A Supervised Machine Learning Approach to Conjunction Disambiguation in Named Entities

Paweł Mazur[*]
[*]Institute of Applied Informatics
Wrocław University of Technology
Wyb. Wyspiańskiego 27,
50-370 Wrocław, Poland
Pawel.Mazur@pwr.wroc.pl

Robert Dale[†]
[*,†]Centre for Language Technology
Macquarie University,
NSW 2109, Sydney, Australia
[†]rdale@ics.mq.edu.au
[*]mpawel@ics.mq.edu.au

Abstract

Although the literature contains reports of very high accuracy figures for the recognition of named entities in text, there are still some named entity phenomena that remain problematic for existing text processing systems. One of these is the ambiguity of conjunctions in candidate named entity strings, an all-too-prevalent problem in corporate and legal documents. In this paper, we distinguish four uses of the conjunction in these strings, and explore the use of a supervised machine learning approach to conjunction disambiguation trained on a very limited set of 'name internal' features that avoids the need for expensive lexical or semantic resources. We achieve 84% correctly classified examples using k-fold evaluation on a data set of 600 instances. We argue that further improvements are likely to require the use of wider domain knowledge and name external features.

1 Introduction

Named entity recognition consists of identifying strings in a text that correspond to named entities, and then classifying each such named entity string as being of a specific type, with typical categories being Company, Person and Location. The range of named entity categories to be identified is usually application dependent.

Introduced for the first time as a separately evaluated task at the Sixth Message Understanding Conference in 1995 (see, for example [Grishman and Sundheim, 1995; 1996]), named entity recognition has attracted a considerable amount of research effort. Initially handled with hand crafted rules (as, for example, in many of the participating systems in MUC-6 and MUC-7) and later by means of statistical approaches (see [Sang, 2002; Sang and Meulder, 2003]), the state-of-the-art provides high performance for named entity identification and classification both for specific domains and for language- and domain-independent systems.

However, our experience with existing software tells us that there are still some categories of named entities that remain problematic. In particular, and much to our surprise, we are aware of almost no work that has explored the disambiguation of conjunctions appearing in named entity strings. Resources such as an appropriate domain lexicon or relevant semantic knowledge might allow a system to emulate a human's ability to determine that a string like *Seshasayee Paper and Boards Limited* is a single company name; but in the absence of such resources, the string could just as easily be interpreted as two separate names. Determining the correct interpretation is clearly important for any application which relies on named entity extraction. We are interested in how such interpretations can be arrived at relatively cheaply, and particularly without recourse to expensive-to-construct resources, so as to allow for rapid development in new domains.

The significance of this kind of ambiguity depends, of course, on the extent to which the phenomenon of conjunctions in named entities is widespread. Our current work focuses on a corpus of 13000 company announcements released through the Australian Stock Exchange: these are documents provided by companies in order to meet both continuous and periodic disclosure requirements, in which we want to track mentions of companies and individuals across time.

From this corpus, we selected 45 documents at random; in these documents, there were a total of 545 candidate named entity strings, of which 31 contained conjunctions. This informal sampling suggests that conjunctions appear, on average, in around 5.7% of candidate named entity strings; however, in some documents in our sample, the frequency is as high as 23%. For comparison, a check on the MUC-7 evaluation data shows that, in that corpus, the proportion of candidate named entity strings containing conjunctions is 4.5%. The documents in our corpus have some features that are not necessarily typical for other corpora. In particular, texts in this domain frequently have some of the characteristics of legal documents, where many sometimes apparently arbitrary elements are given initial capitals. Therefore, we might expect some specific domains, such as those dealing with accountancy and law, to have a higher density of names involving conjunctions. These frequencies are sufficient to suggest that the seeking of an appropriate means of handling conjunctions is a worthwhile and important pursuit.

