Document Classification in Structured Military Messages

Oliver Carr and Dominique Estival
Human Systems Integration Group
Command and Control Division
Defence Science and Technology Organisation
{Oliver.Carr,Dominique.Estival}@dsto.defence.gov.au

Abstract

We present new results for the DSTO project on document classification of military messages. We report more specifically on the improvements to the Part-Of-Speech (POS) tagging, a probabilistic process that assigns a tag to a token, and discuss the training for Date Time Groups POS tags. A new implementation of the rule-based classifier is described. The results obtained on two databases of real military messages are encouraging and the document classification module has now been integrated with a query user interface.

1 Introduction

In (Carr and Estival, 2002), we presented the first tentative results of the Document Classification project we have been conducting at DSTO and we discussed the shortcomings of the approach we were using. In this paper, we present the results we have obtained in the continuation of that project, after having implemented improvements in the POS tagging component and taken a different approach for the rule-based classifier component. These results show that rule-based classifiers can give reasonable results for structured textual information, when using appropriate language models for POS tagging.

1.1 Goals of the project

A large part of the Defence Information Environment (DIE) used at the Deployable Joint Force Headquarters (DJFHQ) is based on Lotus collaborative and messaging applications. The staff members of DJFHQ use Lotus databases to log operational events and Lotus e-mail for actions and administrative functions. Around 200 messages per day are entered into these Lotus Notes log databases. Many DJFHQ staff members have expressed difficulty in finding particular information in their information reservoirs and our goal is to develop a more effective query interface between DJFHQ staff and their information reservoirs. This work already resulted in the development of the Query Building Interface (QBI), which was designed to create a better search interface to multiple log databases and to the users e-mail database. The rule-based Document Classifier we describe here has been trained and evaluated on Lotus operational log databases (OPS logs) from DJFHQ. It can now provide a categorisation for each document from the OPS logs and is integrated with QBI, as described in Section 7.2.

1.2 Proposed Architecture

Fig.1 below shows how QBI and the Document Classifier could be integrated in the existing IT infrastructure. In this new Server Environment, both QBI and the Document Classifier interface with the Lotus Notes database.

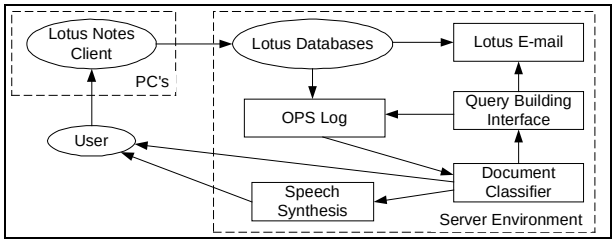

Fig.1 Proposed Architecture

In this architecture, users enter and access documents through Lotus Notes as they do now, and they receive notification of the document classification. One possible scenario is to use a Text-to-Speech module to warn of the arrival of

some pre-specified document types, eg. NOTICAS (Notification of Casualty) or MEDSITREP (Medical Situation Report). For a NOTICAS, the injured person's details could be automatically retrieved and read out to the Commander or sent to a different display.

1.3 Structure of the system

The Document Classifier module in Fig.1 is named SOP-MRC (Standard Operating Procedures Rule Based Multiclass Classifier), and as shown in Fig.2, it consists of two main components: a Part-Of-Speech (POS) Tagger and a Classifier.

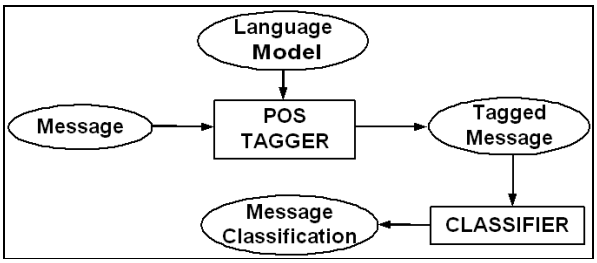

Fig. 2 The SOP-MRC module

The POS Tagger component, described in Section 3, is a probabilistic process that assigns a tag to a token. We also describe the training of this component in Section 3 and present our extension of the POS tagset for the military message data.

