The Utility of Information Extraction in the Classification of Books

Tom Betts, Maria Milosavljevic, and Jon Oberlander

School of Informatics, University of Edinburgh,
2 Buccleuch Place, Edinburgh EH8 9LW, UK
tom@4angle.com, {mmilosav,jon}@inf.ed.ac.uk
http://www.inf.ed.ac.uk

Abstract. We describe work on automatically assigning classification labels to books using the Library of Congress Classification scheme. This task is non-trivial due to the volume and variety of books that exist. We explore the utility of Information Extraction (IE) techniques within this text categorisation (TC) task, automatically extracting structured information from the full text of books. Experimental evaluation of performance involves a corpus of books from Project Gutenberg. Results indicate that a classifier which combines methods and tools from IE and TC significantly improves over a state-of-the-art text classifier, achieving a classification performance of $F_{\beta=1} = 0.8099$.

Key words: Information Extraction, Named Entity Recognition, Book Categorisation, Project Gutenberg, Ontologies, Digital Libraries

1 Introduction

Books have long been classified in ontologies to help users locate relevant titles. The emergence of digital libraries and on-line resellers has released the classification of books from the constraints of a physical space, allowing the use of multi-labelling, and even folksonomies.[1] This is very useful, since books are often inconsistently classified in different systems, or not satisfactorily classified at all.

Text categorisation (TC) is the task of classifying texts, based on their content, into one or more predefined categories. Popular applications include filtering e-mail messages for spam, or indexing newswire articles. Information Extraction (IE) is a process by which we can identify structure within unstructured natural language text. The IE task consists of a series of subtasks, one of which is Named Entity Recognition (NER), which is concerned with finding entities within a text such as people, locations and organisations. Intuitively, we expect that reducing unstructured text into a structured format should provide leverage in TC. Although there has been considerable research in both TC and IE, only a limited amount of work has combined them. We have found no other work which uses content-based categorisation on full-text books or large texts. Our

[1] See, for example www.librarything.com

approach is closest to [12] in terms of representation, combining phrases (where we use named entities) with bag-of-words; cf. the models discussed in Section 3.

Here, we set out to show that by using IE, information derived automatically from the full text of books can lead to good TC performance. We first determine the type of information that can easily be extracted from books using NER. Second, we explore how this information can be incorporated into a TC task and develop a framework for its inclusion. Finally, we show that the use of this information leads to a statistically significant improvement in performance when compared to a state-of-the-art text classifier.

2 Motivation and Context

Major players in the text industry have a considerable interest in the digitisation of books. For example, Amazon.com are scanning books for their 'Search Inside the Book' programme, the Million Book Project at Carnegie Mellon has scanned over 600,000 texts[2] and Google is expected to digitise 10.5 million books for their Google Books project [4]. For the purpose of this work, we use books from Project Gutenberg,[3] which contains over 19,000 copyright-free electronic texts.

Categorising books is non-trivial due to subjectivity and the sheer volume and variety of titles available. Manual classification is time consuming and, therefore, impractical for large collections such as digital libraries or online retailers, who stock millions of different titles. Hence, content-based classification may prove a useful alternative. However, the challenges of heterogeneity and scalability distinguish this problem from traditional automated text categorisation. For example, the length of full-text books may be orders of magnitude greater than newswire stories or e-mail messages classified in past work.

In this work, we use the Library of Congress Classification[4] (LoCC), an ontology for categorising books according to their subject, used by most academic and research libraries. Although LoCC was created to satisfy the constraints of a physical library, it can also be used to categorise digital collections. Because a library cannot contain many copies of the same book, books are traditionally placed into only one or at most two LoCC categories. But digital collections have no such constraints, and in fact, assigning multiple categories is highly desirable in order to help diverse users locate relevant books. Nürnberg et al. [13] provide further detail on the differences between digital and physical libraries.