2 Problem Description

An examination of the candidate named entity strings appearing in our corpus reveals four distinct uses of the conjunction, as exemplified in the following examples:

1. Oil and Gas Ltd
2. Agfa and Fuji
3. John and Mary Smith
4. Company Secretary Resignation and Appointment

In example (1), we have a single named entity that happens to contain an internal conjunction; in example (2), we have a conjunction of two distinct named entities; and in examples (3) and (4), we have conjunctions that, from a linguistic perspective, contain a form of ellipsis, so that one conjunct is incomplete on its own, but can be completed using information provided in the other conjunct.

The fact that conjunctions in named entities are problematic has been noted before, in particular by Mikheev et al. [1998], who suggested the strategy of examining the preceding document context to identify candidate conjuncts that should be considered as separate named entities. Mikheev et al. mention this approach being part of their system used in the MUC-7 competition, but no data is reported on the accuracy of this kind of heuristic; in our experience, there are many cases where there are no antecedent mentions that can be used in this way. Furthermore, in the MUC-7 data, strings like *John and Mary Smith* were considered as one named entity, whereas, for many information extraction applications, it is important to recognize that this string represents two distinct entities. This has the consequence that the MUC-7 data cannot be used for evaluation of our approach without further annotation.

In more recent work of relevance, we would point to the novel approach to segmentation described in [McDonald *et al.*, 2005]. Using multilabel classification, it is possible to tag overlapping and non-contiguous segments. However, to our knowledge there are no available results to indicate how well this approach would work for the conjunction disambiguation problem. Other work [Solorio, 2004] has used the presence of a conjunction as a feature in machine-learning-based NER, but it is unclear what benefits were gained by introducing this feature.

More generally, of course, the processing of conjunctions has been a focus of interest in linguistics; in particular, Categorial Grammar (see, for example, [Steedman, 1985]) provides a sophisticated treatment of the syntax of conjunctions. Linguistic analyses tend to focus on conjunctions involving common nouns and adjectives rather than proper names, but as is clear from the examples above, it is not so easy to draw a clear line between these two categories.

In the present work, our interest is in seeing how far we can go without recourse to more sophisticated linguistic treatments, particularly since these are likely to provide finer-grained analyses than are required for our purposes.

Following from the above, we distinguish four categories of candidate named entity strings containing conjunctions.

Name Internal Conjunction (NI): This category covers those cases where the candidate named entity string contains one named entity, where the conjunction is part of the name. Some examples from our corpus: *Publishing and Broadcasting Limited*, *J B Were & Son*, *Hancock and Gore*, *Acceptance and Transfer Form*, and *Fixing and Planning Phase*.

Name External Conjunction (NE): This category covers those cases where the conjunction serves to separate two distinct named entities. Some examples from our corpus: *Italy and Central Europe*, *Hardware & Operating Systems*, *Mr Danny Fisher and Mr Don Wilson*, and *American Express and Visa International*.

Right-Copy Separator (RC): This category of conjunction separates two named entities, where the first is incomplete in itself but can be completed by copying information from the right-hand conjunct. This is perhaps most common in conjunctions of proper names, as in *John and Mary Smith*, but appears in other contexts as well. Some examples from our corpus: *State and Federal Government*, *Eastern and Western Australia*, and *General & Miscellaneous Equipment*.

Left-Copy Separator (LC): This is similar to the previous category, but instead of copying information from the right-hand conjunct, in order to complete the constituent named entities we need to copy information from the left conjunct. Examples in our corpus: *Gas Supply and Demand*, *Financial Statements and Reports*, *Hospital Equipment & Systems*, *J H Blair Company Secretary & Corporate Counsel*.

Conceptually, we might view the last two categories as subtypes of the more general category **Copying Separator**; however, it makes sense to keep the two categories separate, since the process of reconstructing the unelided conjuncts is different in each case.

3 Our Approach

Our approach to determining the proper conjunction category in a candidate named entity string is to use a machine-learned classifier. We are particularly interested in seeing how far we can address the task using only limited knowledge sources: in the work described here, we restrict ourselves to very limited gazetteers that contain the most frequent proper nouns that appear in our corpus, and to the use of so-called 'name-internal' properties (i.e., characteristics of the candidate string itself, rather than of its surrounding context). Using only limited gazetteers maximises portability; considering only name internal properties will make it easier to see the impact of subsequently adding contextual information. Perhaps more importantly with regard to the specific data we are dealing with, we find many candidate strings appearing in typographic contexts such as tables where the relevant local context can be hard to determine, if it exists at all; in such cases, all we can rely on are the name-internal features.