The Classifier component, described in Section 4, takes as input the list of pairs <token> <tag> produced by the POS Tagger for an incoming message and uses rules to determine the document type (including "free text") of the message.

We present in Section 5 the results we obtained on data from two military exercises. One database (VP-02) contains messages used to train the POS Tagger and to develop the classifier rules. The second database (TT-01) contains similar documents from another military exercise.

2 Shortcomings of the previous approach

There were two types of problems with the first approach we took to classify the SOP documents from the DJFHQ message database. The first one was that the data did not conform to expectations and the second one was that the classification rules were too brittle. Both issues have been addressed by the new approach to writing the classification rules described in Section 4.2.

DJFHQ operators use formatted text in the free text fields of their Lotus Notes operational log databases. This formatted text is defined by Standard Operating Procedure (SOP) documents, and there are 88 different SOP Document Types corresponding to different message types. Our first approach had been to define rules based on the definition of the SOPs, which are available to the operators writing those messages as MS Word documents. However, the actual messages often do not follow the format prescribed by the SOPs and, in addition, they often contain attachments and other material, which makes classification more difficult. The new rules now take into account variations in the way operators actually write their messages and allow more flexibility in the classification. This is described in Section 4.

Another problem was that the POS tagset used by our POS Tagger did not cover some token types that are very important in military messages. In particular, one lesson from our earlier work was that it is necessary to recognise Date Time Group (DTG) expressions and that we would have to develop our own tagset to fit the military domain. The additions we made to the POS tagset, are discussed in more detail in Sections 3.2 and 3.5.

3 POS Tagging

3.1 QTAG

The POS tagger we chose to use is Qtag, a portable trainable language-independent probabilistic tagger developed by the University of Birmingham (Mason, 2003; Tufis and Mason, 1998). There are several training corpora available on the Internet to train POS taggers.[1] Qtag was originally trained with the Industrial Parsing of Software Manuals (IPSM) (Sutcliffe et al, 1996), which uses the Penn Treebank tagset, and it comes with the Birmingham – Lancaster Tagset and the associated resource file trained for English.

Qtag takes free text as input and outputs SGML, with each line containing the tag and the token it corresponds to. An example of input from our corpus and of the output produced by Qtag is given in Fig. 3.

[1] See e.g. the Automatic Mapping Among Lexico-Grammatical Annotation Models (AMALGAM) project: http://www.comp.leeds.ac.uk/amalgam/amalgam/multi-parsed.html

Input (VP-02)	Qtag Output
From HQCLSC, HSS facilities allocated to CLSC as follows, A. 34 Fd Hosp (U.K)	<w pos="IN">From</w> <w pos="JJ">HQCLSC</w> <w pos=",">,</w> <w pos="NN">HSS</w> <w pos="NNS">facilities</w> <w pos="VBN">allocated</w> <w pos="TO">to</w> <w pos="NN">CLSC</w> <w pos="CS">as</w> <w pos="VBZ">follows</w> <w pos=",">,</w> <w pos="NN">A.</w> <w pos="CD">34</w> <w pos="NN">Fd</w> <w pos="NN">Hosp</w> <w pos="NN">(U.K)</w>

Fig. 3 Output of Qtag using original tagset

To deal with the particular type of text contained in SOP documents, 59 new POS tags (mainly formatting tags) were added to the original tagset of 45 tags. Fig. 4 shows the same text as Fig. 3, tagged by Qtag using the language model containing these additional domain specific tags.[2,3]

Input (VP-02)	Qtag Output
From HQCLSC, HSS facilities allocated to CLSC as follows, A. 34 Fd Hosp (U.K)	<w pos="From">From</w> <w pos="VB">HQCLSC</w> <w pos=",">,</w> <w pos="NN">HSS</w> <w pos="NNS">facilities</w> <w pos="VBN">allocated</w> <w pos="TO">to</w> <w pos="NN">CLSC</w> <w pos="as">as</w> <w pos="NPS">follows</w> <w pos=",">,</w> <w pos="FrmA">A.</w> <w pos="CD">34</w> <w pos="NN">Fd</w> <w pos="NN">Hosp</w> <w pos="JJ">(U.K)</w>