2.1 Approaches to Book Categorisation

Manually categorising books has long been the concern of librarians. They assign books to predefined categories from hierarchical ontologies such as LoCC or the Dewey Decimal Classification. Early work on the automated categorisation of library texts (not necessarily books) focussed on the content of abstracts [2].

[2] `http://www.library.cmu.edu/Libraries/MBP_FAQ.html`
[3] `http://www.gutenberg.org`
[4] As outlined at `http://www.loc.gov/catdir/cpso/lcco/lcco.html`

A more recent approach to categorising books has been to use structured metadata, such as titles or subject headings—a less hierarchically structured form of subject annotation for books [8, 9, 6]. Frank and Paynter's [6] work is closest to ours, taking a machine learning approach to categorisation. Using Library of Congress Subject Headings, they report an accuracy of $\sim 55\%$ when categorising 50,000 books into 4214 categories. However, their classification is based on curated metadata, which may be susceptible to annotator inconsistency; by contrast, our approach is content-based. Mooney and Roy [11] use content-based text categorisation in a recommender system for books. They use both structured metadata and unstructured text for categorisation, but limit the use of text to product descriptions, editorial and user reviews.

3 Methods and Tools

Our central hypothesis is that structured information that can be automatically extracted from unstructured book texts is useful for the purpose of categorising them. In testing this hypothesis, we use Named Entity Recognition (NER) in addition to a standard bag-of-words approach. NER can be achieved with good performance, suffers acceptable domain coupling, and should be computationally tractable [5]. Furthermore, named entities can provide important cues to the topic of a text. Thus, we pursue NER as a technique for the extraction of structured information, and we use pre-existing tools to extract entities from each book text. We explore the use of names of people, organisations, locations and dates, based on the assumption that these entities should help to discriminate between the categories in our corpus.

3.1 Corpora

We use full-text books from Project Gutenberg, using their published LoCC as the 'gold standard'. Gutenberg texts have been assigned a LoCC including schedule, denoted by the first letter (e.g. D: History: General and Eastern Hemisphere), and subclass, denoted by subsequent letters (e.g. DA: History: General and Eastern Hemisphere: Great Britain, Ireland, Central Europe). In order to experiment with the extraction of structured information from the available texts, and effectively assess its usefulness in categorisation, we created two corpora containing all English texts from a subset of Gutenberg categories.

Our **development corpus** is used to experiment with and develop our classifiers. It includes 810 texts from 28 categories, including subclasses of schedules B (Philosophy, Psychology, Religion) and D (History: General and Eastern Hemisphere). These categories are selected based on their availability, the intuition that named entities may discriminate classes, and their content-based similarity.

Our **extended corpus** (comprising 1029 texts in 52 categories) combines the development corpus with texts from schedules H (Social sciences) and Q (Science). This provides further texts which may exhibit similarity with the development corpus; however, we do not manually alter our models specifically

to discriminate H or Q categories. There are no features that exploit domain knowledge specific to these categories. Schedule Q, containing physical science subcategories, poses challenges by adding subject diversity, disparate linguistic style and texts which may not be easily discriminated by named entities.

3.2 Models for Representing Texts

In order to assess the utility of named entities in our categorisation task, we created three models that can be used to represent a text:

BOW Model: Bag-of-Words The first model uses a multinomial bag-of-words to represent the text. This model is considered state-of-the-art for TC, and is the baseline against which we evaluate other techniques.

NER Model: Bag-of-Named-Entities There have been several efforts to find more sophisticated representations of texts that capture additional 'knowledge' or 'meaning' present in a text. Techniques include the use of n-gram phrases [10, 15, 7, 12, 3] and word senses [15, 12] instead of words. We create a representation of each book text as a variant of a bag-of-phrases, using n-gram entities extracted with NER. The classification of entities is exploited so that entities with the same string value, but different types (e.g. Chelsea-LOC and Chelsea-PER), are treated as different features.