In the remainder of this paper, we first describe in detail the data used in our experiments (Section 4.1), the name-internal text features used as attributes for classification (Section 4.2), and the data encoding used to encode the features into a feature vector (Section 4.3). Then, in Section 5, we discuss how we determined a baseline for our experiments, and describe the machine learning algorithms we used. Section 6 provides a discussion of the evaluation scheme we adopted, and an overview of the results achieved in the experiments. Section 7 presents details of what went wrong by analysing misclassified examples from our data set. Finally, in Section 8, we

present a discussion of possible directions in which the approach described here could be further developed. A particular approach to extending a standard named entity recognizer with the conjunction disambiguation functionality implemented as a separate module is presented in [Mazur and Dale, 2006].

4 Experimental Setup

4.1 Corpus and Data Preparation

The focus of our project is a data set from the Australian Stock Exchange (ASX). This data set consists of a large number of company announcements: for a variety of regulatory reasons, listed companies provide around 100000 documents to the ASX each year, and the ASX subsequently makes these available to users via the web. For more information about the documents in this corpus, and a discussion of our general approach to processing them, see [Dale *et al.*, 2004].

The corpus used for our research consisted of a 13460 document sub-corpus drawn from a larger corpus of company announcements from the ASX. The documents range in length from 8 to 1000 lines of text.

Evaluation data was prepared as follows. For our purposes, we define a candidate named entity string to be any sequence of words with initial capitals and one embedded conjunction. We also allowed these strings to contain the lowercased preposition *of* and the determiners *a*, *an*, and *the*. Candidate named entity strings from sentences written completely in uppercase or with every word being initcapped (i.e., strings in 'title case') were ignored. Using a Perl script, we extracted 10925 candidate named entity string instances from our corpus, corresponding to 6437 unique forms. From the set of unique forms, we randomly selected 600 examples for our test data set. In a small number of cases, problems arising from typographic features such as ASCII formatted tables caused us to manually correct some individual strings. An example of the need for such correction is demonstrated by the candidate extracted string *Name of Entity Hancock & Gore Limited*, where it turns out that *Name of Entity* is a label in a list, and *Hancock & Gore Limited*, being a company name, is the value of that label; however, in our data, the text extraction process has caused the separating formatting to be lost, resulting in the two strings being concatenated. In this case we remove *Name of Entity* from the string extracted by our Perl script, on the assumption that a smarter text extraction technique would be able to interpret the layout more accurately.

The resulting set of strings was then annotated using a set of small gazetteers listing common person names, company names, locations and other named entity elements that are frequent in our corpus and related to our tagset, which is described in the next section.[1]

The categories of the conjunctions in the candidate named entity strings were assigned by a human annotator. Table 1 presents the distribution of evaluation instances across the four conjunction categories introduced above.

[1]This is part of our strategy for fast deployment in a new domain, where a seed lexicon is constructed from the most frequent words that contain initial capitals.

NI	NE	RC	LC	Sum
185	350	39	26	600
30.8%	58.3%	6.5%	4.3%	100%

Table 1: Example distributions in categories.

4.2 The Tag Set

We developed a 16-tag tag set, presented in Table 2, to annotate the tokens in our corpus of candidate named entity strings. Most of the tags, such as Loc, Org, GivenName, AlphaNum, Dir, and PersDesig, are the same as those used by many other named entity recognizers; some, however, are specific to our needs. The Son tag is used to annotate tokens whose surface form is either *Son* or *Sons*: these occur relatively often in company names (as, for example, in *A Davies & Sons Pty Ltd*), and are a strong indicator of the Name Internal Conjunction category. The Of and Det tags are used to mark the preposition *of* and the determiners *the*, *a* and *an*, irrespective of casing. Finally, InitCapped is used to annotate any tokens that do not belong to the other categories, or which are ambiguous between those categories.

