected (since these are already well balanced), no significant improvement in overall score was possible.

In some respects the multi-word tokenisation must surely have a negative impact. As already mentioned, the tagger has built-in bigram and trigram features that will be skewed in these experiments. Also, the tagger makes use of prior knowledge of word frequencies derived from large English text corpora (1 billion words), and at least 5/6^{ths} of the tokens will appear as unknown words. Therefore the fact

Gold Standard	Classifier Error	Comment
J I McKinley PERSONAME Dental School SITENAME National Library of Scotland SITENAME St Abb's Head PLACE London Mint O three of which SITETYPE dyke SITETYPE the enclosure SITETYPE East Cults Parish Church SITENAME	I McKinley PERSONAME Dental School ADDRESS same 4 tokens, tagged as ORG St Abb's Head SITENAME London Mint ADDRESS tagged as O this dyke SITETYPE enclosure SITETYPE Cults Parish Church SITENAME Parish Church SITETYPE	<i>partial string tagged wrong class, but ok for querying as above as above classifier choice seems better co-reference not handled well determiner wrongly included determiner wrongly omitted 2 errors, but ok for querying</i>

Table 4. Examples of NER errors

that a slight improvement in overall performance is possible suggests that there must be a definite positive effect balancing the negative.

As is usual, the results for precision, recall and F-score are given as percentages. This is slightly misleading, as the total number of entity strings is higher in the multi-word experiments than in the single baseline where only one of each nested entity set is available. For runs 1 to 7 the average number of NEs that could be found is 24,448. (The total varies for each of the random runs, from 23,020 to 24,675, because a long entity string may contain several shorter ones.) For runs 8 to 14 it is the total NE population: 27,453. Thus the multi-token method can output an average of over 12% more NEs than the single-token method. The actual number of correctly predicted NEs for each run is included in the last column of Table 2. If one were to compare the results on the total NE population (i.e. deeming the single-token model to have missed all the NEs unavailable to it), then the best single-token run (number 7) would have recall of only 66.95%.

Table 3 gives the detailed results for precision, recall and F-score within each NE class, for runs 7, 12 and 15. Numbers 7 and 12 are the best from each method with standard constraints on the maximum entropy model, and 15 is the same as 12 but with smoothing applied to balance precision and recall as closely as possible.

As noted, precision is much improved by the multi-word technique, whilst recall suffers. The ARTEFACT class performs poorly in all variants, but particularly with the multi-word tokens. It is one of the sparser classes and the members do not seem to be sufficiently distinctive to be easy to model. The scores for ROLE are probably not very reliable, as this class is tiny (see Table 1). With smoothing, the multi-word model produces a slightly better F-score than the single-token method (77.15–77.32 compared to 76.07), whilst outputting considerably more NEs: 20,825 as against 18,379 — or 13.3% more.

Analysis of the errors made by the multi-token classifier

Run	Precision	Recall	F-score	Correct NEs
7	83.54	81.82	82.67	18,906
12	93.07	71.32	80.76	18,986
15	85.14	82.46	83.78	21,952

Table 5. “Unlabelled” results for Table 3 runs

is interesting. In calculating the error percentages, all deviations from the gold standard are counted as errors; but in practice some are more significant than others. Table 4 lists some examples of common classification errors that, from the point of view of building a graph of relationships between entities, would not be very harmful. Unfortunately, it is difficult to calculate what percentage of the errors are like these, as opposed to more damaging failures. Including or excluding extraneous tokens (such as a preceding determiner) is a pity, but much less serious than missing an entity altogether. Similarly, it is more important to detect the presence of an entity, or “content carrying term” than to classify it correctly. Table 5 compares “unlabelled” results corresponding to Table 3’s, where the scoring is just on presence or absence of an NE as detected by those models (without retraining). It might be interesting, given more time, to retrain the models for different NE class sets, perhaps merging the locational classes (PLACE, ADDRESS, SITENAME) together, and the time-based ones (PERIOD, DATE), that were noted in Section 2.1 as likely to be hard to distinguish.

Finally, as was mentioned at the end of Section 4, a simple experiment was performed using the ZLMaxent tagger, which is a general purpose maximum entropy classifier, with no NE tuning. It treats the data simply as a collection of instances with attached features, rather than as sentences made up of sequences of tokens. For this kind of classifier, the multi-word tokens are at no disadvantage, since there is no background knowledge of English word frequencies and so forth. A comparison was made using the baseline set of

multi-word tokens (all 27,453 NEs that are 6 tokens long or shorter) against the corresponding single-word set (as used for run 6 in Table 2, having 24,500 NEs), with the same minimal set of features in each case. The results are shown in Table 6. The scores are very low, as is to be expected for a completely untuned system, but those for the multi-word tokens are a good deal higher, and the bias towards precision is strongly confirmed.