Fig. 4 Output of Qtag using new tagset

3.2 Date Time Groups (DTGs)

As was described in (Carr and Estival, 2002), the analysis of our previous results showed that they were unsatisfactory in part because the POS Tagger did not recognise Date Time Groups (DTGs), which are very common in our texts, and which play an important role in document type recognition for SOPs. For the purpose of document classification, a DTG is a single unit of information, but there are 3 types of DTGs that appear in the SOPs:

- DTG_S, with time and time zone information,
- DTG_M, with day, time and time zone information,
- DTG_L, with day, time, time zone, month, and year information.

Examples of these are given in (1) with their corresponding POS tag.

(1) DTG_S 1259Z
DTG_M 310745Z
DTG_L 200830ZAUG02

Although the 3 different types of DTGs are not often distinguished by the classifier rules, the POS Tagger needs to be trained on these 3 different DTG types, to avoid confusion with other alphanumeric strings. (2) is an example of the output from Qtag, where the DTG_S tag is correctly assigned to the text "1100K".

(2) <w pos="at">AT</w>
<w pos="DTG_S">1100K</w>
<w pos="NN">C130</w>

As shown in Table 1, the baseline performance of Qtag (trained on 80% of the VP-02 data and tested on the remaining 20%) for DTGs was fairly low. This is due to the inadequate training data for DTGs in this corpus, which comes from one military exercise covering a short period of time, and thus conatining few variations for dates.

	POS Baseline			
	DTG_S	DTG_M	DTG_L	All Tags
Recall	9.68%	15.72%	24.28%	74.39%
Precision	6.90%	13.74%	14.86%	75.23%

Table 1 Baseline Performance of Qtag

3.3 Training with additional data.

To improve recognition of DTGs, we decided to create additional examples of DTGs to boost the training data for Qtag. For each of the DTG types, additional data was created in a systematic way to obtain instances of DTGs covering a wider range of dates and times. Table 2 shows the performance of Qtag with additional training data for DTGs.

[2] HQCLSC stands for "Headquarters Combined Logistics Support Command". Note that the POS tags assigned to it (JJ in Fig.3 and VB in Fig.4) are incorrect, as are several of the other tags.

[3] The tag <FrmA>, meaning a "formatted A character", covers the strings "\nA.", "\n(A)" and "\nA)".

	additional DTG_M			
	DTG_S	DTG_M	DTG_L	All Tags
Recall	100.00%	98.91%	95.65%	90.81%
Precision	80.52%	98.91%	95.65%	85.94%
	additional DTG_S			
	DTG_S	DTG_M	DTG_L	All Tags
Recall	100.00%	98.91%	95.65%	92.75%
Precision	80.52%	98.91%	95.65%	87.77%
	additional DTG_L			
	DTG_S	DTG_M	DTG_L	All Tags
Recall	100.00%	98.91%	95.65%	90.95%
Precision	80.52%	98.91%	59.46%	85.94%

Table 2 Additional training data for DTGs

Since the worst performing category had been DTG_M, we first added 482 additional instances of DTG_M to the training file for Qtag. The same process was repeated for DTG_S and DTG_L, with 158 and 8,063 additional instances respectively.

3.4 Overtraining

Table 2 shows that the performance of Qtag improved when the DTG_M data was first added but decreased significantly after DTG_L data was added (additional DTG_S training data did not make any significant difference). The decrease in performance after the DTG_L data was added is due to overtraining of Qtag. Using the same recursive algorithm, adding year information leads to the creation of many more instances of DTG_L than DTG_M and DTG_S and skews the training data, resulting in many false positives for that category. Since the training text with the added DTG_M and DTG_S gave the best performance, this is what we used to create the Qtag language model.

3.5 New POS tags for measure units

Table 3 shows examples of DTG tokens that were miscategorised by Qtag. We can see that most of these are in fact genuine DTGs, which is good news since the classifier rules are not concerned with the type of DTG (DTG_S, DTG_M, or DTG_L) but only with the occurrence of a DTG.