No	Tag	Meaning
1	Loc	The name of a location
2	Org	The name of an organization
3	GivenName	A person's given name
4	FamilyName	A person's family name
5	Initial	An initial in the range A-Z
6	CompPos	A position within a company
7	Abbrev	Abbreviation
8	PersDesig	A person designator
9	CompDesig	A company designator
10	Son	*Son(s)*
11	Dir	A compass direction
12	AlphaNum	An alphanumeric expression
13	Month	The name of a month
14	Of	Preposition *of*
15	Det	Determiners *the*, *a*, *an*
16	InitCapped	Unrecognized initcapped token

Table 2: The tagset used for text annotation.

We also recognize multi-word elements where there is no ambiguity (for example, in the case of unambiguous person, location and company names). For example, although the company name *Australia and New Zealand Banking Group Limited* is not in our gazetteer, *New Zealand* as a country name is, and so this string is recognized as a sequence of tokens whose types are marked as Loc and Loc Org CompDesig; here the second Loc tag corresponds to the pair of tokens *New Zealand*.

We refer to the sequence of tags assigned to a particular string as a **pattern**. A pattern also indicates the conjunction type present in the string, as determined through the human annotation; so, for the example above, the complete pattern is ⟨Loc and Loc Org CompDesig, Internal⟩.

Table 3 presents the number of tags of each type used to annotate our data set; in total there were 2190 tags assigned over the 600 candidate named entity strings, for an average of 3.65 tags per instance.

Tag	Occurrences	Percentage
InitCapped	925	42.24
Loc	245	11.19
Org	175	7.99
FamilyName	164	7.49
CompDesig	138	6.30
Initial	108	4.93
CompPos	99	4.52
GivenName	89	4.06
Of	76	3.47
Abbrev	73	3.33
PersDesig	39	1.78
Det	31	1.42
Dir	12	0.55
Son	7	0.32
Month	6	0.27
AlphaNum	3	0.14

Table 3: The popularity of tags in annotated data.

Notably, a significant number of the tokens are tagged as simply being of type InitCapped; this is in keeping with our deliberate use of small gazetteers, and is likely to be the case in any domain where new names are constantly being introduced.

4.3 Encoding

For the purposes of machine learning, we encode each pattern in the following way. We create an attribute for each of the 16 tag types for each of the left and right sides of a conjunction, for a total of 32 attributes. The attributes are of integer type with values $\{0, 1\}$, thus signaling either the presence or absence of a token of that type anywhere within either conjunct. We also introduce an addition binary attribute, ConjForm, for encoding the lexical form of a conjunction in the string: 0 denotes *&*; 1 denotes *and*.

With each data instance there is associated a categorical ConjType attribute with the values {Internal, External, Right-Copy, Left-Copy}; this is used to encode the actual category of the conjunction in the string.

5 The Algorithms

5.1 Baseline

It is quite common to determine a baseline using the 0-R algorithm, which simple predicts the majority class [Witten and Frank, 2005]. On our data set, with this approach we get a baseline accuracy of 58.33%. However, we have found that with the 1-R algorithm, described in [Holte, 1993], we obtain a better-performing model based simply on the lexical form of the conjunction:

```
IF ConjForm='&' THEN PredCat←Internal
IF ConjForm='and' THEN PredCat←External.
```

This very simple rule provides a baseline of 69.83%.

5.2 Classifiers

The experiments were conducted using the WEKA toolkit [Witten and Frank, 2005]. This provides implementations of several machine learning algorithms, along with the data structures and code needed to perform data input and output, data filtering, and the evaluation and presentation of results.

After some initial exploration using a variety of algorithms for supervised machine learning available in WEKA, we chose the following: the Multilayer Perceptron, two lazy algorithms (IBk and K*), and three tree algorithms: Random Tree, Logistic Model Trees and J4.8. We also include here the results for Naïve Bayes and SMO given the popularity of these methods in the field. We provide here only short descriptions of these algorithms, and point to appropriate references for further information:

LMT (Logistic Model Trees) are decision trees that contain logistic regression functions at the leaves [Landwehr *et al.*, 2005].