GAZ Model: Generalising Named Entities One apparent difficulty with the use of phrases in TC is that they suffer from inferior statistical qualities when compared to word-based models [12, 14]. A named entity model will have similar problems, since it is not clear that the same named entity will be observed across texts. An attempt is made to reduce sparse data difficulties by generalising the observed named entities into a smaller number of synthetic features. We incorporate domain knowledge to create features that exploit ontological structure. For example, we observe that the subclasses of schedule D (History) are organised by geographical region. Features are created using geographic gazetteers, defined according to the region of each category. For subclasses of schedule B (Philosophy, Psychology, Religion), we can use gazetteers of names of relevant people and organisations.

Although we evaluate the baseline BOW, NER and GAZ models individually, previous work with phrases for TC [12, 10] has combined a bag-of-words feature space with phrases to overcome their inferior statistical qualities [14]. We use a similar approach, creating a continuous feature space by combining the feature spaces of individual models. The models are:

NER-GAZ combines the feature spaces of NER and GAZ, exploiting GAZ as smoothing, as discussed above;

BOW-NER combines NER features with the baseline BOW feature space, helping to reduce the negative effects of phrase-based representations, such as lack of partial matching and misclassified or missing named entities; and,

BOW-NER-GAZ involves the combination of all three individual models.

3.3 Classifier Configurations

This task is a multiclass problem because Gutenberg texts are only assigned one of the available LoCC. We evaluate two approaches to multiclass classification.

Multiclass This requires training a single classifier to distinguish between all categories, where the objective function we optimise is the accuracy of classification when assigning one of many categories. This classifier exploits the mutual exclusivity of class labels during training, whereas binary classifiers combined in one-vs-all classification (described below) treat category assignment decisions independently (it is only during the final assignment that mutual exclusivity is exploited). The classifier used in these experiments is a multiclass generalisation of SVM, SVM^{multiclass}[17].

One-vs-All We evaluate one-vs-all classification because it can assign multiple labels to a book (motivated in Section 2). Unfortunately, training and evaluating classifiers combined using one-vs-all may be computationally expensive, due to the need to use one classifier per category. However, it follows that the complexity of individual classifiers should be relatively low (compared to the direct approach described above). We use multiple instances of SVM^{light}[16].

We make classifications by selecting the classification that we are most confident about[5] and pursue two approaches to category assignment. A **relaxed** classification is made by assigning the category of the classifier outputting the largest positive value. A **forced** classification is made by assigning the 'least bad' classification—selecting the largest value, regardless of whether positive or negative. The relaxed classifier, which comes closest to generalising to a multi-label classification task, guarantees that at most one category is assigned, but does not guarantee to assign any. The intuitive result of forcing a classification would be an improvement in recall, but at the expense of precision.

3.4 Term Reduction and Feature Selection

Initial experiments were conducted using the entire feature space, and even with our relatively small corpus, execution was prohibitively slow. Two processes can reduce the size of models—the removal of infrequent terms and feature selection.

Term reduction involves removing terms that occur with high or low frequency. Due to the scale of texts being represented, we opt for maximum reduction, removing terms that occur fewer than five times in the corpus [14]. This provides an initial reduction in model size but is not sufficiently aggressive. However, it is a vital process because the technique that we use for feature selection, χ^2 [19], cannot provide accurate estimates for low frequency terms.

We use χ^2 to estimate the lack of independence between class labels and features. In particular, we use Yang and Pedersen's [19] definition of χ^2_{max} to

[5] Direct comparisons are made between binary classifiers even though we would not expect these confidences to be normalised, as discussed in Section 5.

obtain a single value for each feature. Features are sorted by χ^2_{max}, and the n (a variable in our experiments) largest are selected for inclusion in a model.