Number	Example	Tag	Correct Tag(s)
1	030/02OF170015ZMAY02	NN	DTG_L
1	2100S	DTG_S	LAT_LONG_S
2	4000FT,	DTG_S	DST
8	4000L	DTG_S	WGT
5	171659Z	NN	DTG_M
13	WEST	req	NN
8	5000M	DTG_S	DST
32	(2)AT	NN	Frm2 at
7	(0.5-0.7	NN	CD
1	A.151206KMAR02	NN	FrmA DTG_L
2	PD:130800K	NN	Pd DTG_M
1	C.LAND	NN	FrmC VB

Table 3 Errors in POS tagging for DTGs

Further analysis of the miscategorisations shown in Table 3 suggests ways in which the performance of the POS Tagger can be improved:

- § add additional training data for DTGs with different minute information than 0 or 5;
- § add POS tags for measure units, such as <WGT> for weights, <DST> for distances, <SPD> for speeds,
- § add POS tags for the various different types of Latitude and Longitude information or Grid reference.

Some examples are given in (3).

(3) <w pos= "WGT">2500KG</w>
<w pos ="DST">500NM</w>
<w pos = LATLONG>15.35S/151.20E </w>

In the end, 71 extra tags were added to the tagset, giving a total of 116 POS tags.[2] The new Qtag language model was trained on 80% of the POS tags from the VP-02 data. The remaining 20% (36862 tags from 430 messages from VP-02) were used to test the performance of the POS Tagger. Table 4 shows the results obtained for the DTG_S, DTG_M and DTG_L tags, after Qtag was trained with the additional training data for the new measure units POS tags.

New Tags	Recall	Precision
DTG_S	100.00%	91.18%
DTG_M	98.91%	99.27%
DTG_L	97.34%	100.00%

Table 4 DTGs with new language model

[2] There were 57 tags for formatting, 3 for DTGs, 3 for measure units and 6 for Lat/Long/Grid. Only 111 out of the 116 different POS tags appear in our test data.

Table 5 shows the overall results for Qtag using the macroaverage and microaverage statistics as described in (Sebastiani, 2001). Almost half of the POS tags in the test data were <NN>. We believe using the microaverage result without <NN> gives a better indication of performance.

Recall and Precision Averages	Recall	Precision
Macroaverage	89.69%	95.65%
Microaverage	97.84%	97.08%
Microaverage (no NN)	92.67%	94.75%

Table 5 Overall Performance of Qtag

4 Rule-based Classifier

Unlike most work on document classification (see Jackson and Moulinier, 2002, or Manning and Schütze, 1999), we do not rely on the semantic content of the documents to classify our documents, but take advantage of the very highly constrained structure of the documents. This is an example of *Category-Pivoted Text Classification* where the classifier is given a classification and must find which messages should be assigned to a given class, as opposed to *Document-Pivoted Text Classification*, which tries to determine the appropriate classifications for a set of documents (Sebastiani, 2002).

Quoting from (Jackson and Moulinier, 2002), there are two views of NLP: "Symbolic NLP tends to work top-down by imposing known grammatical patterns and meaning associations upon texts. Empirical NLP tends to work bottom-up from the texts themselves, looking for patterns and associations to model, some of which may not correspond to purely syntactic or semantic relationships." Empirical NLP has been widely used since the early 1990's while Symbolic NLP has been viewed less favourably. The system we describe here is in the tradition of Symbolic NLP, as the categories we use have been pre-defined and do not emerge from the data. However, at this point, classification is mainly performed on the basis of formatting structures, not on linguistic constructs.

Our first rule-based classifier used an "if, else" structure to parse the tags returned from Qtag one at a time. The document type was determined solely on the basis of the previous tag and the current tag, and only <Start> tags and the last one or two <End> tags were used to classify a message. A large amount of code (in Python) was written to implement this method, which turned out to be neither efficient nor successful.