J4.8 is an implementation of the decision tree algorithm C4.5 revision 8, the last public version of the C4.5 decision tree learner [Quinlan, 1993].

Random Tree is an algorithm for constructing a decision tree that considers K random features at each node (we use different values for K); it performs no pruning.

IBk is a K-nearest neighbours classifier (K=1); see [Aha *et al.*, 1991].

K* is an instance-based classifier: i.e, the class of a test instance is based upon the class of those training instances similar to it, as determined by some similarity function. It differs from other instance-based learners in that it uses an entropy-based distance function; see [Cleary and Trigg, 1995].

Multilayer Perceptron is a multilayer neural network using back propagation for training [Rojas, 1996].

Naïve Bayes is a specialized form of Bayesian network with two assumptions: (1) predictive attributes are conditionally independent given the class; (2) no hidden or latent attributes influence the prediction process [John and Langley, 1995].

SMO (Sequential Minimal Optimization) is an algorithm for the fast training of Support Vector Machines [Platt, 1999].

6 Results

6.1 Evaluation Scheme

For evaluation, we used the k-fold method with $k = 10$, so that our data set of 600 examples was divided into ten folds

by random selection of instances from the original data set. Then, for each of the folds, the classification models were built on the remaining 540 examples and tested on the held-out fold. The sum of correctly classified examples for all folds is the final result. There are of course some side effects of this evaluation approach, which we mention in Section 7; however, it still makes more sense to use this approach for our small data set of 600 examples, than artificially dividing this set into even smaller training and test data sets.

Algorithm	Correctly classified (out of 600)
IBk	84.00% (504)
Random Tree	83.83% (503)
K*	83.50% (501)
Mult. Perc.	82.17% (493)
LMT	81.17% (487)
J4.8	79.50% (477)
SMO	78.00% (468)
Naïve Bayes	70.67% (424)
Baseline	69.83% (419)

Table 4: Results for k-fold evaluation.

6.2 Classification Results

Table 4 presents the results achieved in the experiments. All algorithms scored above the baseline, though Naïve Bayes, with the worst result, was very close to the baseline.

Category	Precision	Recall	F-Measure
Name Internal	0.814	0.876	0.844
Name External	0.872	0.897	0.885
Right-Copy	0.615	0.410	0.492
Left-Copy	0.800	0.462	0.585
weighted mean	0.834	0.840	0.833

Table 5: Detailed accuracy by category of a conjunction for results of IBk classifier.

NI	NE	RC	LC	→ classified as ↓
162	28	6	3	NI
18	314	17	11	NE
4	6	16	0	RC
1	2	0	12	LC

Table 6: Confusion matrix for IBk.

The best classifier turned out to be IBk, the K-nearest neighbours algorithm. The precision, recall and F-measure for this case are presented in Table 5. Table 6 provides a confusion matrix with the desired and actual classification of examples. The best results are for Name Internal and Name External conjunctions. The low results for Right- and Left-Copy Separator conjunction types are mainly because of low recall for these categories: 0.410 and 0.462, respectively. This is most likely caused by the fact that there are very few examples of these categories: 6.5% and 4.3%, respectively (see Table 1).

We used the χ^2 test for equality of distributions and a significance level of 90% to check whether the difference between the result of IBk and other algorithms is statistically significant; on this basis, we find that only the difference between the IBk algorithm and the Random Tree algorithm is no greater than chance.

It is interesting to note that the relatively simple Random Tree algorithm scored so highly. We tried different values for its parameter K, the number of randomly chosen attributes to be considered at each node. The result presented in the table is for $K = 22$; for the default $K = 1$, the algorithm correctly classified 490 examples.

7 Analysis

7.1 Conjunction Category Indicators

A statistical analysis of the data reveals some strong conjunction category indicators.