3.5 Evaluation

There is considerable choice of metrics for performance evaluation, including micro-averages, a per-document metric, and macro-averages, performing averaging over categories. Given that a motivation for this work is a reduction in human effort required to categorise texts, it follows that a goal is accurate classification of as many *texts* as possible, rather than accurate classification of as many *categories*, since the latter could result in the accurate categorisation of very few texts. Evaluation and optimisation, therefore, focus on micro-average performance. $F_{\beta=1}$ is used as the primary metric for binary decisions and we occasionally discuss this in terms of precision and recall to provide further insight.

4 Results

Due to the quantity of metrics used for analysing experimental output, it is not feasible to comprehensively document all our results in this paper. For clarity, situation-specific visualisations and statistics are included to facilitate this discussion, and an overview of micro-$F_{\beta=1}$ for each classifier, displaying results for every model, is provided in Fig. 1. A comprehensive account of the results can be found in [1]. Unless otherwise stated, metrics quoted are micro-averages. The best result obtained using our baseline model was $F_{\beta=1} = 0.7914$, and this was improved to $F_{\beta=1} = 0.8099$ when incorporating named entities.

4.1 Baseline: Bag-of-Words

We begin this evaluation with our baseline, noting that micro-$F_{\beta=1}$ is indicated in Fig. 1. Both classifiers that guarantee to make exactly one classification, one-vs-all forced (Fig. 1A) and direct multiclass (Fig. 1C) classifiers, exhibit relatively straightforward optimisation of $F_{\beta=1}$, which increases as features are added. However, for the forced one-vs-all classifier, performance begins to fall after 5000 features, labelled (a). In the relaxed one-vs-all classifier (Fig. 1B), $F_{\beta=1}$ performance begins a continuous decline after only 500 features, indicated (b). Between points (b) and (c), $F_{\beta=1}$ falls from 0.7264 to 0.6564, which corresponds to a fall in the number of correct classifications from 507 to 405. This can be explained by a fall in certainty of classifications as the number of features increases. It follows that the classifier makes fewer category assignments, and precision increases, because the most 'uncertain' texts were those most likely to be incorrectly classified. Increasing the number of model features eventually reduces the ability of classifiers to generalise because the trained classifiers will tend to overfit, given our limited training data. This is also true for forced one-vs-all classifiers. Continuing to force classifications leads to a decrease in $F_{\beta=1}$,