Our first approach was too optimistic and too reliant on the document structure given in the SOPs, and our rules did not perform well. Our second implementation of a rule-based classifier uses regular expressions to state the rules. Regular expressions allow us to define more detailed rules and they also allow for more flexibility.

4.1 Regular Expressions as rules

As discussed in (Carr and Estival, 2002), the discrepancy between the format prescribed by the SOPs and the real text input by the operators was one of the main causes of errors. The use of regular expressions as rules allows flexibility in rule definition and result in shorter and more effective code. Several rules can be written to recognise one SOP document type.

The output of Qtag is read into a string. This string contains the list of POS tags for a message. Each rule recognises the tags for one SOP document type and allows any number of other tags in between. Only those POS tags required by the classifier rules are read into the Classifier. Having all the POS tags in a string also allows message headers and multiple SOPs to be pruned off or recognised differently very easily.

We give in (4) an example of a classifier rule for document type "P", where there can be any number of tags before <Frm1> and at least one instance of the separator or more tags before <Frm2>. In (5), we give an example of a message containing a document of type "P".

(4) P = ([a-za-z]|[A-Za-za-z0-9]|[A-ZA-Z]|[DTG_S]|[DTG_M]|[DTG_L]|[a-za-za-za-za-za-za-z]|[A-Za-za-za-z]|[ ]){0,}Frm1([a-za-z]|[A-Za-za-z0-9]|[A-ZA-Z]|[DTG_S]|[DTG_M]|[DTG_L]|[a-za-za-za-za-za-za-z]|[A-Za-za-za-z]|[ ]){1,}Frm2.([a-za-z]|[A-Za-za-z0-9]| [A-ZA-Z]|[DTG_S]|[DTG_M]|[DTG_L]|[a-za-za-za-za-za-za-z]|[A-Za-za-za-z]|[ ]){0,}

(5) CD DTG_M Frm1 CD From at Frm2 DTG_L

4.2 SOP-MRC rules

As mentioned earlier, the classifier rules were first created following the 88 SOP document definitions. They were later derived from a corpus

analysis and further refined after analysis of the results on the same corpus. The rules use mostly POS tags relating to formatting, eg. <Frm1> ("formatted 1") or <FrmB> ("formatted B"), but also some content information, with the POS tags for DTGs and <CD> (number). A total of 66 rules were used to recognise the 37 document types that appeared in the VP-02 data. Of these 66 rules, 44 rely on the POS tags for DTGs or <CD>.

One disadvantage of using regular expressions to implement classifier rules might be that they can be fairly long. The example in (4) is one of the shortest rules in terms of number of elements. However, this problem can be alleviated by the use of named groups and the Python interpreter is useful to test the regular expressions before they are included into the classifier.

It is also worth noting that these handcrafted rules were in fact written very quickly, much more quickly than "one rule in two days" as described by (Jackson and Moulinier, 2002).

Each message is tested against all the rules for SOP document types. If no match is found, then the document is assigned to the document type "Free Text". Some rules are in fact subsets of other rules. This defines a hierarchy of rules which can be used to determine the correct SOP document type, see Section 6.

5 Results

We present the results obtained by SOP-MRC on two different message databases. The VP-02 database was used for training the POS Tagger and to define the classifier rules. It contains 2328 messages and 37 document types. The TT-01 database contains 3131 messages and 18 document types. The detailed results for each document type, for both VP-02 and TT-01, are given in Appendix 1 and 2.

The first part of Table 6 shows the overall results of SOP-MRC for VP-02. Since over 75% of the messages are "Free Text", we also show the microaverage result without the "Free Text" category to give a better indication of performance.

The second part of Table 6 shows the overall results of SOP-MRC for TT-01. In this corpus, over 85% of the messages are "Free Text" and the microaverage result is again given without the "Free Text" category.

Recall and Precision Averages	Recall	Precision
VP-02		
Macroaverage	79.99%	67.94%
Microaverage	82.49%	81.52%
Microaverage (no Free Text)	70.53%	43.41%
TT01		
Macroaverage	12.72%	13.09%
Microaverage	86.77%	83.88%
Microaverage (no Free Text)	77.39%	26.81%

Table 7 SOP-MRC for VP-02 and TT01

These results are very encouraging. Although the macroaverage for TT-01 is not very good, this is explained by the fact that there were a number of False Positives for document types which do not occur in this data (see Appendix 2). The microaverage shows that the document types with larger numbers of documents are giving as good results for the new unseen data as for VP-02.