For the Name External these are:

- a Month tag in the left conjunct (as in *September and December*);
- a Comp-Desig or Abbrev tag in the left conjunct (as in *Alliance Technology Pty Ltd and Suco International* or *NLD and BRL Hardy*); but there are exceptions: *JP Morgan Investment Management Australia Ltd and Associates*, *Association of Mining & Exploration Companies* and *ASX Settlement and Transfer Corporation*, which are all Name Internal;
- a Month or PersDesig tag in the right hand conjunct (as in *February and March* or *Mr R L Hanwright & Mrs M J Hanwright*; and
- a GivenName, Dir or Abbrev tag in the right hand conjunct, although there are exceptions: *Beaches and Quay West Brisbane* and *SMDS and ATM WANS* (both are of the Right-Copy Separator type).

The presence of a Son tag is a strong indicator of a Name Internal conjunction.

7.2 Error Analysis

Our experiment demonstrates that with supervised machine learning over a simple set of features, we can reach a classification error rate of 16–18%. In this section, we provide some discussion of the classification errors made by the best-performing learner, the IBk algorithm.

InitCapped

Of the 96 misclassified examples, 38 (39.58%) consist of a pattern consisting entirely of InitCapped tags. In such cases, classification ends up being determined on the basis of the

ConjForm attribute: if the value is *&*, then the conjunction is classified as being Name Internal, and if its value is *and*, the conjunction is classified as being Name External. Consequently, the following examples are misclassified: *Victorian Casino and Gaming Authority*, *Coal Handling and Preparation Plan*, *Gas Supply and Demand Study*, and *Explanatory Memorandum & Proxy Form*.

At the same time, there were 96 InitCapped-only patterns that were classified correctly; this means that out of all 134 InitCapped-only patterns 71.64% were classified correctly, which is quite consistent with the previously-discussed baseline.

There were also another 11 misclassified instances consisting mainly of InitCapped tags along with some other tags; examples of these are: *Australian Labor Party and Independent Members* ⟨Loc InitCapped Org and InitCapped InitCapped⟩, *Association of Mining & Exploration Companies* ⟨CompDesig Of InitCapped & InitCapped InitCapped⟩ and Securities and Exchange Commission ⟨InitCapped and InitCapped Org⟩.

Long Patterns

Two misclassified instances were represented by relatively long patterns: for example, *Fellow of the Australian Institute of Geoscientists and The Australasian Institute of Mining*, represented by the 12-tag pattern ⟨CompPos Of Det Loc Org Of InitCapped and Det Loc Org Of InitCapped⟩.

Other Interesting Cases

There were two cases of misclassified strings with patterns containing patterns of other, rather common, examples. The additional tag in the extended pattern turned out to be insufficient information for a classifier to classify the extended pattern properly. As a result the examples with extended patterns were still classified like examples with shorter patterns. One example is the string *WD & HO Wills Holdings Limited*, being the name of a company (so the conjunction is of Name-Internal type) and having the pattern ⟨Initial Initial & Initial Initial FamilyName CompDesig⟩, was incorrectly classified as containing a Right-Copy Separator conjunction. This is because the conjunction is Right-Copy in the common pattern ⟨Initial Initial & Initial Initial FamilyName⟩.

A similar case is the string *Vancouver and Toronto Stock Exchanges*, which has the pattern ⟨Loc and Loc Org⟩; since the pattern ⟨Loc and Loc⟩ has category Name External, the model classifies this example in the same way, effectively ignoring the Org tag, whose presence might be taken as an indicator of the Right-Copy conjunction type. This is a case where only a slightly more sophisticated linguistic analysis might have helped: since *Exchanges* is plural, this might serve to indicate a Right-Copy conjunction.

The string *Wayne Jones and Topsfield Pty Ltd*, which in reality involves a Name External conjunction, was classified as Name Internal. We would note here that, in the absence of additional contextual information, conjunctions of person names and company names are often ambiguous even for humans.

Another related highly ambiguous type of example corresponds to the pattern ⟨FamilyName and FamilyName⟩, which can either be a conjunction of two person names or just one company name.