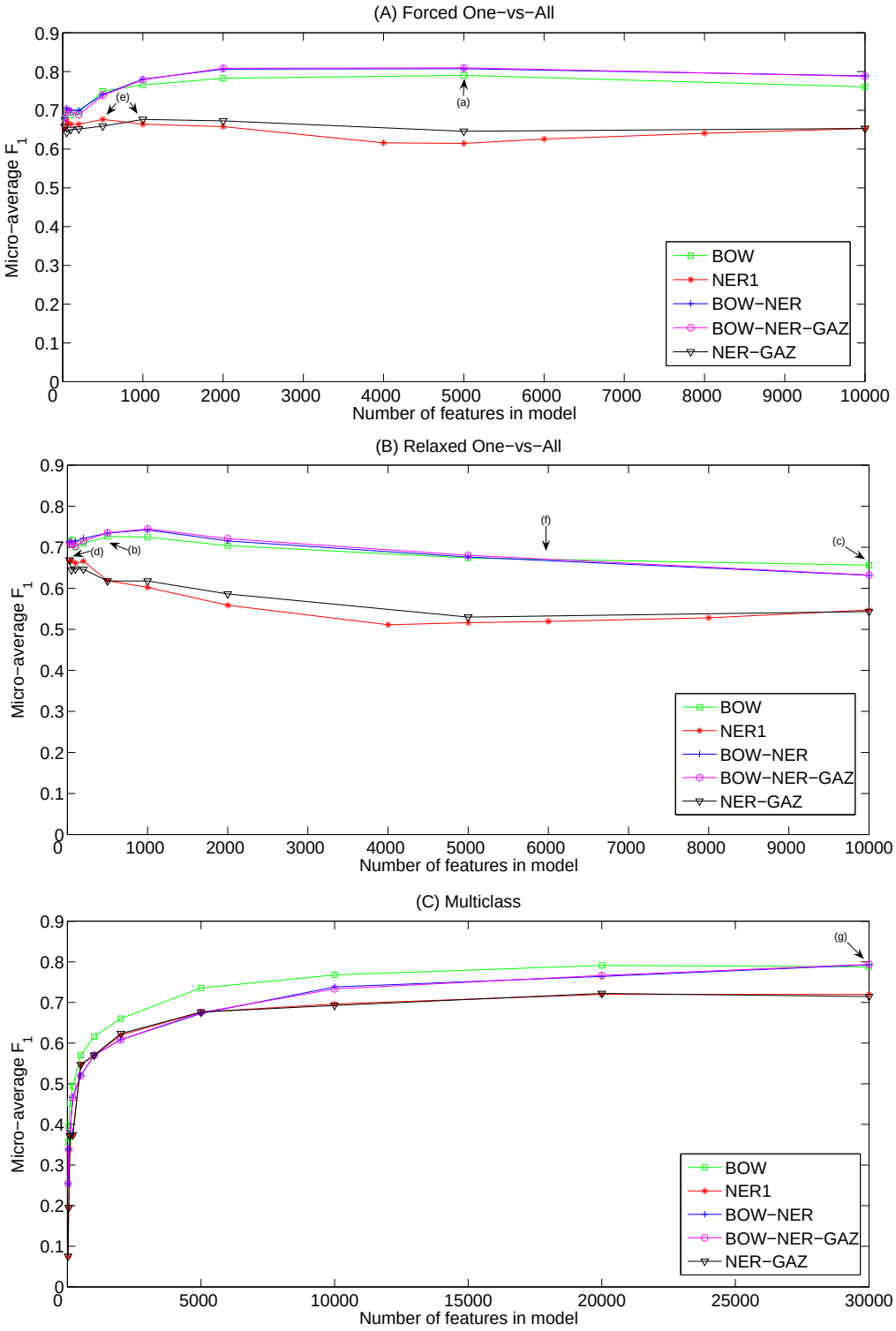

Fig. 1. Micro-average $F_{\beta=1}$ as number of features is varied for each classifier configuration (one per plot) and document model (all models per plot)

which we observe at point (a). From Fig. 1C we can see that multiclass classification appears to avoid overfitting (in the limited dimensions that we explore), and may be more robust because class assignment decisions are dependent on one another, whereas local decisions in one-vs-all classifiers are independent.

Comparing the classifier configurations, we observe that one-vs-all classifiers achieve respectable performance with very few dimensions, whereas multiclass classifiers require larger representations. Multiclass and one-vs-all forced classifiers exhibit the best $F_{\beta=1}$, which is broadly similar between them (~ 0.79), suggesting that the requirement for assigning exactly one class can be effectively exploited. Although the relaxed classifier performs worst overall, it achieves a precision of 0.955, the highest in the evaluation, but at the expense of recall.

4.2 Combining Models

In this section we evaluate the hybrid models, discussed in Section 3.2, that facilitate the combination of features from individual models.

NER-GAZ: Named Entities With Smoothing Motivating the GAZ model in Section 3.2, we discussed the need for smoothing of NER model features to prevent overfitting. By combining features from NER and GAZ, we hope to improve recall in the NER model. In Fig. 1A and 1B we observe a general rise in $F_{\beta=1}$ when comparing the NER-GAZ and NER models, which we attribute to to an increase in average recall. Although generally NER-GAZ offers improvements over the NER model, optimal $F_{\beta=1}$, indicated (d) and (e), is only marginally improved (relaxed), or not at all (forced). However, the generally increased $F_{\beta=1}$ offered by NER-GAZ is desirable in order to aid generalisation without exhaustive searches of parameter space. As seen in Fig. 1C, multiclass classification appears largely unaffected by the addition of GAZ features to the NER model.