An explanation for the discrepancies between the document types used in VP-02 and in TT-01 is that the SOP definitions were actually developed at DJFHQ, and that VP-02 was a military exercise which only involved that headquarters, with all the messages coming from DJFHQ, while TT-01 was a four nation exercise, with messages coming from a number of different headquarters.

Another issue concerns the "Shift Handover" documents. The Shift Handover form is filled in by officers "handing over" their shift to another officer who "watches" the database for outstanding issues, and is a summary of the past 12 or 24 hours. Although this form is essentially free text, because the officers tend to think in terms of formatted documents, they often write it as another formatted document, eg. with numbered items for new paragraphs. If we classified the Shift Handover form as "Free Text", the accuracy would improve. This can be seen as another example of the well-know fact that the operator or human element is a large factor in system success.

6 Multilabel classification

One of the lessons from our earlier work was that we needed to use a multilabel classification rather than a simple multiclass classification. In multiclass classification, each message is assigned to only one of several possible classes, while in a multilabel classification, a message can be assigned to one or more classes (Lewis, 2002). Our

new classifier rules now perform a type of multilabel classification, by assigning a complex label to each message. An example of this complex label is shown in (6).[3]

(6) C:B:A:Free Text

This example shows the output of a message that contains a document of type C. As mentioned above, some rules are actually subsets of other rules, thereby defining a hierarchy of document types. In this case, the rule for document type C includes the rule for document type B, which includes the rule for document type A; thus A and B are also included in the complex label, as well as Free Text, the default classification.

In our current implementation, we choose the label returned by the more specific rule in the complex label, and return it as the single label, or multiclass classification, for the message being classified (in this case, C). Although Sebastiani (2002) argues that a multilabel classifier cannot be used as a single label classifier, the complex label that is returned by our classifier component is in fact a multilabel classification in terms of the hierarchical structure of the classifier rules. This hierarchy can be thought of as a set of binary classifiers (implemented as classifier rules) listed in order, from smallest (more general) to largest (more specific). This set is ordered such that a document type is a subset (in terms of structure, not content) of the next document type.[4]

7 Conclusions and future work

7.1 Improvements

The expanded POS tagset provides a better coverage for the texts in our domain, and the POS Tagger component is now trained for the real data found in military message databases.

The new classifier component is much cleaner and more efficient. Python provides high-level methods to implement regular expressions and use them to search strings of text, which makes it easier to modify the classifier and to add or change the rules.

The results on a new database of messages, which were not used to create the classifier rules, are encouraging and indicate that we can improve the performance of SOP-MRC with little effort.

7.2 Integration with QBI

QBI is an improved search interface to the Lotus Notes operational log database used at DJFHQ, which is developed by the same DSTO team as SOP-MRC. We aim to incorporate the output of SOP-MRC with QBI by providing a category-pivoted view of the documents as categorised by SOP-MRC. An example of this view is shown in Fig. 5.

The categorised view will allow the users of the QBI to quickly find messages in a Lotus Notes operational log database by using document types to limit their search or to locate the relevant message.

7.3 Other Improvements to SOP-MRC

The performance of the POS Tagger could be improved by pre-processing the messages. Text such as "10 KM" could be normalised to "10KM" so the POS Tagger can properly tag it <DST> rather than <CD> <NN>. This would also help improve the classifier's performance.

The current implementation relies on a one-to-one correspondence between classifier rules and document types. We are looking at another approach, in which a classifier rule would be a subset of a number of rules for a few document types, in other words we would have a more general rule for a set of document types. This would correspond to implementing the true multilabel classification mentioned in Section 6, where the hierarchy of rules also correspond to the conceptual hierarchy of document types.

[3] The names of the document types have been replaced by alphabetical labels for presentation; in the real system, the categories have meaningful labels.