We also note here the impact of the k-fold evaluation approach. Since a new model is built for each fold, it turns out that the IBk classifier assigned category Name Internal to instances of the pattern ⟨InitCapped and InitCapped Org⟩ in one case, but assigned Right-Copy in another case. Consequently, both *Federal and State Government* (Right-Copy), being in one fold, and *Securities and Exchange Commission* (Name Internal), being in another fold, were misclassified.

Other Observations

There are also some cases which we expected to be handled easily, but which turned out to be problematic. For example, *D J Carmichael Pty Limited and Kirke Securities Ltd* was classified as Name Internal, although it contains company designators in both conjuncts and the form of conjunction is *and*. Similarly, the string *Department of Transport and Department of Main Roads* (with the pattern ⟨Org Of InitCapped and Org Of InitCapped InitCapped, External⟩) was classified as Name Internal.

Finally, there is a group of around 15–20 examples for which it is difficult to provide a clear explanation for misclassification along the lines of the cases above; in these cases, the major issue is the classifier's ability to generalize the rules (which is not necessarily due to a deficiency in the algorithm, but perhaps due to the simple tagset we use).

8 Conclusions and Future Work

We have presented the problem of conjunction disambiguation in named entities and defined four categories of conjunction in candidate named entity strings. We defined the problem as one of classification and showed that it can be handled well using supervised machine learning algorithms and a limited set of name-internal features.

Given the similarity in results for most of the different machine-learned classifiers we used, we conclude that a significant improvement of results lies in a richer feature selection rather than in choice of the classifier. This conclusion is also supported by the fact that some examples are difficult for a human to classify without wider context or domain knowledge.

A number of issues arise in the work reported here as candidates for future work.

We have restricted ourselves to candidate named entity strings which contain a single conjunction; however, there are of course cases where multiple conjunctions appear. One category consists of examples like *Audited Balance Sheet and Profit and Loss Account*, where again the kinds of syntactic ambiguity involved would suggest a more syntactically-driven approach would be worth consideration. Another category consists of candidate named entity strings that contain commas as well as lexicalised conjunctions.

A rudimentary analysis of frequently occurring n-grams in our corpus makes it clear that some strings containing conjunctions appear frequently. For example, in our corpus there are 296 occurrences of the string *Quarter Activities and Cash-*

flow Report,[2] making it the most frequent 5-gram. Moreover, there are another 34 occurrences of this string with the conjunction *&* in place of *and*, and another six strings with the variant spelling *Cash Flow*. In any real application context, it would make sense to filter out these common cases via table lookup before applying a machine learning process to classify the remaining conjunctions. This kind of preprocessing could identify frequent strings containing either Name Internal or Name External conjunctions. Another form of preprocessing could involve the analysis of abbreviations: for example, in the string *ASX Settlement and Transfer Corporation (ASTC)*, the abbreviation *ASTC* could be used to decide that the conjunction in the preceding string has the category Name Internal.

More generally, there are three directions in which we might move in order to further improve performance.

First, we can always use larger gazetteers to reduce the number of tokens that can only be tagged as InitCapped. This, of course, has a cost consequence; in current work, we are exploring how performance on this task improves as larger numbers of frequent name elements from the corpus are incorporated into the gazetteers. Another consequence of extending gazetteers is the problem of the same token being in two or more gazetteers, for example Location and FamilyName. A naive approach would be to assign these tokens the catch-all InitCapped tag, but since this is what we want to avoid, we could also assign all the ambiguous tags and indicate this fact in the feature vector. This would of course require a redesign of the feature vector.

Second, we can make more sophisticated use of the name internal properties of the candidate string. This includes, as noted above with regard to the *Exchanges* example, taking account of the syntactic number of the constituent tokens. Armed with a part of speech tagger, we could also attempt heuristic chunking of the candidate strings which might assist in determining conjunction type; and a resource like WordNet might be used to identify terms with shared superordinates, as in the *Paper and Board* example mentioned in Section 1.