BOW-NER and BOW-NER-GAZ: Combining Word and Named Entity Based Models By including named entities in addition to word-based features, we hope to improve precision without reducing recall. When evaluating performance of optimised models (summarised in Table 1), both appear to improve performance over any of their constituents. Furthermore, for one-vs-all classifiers this improvement appears substantial. We can see from the model performance in Fig. 1A and 1B that these models tend to perform similarly to their BOW counterparts. Given that the BOW feature space is denser, this is perhaps unsurprising, and indicates that BOW features have greater influence over the performance of these hybrid models than NER or GAZ features.

In Fig. 1A we see that forced one-vs-all classifiers exhibit consistent improvement for BOW-NER and BOW-NER-GAZ models, compared to the baseline BOW model. They also improve relaxed classifiers when the number of features included in the model is < 6000, labelled (f); however, with more features a BOW classifier outperforms the combined models. As previously discussed for the baseline, overall performance of relaxed classifiers suffers from poor recall.

Table 1. Development corpus: optimal micro-$F_{\beta=1}$

Model	One-vs-All (Forced)		One-vs-All (Relaxed)		Multiclass	
	Features	$F_{\beta=1}$	Features	$F_{\beta=1}$	Features	$F_{\beta=1}$
BOW	5000	0.7901	500	0.7264	20000	0.7914
NER	500	0.6765	50	0.6676	20000	0.7198
GAZ	28	0.6407	28	0.6499	28	0.5802
NER-GAZ	1000	0.6765	25	0.6689	20000	0.7222
BOW-NER	5000	0.8074	1000	0.7424	30000	0.7926
BOW-NER-GAZ	5000	0.8099	1000	0.7447	30000	0.7938

Compounding this with the precision-improving characteristics of NER features may be the cause of this eventual decline in combined model performance. The opposite appears true of multiclass classification, in which $F_{\beta=1}$ grows more slowly for the combined models than the baseline. In fact, multiclass does not benefit from the combined models until the final dimension that we sample, labelled (g), where the combined models offer marginal improvement.

4.3 Development Corpus: Discussion of Results

Table 1 contains the optimal parameters (number of features in the model) for every model and classifier configuration, and the respective development corpus $F_{\beta=1}$. For comparison, a naïve classifier could obtain $F_{\beta=1} = 0.1843$ on the development corpus by classifying every document as the most numerous category. We note that the BOW-NER-GAZ model consistently produces the best $F_{\beta=1}$. Although the improvements for the combined feature spaces do not always appear substantial over their discrete constituents, it is satisfying that the combined models at least exhibit the improvements intended by design. NER-GAZ improves recall and categorisation performance on texts from unpopular categories (over the NER model), and the BOW-NER and BOW-NER-GAZ models improve the baseline model by enhancing precision, which in many cases leads to an increase in $F_{\beta=1}$.

Statistical Significance In order to determine whether the performance increases reported between models and our baseline are likely to have occurred by chance, we perform statistical significance testing. We use the micro sign test (or s-test) [18], which compares two systems (A and B) based on the binary decisions that they make. This test evaluates models at a micro level, which aligns with our aim to optimise micro-$F_{\beta=1}$. The s-test produces a (one-sided) P-value for the hypothesis that system A performs better than system B. A subset of the results of this testing are given in Table 2. A smaller P-value indicates a more significant result, and we assume that a P-value > 0.1 indicates that the improvement reported for system A over B is not statistically significant. Using the s-test, we find that two results are significant, and these are both forced one-vs-all classifications. The BOW-NER-GAZ model offers the most significant improvement over the baseline, indicated by the smallest P-value.