[4] For example, a MEDSITREP (Medical Situation Report) is conceptually a kind of SITREP (Situation Report), but a SITREP is not a kind of "Free Text", even though the rules for "SITREP" and "Free Text" are in a subset relation.

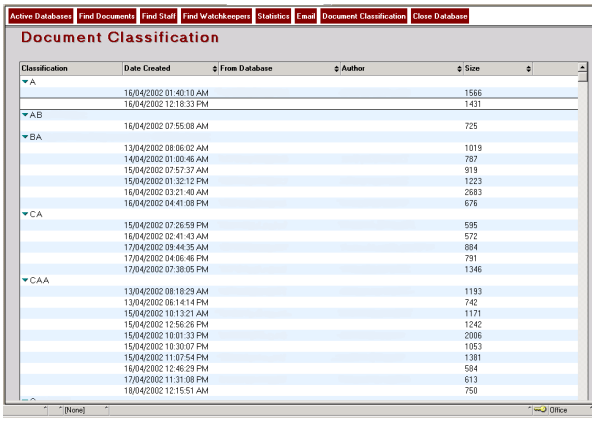

Fig. 5 Screenshot of QBI with SOP-MRC

If an incoming message matches the more general rule, it can then be tested against more specific rules. If the message fails to match one of these, then other methods can be used to determine the more specific document type, but at least the more general category can be kept, rather than defaulting to "Free Text", as is currently the case.

Further analysis to determine the more specific document type would involve the number of certain POS tags, the ordering of these tags or the absence of certain tags.

Another improvement concerns the addition of further POS tags for our domain, for example, tags for unit names and ranks. This information would also be useful in further work on both document classification and information extraction. For instance, it would allow extracting information about which units are involved or identify the personnel injured from NOTICAS messages.

We are also investigating the development of a trainable system, using approaches such as TF-IDF, Rocchio Method, Support Vector Machines or hybrid solutions such as Learning from Positive and Unlabeled text documents (Jackson and Moulinier, 2002; Joachims, 1998; Vapnik, 1995; Lee and Lui, 2003).

References

Carr, O. and Estival, D. (2002) Text Classification of Formatted Text Document. *Proceedings of the 5th Australasian Natural Language Processing Workshop*. Canberra. pp. 49-54.

Jackson, Peter and Moulinier, Isabelle (2002) *Natural Language Processing for Online Applications – Text Retrieval, Extraction and Categorization*, Vol. 5, Natural Language Processing, John Benjamin Publishing Company, Philadelphia.

Joachims, T. (1998) Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of the European Conference on Machine Learning*, Springer.

Johnson, D.E., Oles, F.J., Zhang, T., Goetz, T. (2002), A Decision-Tree-Based Symbolic Rule Induction System for Text Categorization, *IBM Systems Journal*, Vol. 41, No. 3, IBM Corporation.

Lee, W.S. and Liu, B. (2003) Learning with Positive and Unlabeled Examples using Weighted Logistic Regression. *Proceedings of the Twentieth International Conference on Machine Learning* (ICML-2003), Washington, DC, USA.

Lewis, David (2002). *Machine Learning for Text Classification Applications*, Tutorial. 40th Meeting of the Association of Computational Linguistics, University of Pennsylvania. Philadelphia, USA.

Manning, C.D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts.

Mason, Oliver (2003). *Qtag 3.1*. Department of English, School of Humanities, University of Birmingham, http://web.bham.ac.uk/O.Mason/software/tagger/

Sebastiani, Fabrizio. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, Vol. 34, No. 1, pp. 1-47.

Sutcliffe, Richard, Koch, Heinz-Detlev and McElligott, Anne (eds) (1996), *Industrial Parsing of Software Manuals*. Amsterdam, Rodopi.

Tufis, Dan and Mason, Oliver (1998). Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger, *Proceedings of the First International Conference on Language Resources & Evaluation (LREC)*, Granada, Spain, 28-30 May 1998, p.589-596.