Third, we can extend the learning process to take account of contextual features. As noted earlier, there are cases where the local context cannot be easily determined, but in many cases local syntactic information such as the number of an associated verb can serve to distinguish the type of conjunction being used. However, as demonstrated here, it is already possible to achieve a high level of accuracy without recourse to name external features; as we noted earlier, this is important in our domain, where names often appear in tables, making local context unavailable.

There are also aspects of our evaluation process which are open to improvement. First, we aim to increase the size of the data set used to see if the present results are robust; and second, the techniques here need to be evaluated in the context of a complete named entity recognition process, so that we can determine the overall diference in performance when operating with and without a conjunction disambiguation capability.

[2] This appears frequently as a substring of longer expressions like *First Quarter Activities and Cashflow Report*, *Second Quarter Activities and Cashflow Report*, and so on.

9 Acknowledgements

The work reported here was carried out while the first author was a visiting scholar at the Centre for Language Technology at Macquarie University. The second author acknowledges the support of the Capital Markets Cooperative Research Centre in carrying out this work.

References

[Aha *et al.*, 1991] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Mach. Learn.*, 6(1):37–66, 1991.

[Cleary and Trigg, 1995] John G. Cleary and Leonard E. Trigg. K*: An Instance-based Learner Using an Entropic Distance Measure. In *Proceedings of the 12th International Conference on Machine Learning*, pages 108–114. Morgan Kaufmann, 1995.

[Dale *et al.*, 2004] R. Dale, R. Calvo, and M. Tilbrook. Key Element Summarisation: Extracting Information from Company Announcements. In *Proc. of the 17th Australian Joint Conf. on AI, 7th-10th Dec. 2004, Australia*, 2004.

[Grishman and Sundheim, 1995] Ralph Grishman and Beth Sundheim. Design of the MUC-6 Evaluation. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference held in Columbia, Maryland, November 6-8, 1995*, Los Altos, Ca., 1995. Morgan Kaufmann.

[Grishman and Sundheim, 1996] Ralph Grishman and Beth Sundheim. Message Understanding Conference-6: A Brief History. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, Los Altos, Ca., 1996. Morgan Kaufmann.

[Holte, 1993] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11:63–91, 1993.

[John and Langley, 1995] George H. John and Pat Langley. Estimating Continuous Distributions in Bayesian Classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[Landwehr *et al.*, 2005] N. Landwehr, M. Hall, and E. Frank. Logistic Model Trees. *Machine Learning*, 59(1/2):161–205, 2005.

[Mazur and Dale, 2006] P. Mazur and R. Dale. Named entity extraction with conjunction disambiguation. In *Proc. of 5th LREC, Italy, 24th-26th May*, pages 1752–1755. ELRA, 2006.

[McDonald *et al.*, 2005] Ryan McDonald, Koby Crammer, and Fernando Pereira. Flexible text segmentation with structured multilabel classification. *EMNLP*, 2005.

[Mikheev *et al.*, 1998] A. Mikheev, C. Grover, and M. Moens. Description of the LTG System Used for MUC-7. In *Proc. of MUC-7 Conf.*, 1998.

[Platt, 1999] John C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, pages 185–208, Cambridge, MA, USA, 1999. MIT Press.

[Quinlan, 1993] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[Rojas, 1996] Raúl Rojas. *Neural networks: a systematic introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996.

[Sang and Meulder, 2003] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of the 7th Conference on Natural Language Learning*, pages 142–147. Edmonton, Canada, 2003.

[Sang, 2002] Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In Dan Roth and Antal van den Bosch, editors, *Proceedings of the 6th Conference on Natural Language Learning*, pages 155–158. Taipei, Taiwan, 2002.

[Solorio, 2004] T. Solorio. Improvement of Named Entity Tagging by Machine Learning. Technical Report CCC-04-004, Coordinacin de Ciencias Computacionales, 2004.

[Steedman, 1985] M. Steedman. Dependency and Coordination in the Grammar of Dutch and English. *Language*, 61:523–568, 1985.

[Witten and Frank, 2005] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2005.