Table 2. Statistical significance test results for development corpus using s-test

System A	System B	Classifier	s-test[a]
BOW-NER	BOW	Forced One-vs-All	$>$
BOW-NER-GAZ	BOW	Forced One-vs-All	$\gg$
BOW-NER	BOW	Relaxed One-vs-All	$\sim$
BOW-NER-GAZ	BOW	Relaxed One-vs-All	$\sim$
BOW-NER	BOW	Multiclass	$\sim$
BOW-NER-GAZ	BOW	Multiclass	$\sim$

[a] where "$\gg$" indicates P-value ≤ 0.05; "$>$" indicates $0.05 <$ P-value ≤ 0.10; and "$\sim$" indicates P-value > 0.10

4.4 Extended Corpus Evaluation

We analyse classification performance on our extended corpus, as introduced in Section 3.1. Models were trained and evaluated using the parameters established on the development corpus. The results are given in Table 3.

It is encouraging to see similar patterns between these results and those for the development corpus given in Table 1. In particular, we see that both BOW-NER and BOW-NER-GAZ models achieved equal or better performance than the baseline, and in the case of forced one-vs-all and multiclass classifiers, this improvement was larger than that reported for the development corpus. It is interesting that the BOW-NER and BOW-NER-GAZ models performed identically for all classifiers. Further inspection reveals that no GAZ features were selected for use in the BOW-NER-GAZ model, presumably because these features are noisy, given that they are not tailored to the new categories.

Table 3. Extended corpus: micro-$F_{\beta=1}$ based on development corpus parameters

Model	One-vs-All (Forced)		One-vs-All (Relaxed)		Multiclass	
	Features	$F_{\beta=1}$	Features	$F_{\beta=1}$	Features	$F_{\beta=1}$
BOW	5000	0.6673	500	0.6359	20000	0.6683
NER	500	0.2917	50	0.4761	20000	0.6341
GAZ	28	0.3522	28	0.3506	28	0.3844
NER-GAZ	1000	0.5141	25	0.4907	20000	0.55
BOW-NER	5000	0.7092	1000	0.6360	30000	0.6761
BOW-NER-GAZ	5000	0.7092	1000	0.6360	30000	0.6761

Although performance is universally worse than for the development corpus, this is to be expected, and we cannot isolate the cause of this penalty. The extended corpus is a harder corpus to categorise: it has more categories; the new categories are poorly populated with texts; it contains subcategories that are less likely to be easily discriminated using named entities; and we do not tailor our approach to this corpus as we did during development.

5 Conclusions and Further Work

Our classification system makes progress towards the classification of books using their full text. The extraction of structured data from full-text books, using Information Extraction, is explored and evaluated for the purpose of assigning a single LoCC to each text. Furthermore, the techniques developed can be generalised to assign multiple category labels to a single text, although this is not evaluated here and is left for further work. In addition to assigning a categorisation, the extracted metadata (such as dates, and names of people and locations) may be useful for creating domain specific interfaces, such as maps for browsing the distribution of geographic entities, or timelines for browsing dates. Examples can be seen in the Perseus Digital Library and Gutenkarte.[6]

We have seen that there are implications for precision and recall when selecting classifiers, and that we can make very precise classifications using a relaxed one-vs-all classifier, or gain higher recall by selecting a forced architecture. Furthermore, we have also seen that one-vs-all classifiers require far fewer features than multiclass classifiers to perform optimally. This is also true of NER models, which required fewer dimensions to achieve optimal performance. This observation may be critical for the scalability of a solution to larger corpora.

We also found that in one-vs-all classifiers, adding features tended to be precision improving, but at the expense of recall, leading to a fall in $F_{\beta=1}$. This overfitting is most likely an artifact of limited training data. Of the architectures, direct multiclass classification appears to be the most robust and in the feature space sampled, we found little evidence of overfitting. However, given that the data is partitioned differently for each binary classifier used in one-vs-all, the respective hyperplanes and distances from hyperplanes to data points (and hence confidences) will not be normalised by default. Although it is not expected to affect the consistency of our results, it may be suboptimal to make decisions by direct comparison of these output confidences.