Vapnik, V. N. (1995) *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Doc Type	Gold	Total	TP	FP	FN	Recall	Precision
A	1	2	1	1	0	100.00%	50.00%
B	1	1	1	0	0	100.00%	100.00%
C	10	6	6	0	4	60.00%	100.00%
D	2	5	1	4	1	50.00%	20.00%
E	14	10	3	7	11	21.43%	30.00%
F	1	2	1	1	0	100.00%	50.00%
G	6	5	4	1	2	66.67%	80.00%
H	1	1	1	0	0	100.00%	100.00%
I	2	2	2	0	0	100.00%	100.00%
J	1	3	1	2	0	100.00%	33.33%
K	11	18	7	11	4	63.64%	38.89%
L	33	16	9	7	24	27.27%	56.25%
Free Text	1846	1657	1625	32	221	88.03%	98.07%
M	3	2	2	0	1	66.67%	100.00%
N	19	28	8	20	11	42.11%	28.57%
O	2	4	2	2	0	100.00%	50.00%
P	164	391	131	260	33	79.88%	33.50%
Q	34	22	13	9	21	38.24%	59.09%
R	3	3	3	0	0	100.00%	100.00%
S	32	27	27	0	5	84.38%	100.00%
T	1	4	1	3	0	100.00%	25.00%
U	20	30	18	12	2	90.00%	60.00%
V	5	5	5	0	0	100.00%	100.00%
W	1	3	1	2	0	100.00%	33.33%
X	1	2	1	1	0	100.00%	50.00%
Y	5	2	2	0	3	40.00%	100.00%
Z	1	11	1	10	0	100.00%	9.09%
AA	1	1	1	0	0	100.00%	100.00%
BB	2	3	2	1	0	100.00%	66.67%
CC	1	1	1	0	0	100.00%	100.00%
DD	81	31	31	0	50	38.27%	100.00%
EE	3	4	2	2	1	66.67%	50.00%
FF	1	2	1	1	0	100.00%	50.00%
GG	1	1	1	0	0	100.00%	100.00%
HH	1	1	1	0	0	100.00%	100.00%
II	13	19	8	11	5	61.54%	42.11%
JJ	4	3	3	0	1	75.00%	100.00%
Total	2328	2328					

Appendix 1 SOP-MRC on VP02

Doc Type	Gold	Total	TP	FP	FN	Recall	Precision
A	0	1	0	1	0	0.00%	0.00%
B	0	1	0	1	0	0.00%	0.00%
C	1	0	0	0	1	0.00%	0.00%
D	2	0	0	0	2	0.00%	0.00%
E	10	11	0	11	10	0.00%	0.00%
F	0	1	0	1	0	0.00%	0.00%
G	0	1	0	1	0	0.00%	0.00%
J	0	2	0	2	0	0.00%	0.00%
K	0	14	0	14	0	0.00%	0.00%
L	26	10	4	6	22	15.38%	40.00%
Free Text	2767	2493	2471	22	296	89.30%	99.12%
M	0	1	0	1	0	0.00%	0.00%
N	5	26	3	23	2	60.00%	11.54%
O	2	2	2	0	0	100.00%	100.00%
P	176	521	167	354	9	94.89%	32.05%
Q	57	0	0	0	57	0.00%	0.00%
S	5	1	0	1	5	0.00%	0.00%
T	0	3	0	3	0	0.00%	0.00%
U	0	1	0	1	0	0.00%	0.00%
V	1	0	0	0	1	0.00%	0.00%
W	0	3	0	3	0	0.00%	0.00%
Y	0	1	0	1	0	0.00%	0.00%
Z	0	7	0	7	0	0.00%	0.00%
AA	14	0	0	0	14	0.00%	0.00%
CC	1	0	0	0	1	0.00%	0.00%
DD	50	10	1	9	49	2.00%	10.00%
EE	2	16	0	16	2	0.00%	0.00%
FF	1	0	0	0	1	0.00%	0.00%
II	3	1	0	1	3	0.00%	0.00%
JJ	5	1	1	0	4	20.00%	100.00%
Total	3128	3128					

Appendix 2 SOP-MRC on TT01