	P %	R %	F %
single-word tokens	41.06	48.56	44.49
multi-word tokens	78.59	46.90	58.75

Table 6. Comparison using ZLMaxent

6 Summary and discussion

This paper has presented a novel method for nested entity recognition, that appears to be promising, though further exploration is needed to test the limits of performance that could be achieved. The drawback is the longer time needed to train the classifier, but the significant advantage of the multi-word system is that the classifier can output *all* the NEs in the corpus, instead of only one of each family of nested entities. The depth of nesting is immaterial. In the experiments described, 13.3% more NEs were correctly found by the multi-word model as compared with the baseline single token method.

Future work will involve extracting binary relations between entities detected. Finding all levels of nested entities is important because, if an entity is missing, any relation using it as either source or target will automatically be lost also.

The maximum length of entity string to search for must be determined in advance — in this case a maximum length of 6 was chosen, which excludes only 2.9% of entities in this cultural archive corpus. Arguably, excluding long entities is a sensible tactic anyway for this project, which aims to build a queryable graph from text relations, so entity strings can be considered as candidate query terms. There is likely to be a performance gain from dropping the longer entity strings, and they are improbable as end-user query terms.

The method increases precision at the expense of recall. For a system ultimately intended to deal with queries over cultural data by non-specialists being able to achieve high precision is good news. This type of user is generally not greatly concerned with recall — he or she does not need to see *every* example of a long barrow in Scotland — but precision is crucial. It would be a bad mistake to tell such a user that a long barrow exists where it does not. Naturally,

specialists in the fields of archaeology or architectural history (the topics this data covers) may be interested in good recall as well, but this programme of work is specifically directed towards assisting non-experts, for whom trustworthy information is much more important than complete coverage.

References

- [1] B. Alex, B. Haddow, and C. Grover. Recognising nested named entities in biomedical text. In *Proceedings of BioNLP 2007 (to appear)*, Prague, Czech Republic, June 2007.
- [2] N. Collier, H. S. Park, N. Ogata, Y. Tateisi, C. Nobata, T. Ohta, T. Sekimizu, H. Imai, K. Ibushi, and J. ichi Tsujii. The GENIA Project: Corpus-based Knowledge Acquisition and Information Extraction from Genome Research Papers. In *Proceedings of EACL'99*, pages 271–272, 1999.
- [3] N. Crofts, M. Doerr, and T. Gill. The CIDOC conceptual reference model: A standard for communicating cultural contents. *Cultivate Interactive*, (Issue 9), February 2003.
- [4] J. Curran and S. Clark. Maximum entropy tagging for named entity recognition. Informatics research report, University of Edinburgh, School of Informatics, ICCS, December 2003.
- [5] B. Gu. Recognizing Nested Named Entities in GENIA corpus. In *Proceedings of the BioNLP Workshop on Linking Natural Language Processing and Biology at HLT-NAACL 06*, pages 112–113, New York City, June 2006. Association for Computational Linguistics.
- [6] R. McDonald, K. Crammer, and F. Pereira. Flexible text segmentation with structured multilabel classification. In *Proceedings of EMNLP05*, 2005.
- [7] MIDAS. *MIDAS: A Manual and Data Standard for Monument Inventories*. English Heritage, Data Standards Unit, National Monuments Record Centre, Swindon, 3rd reprint edition, 2003. ISBN: 1-873592-33-7.
- [8] A. Mikheev, M. Moens, and C. Grover. Named Entity recognition without gazetteers. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL'99)*, pages 1–8, June 1999.
- [9] A. Smith and M. Osborne. Using gazetteers in discriminative information extraction. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 133–140, New York City, June 2006. Association for Computational Linguistics.
- [10] D. Tudhope and C. Binding. A Case Study of a Faceted Approach to Knowledge Organisation and Retrieval in the Cultural Heritage Sector. *Digicult – Thematic Issues*, (6: Resource Discovery Technologies for the Heritage Sector):28–33, June 2004.
- [11] L. Zhang and T. Yao. Filtering Junk Mail with a Maximum Entropy Model. In *Proceedings of 20th International Conference on Computer Processing of Oriental Languages (IC-CPOL03)*, pages 446–453, 2003.
- [12] G. Zhou, J. Zhang, J. Su, D. Shen, and C. Tan. Recognizing Names in Biomedical Texts: a Machine Learning Approach. *Bioinformatics*, 20(7):1178–1190, 2004.