We found that combining our models based on named entities with bag-of-words representations resulted in an increase in performance over the baseline system, and furthermore, that this improvement has some statistical significance. We expect that these techniques could be widely applied to text categorisation and that the results described may not be specific to books.

Following the work in [6], we intend to develop mechanisms for evaluating the degree of incorrectness in misclassifications of books. In particular, we note that many misclassifications are in fact too general or too specific rather than strictly incorrect. In order to assess the utility of a less strict classification metric, we aim to define a 'lenient' metric as partial success, whereby, for example, a text from category DA may have been classified as D, or vice versa. We will then use this in order to assess the extent to which a classifier appears to make at least partially correct decisions.

[6] `http://www.perseus.tufts.edu` and `http://www.gutenkarte.org`

Acknowledgments

The authors would like to thank Robert Japp and Sharon Givon for their input in this work.

References

1. Betts, T.: Using Text Mining to Place Books into an Ontology. Masters thesis, University of Edinburgh, Edinburgh, UK. (2006)
2. Borko, H.: Measuring the reliability of subject classification by men and machines. American Documentation. **15** (1964) 268–273
3. Caropreso, M.F., Matwin, S., Sebastiani, F.: A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Text Databases and Document Management: Theory and Practice. Idea Group Publishing. (2001) 78–102
4. Crane, G.: What Do You Do with a Million Books? D-Lib Magazine. **12**:3 (2006)
5. Curran J.R., Clark, S.: Language independent NER using a maximum entropy tagger. In: Proceedings of CoNLL-03, the Seventh Conference on Natural Language Learning. Edmonton, Canada. (2003) 164–167.
6. Frank, E., Paynter, G.W.: Predicting library of congress classifications from library of congress subject headings. J. of the American Society for Information Science and Technology. **55**:3 (2004) 214–227
7. Fürnkranz, J.: A study using n-gram features for text categorization. Technical Report OEFAI-TR-9830, Austrian Institute for Artificial Intelligence. (1998)
8. Hamill, K.A., Zamora, A.: The Use of Titles for Automatic Document Classification. J. of the American Society for Information Science. **31**:6 (1980) 396–402
9. Larson, R.R.: Experiments in automatic Library of Congress Classification. J. of the American Society for Information Science. **43**:2 (1992) 130–148
10. Mladenić, D., Globelnik, M.: Word sequences as features in text learning. In: Proceedings of ERK-98, the Seventh Electrotechnical and Computer Science Conference. Ljubljana, Slovenia. (1998) 145–148
11. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: Proceedings of DL-00, 5th ACM Conference on Digital Libraries. San Antonio, US. (2000) 195–204
12. Moschitti, A., Basili, R.: Complex linguistic features for text classification: a comprehensive study. In: Proceedings of ECIR'04, Sunderland, UK. (2004)
13. Nürnberg, P.J., Furuta, R., Leggett, J.J., Marshall, C.C., Shipmann, F.M.: Digital Libraries: Issues and Architectures. In: Proceedings of the 1995 ACM Digital Libraries Conference. Austin, TX. (1995) 147–153
14. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys. **34**:1 (2002) 1–47
15. Scott, S., Matwin, S.: Feature engineering for text classification. In: Proceedings of ICML-99. Bled, Slovenia. Morgan Kaufmann Publishers. (1999) 379–388
16. Joachims, T.: SVM^{light} 6.01, `http://svmlight.joachims.org` (2004)
17. Joachims, T.: SVM^{multiclass} 1.01, `http://svmlight.joachims.org/svm_multiclass.html` (2004)
18. Yang, Y., Liu, X.: A re-examination of text categorization methods. Proceedings of the 22th ACM SIGIR. Berkley, US. (1999) 42–49
19. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proceedings of ICML-97. Nashville, US. (1997) 412–420