Machine Learning based Text Analysis for Intelligence Collation

Claire Thie, Zoe Lock, Darrell Smith, Emma Cribb, Ashley Ford and Andrew Rankin
QinetiQ Ltd,
Malvern Technology Centre,
Malvern, Worcs.,
WR14 3PS, UK.
e-mail: <cjthie, zplock, djsmith6, ecribb, aford, ajrankin>@qinetiq.com

Abstract—This paper presents the initial findings of an investigation into automated support for fusing textual information and non-text data from different sources within the military domain. Our aim is to develop a prototype fusion capability that can categorise, fuse and present intelligence information to military officers. To date, our focus has been on text information categorisation using both existing text classification techniques (e.g. Weighted Feature Vector (WFV) classification) and machine learning algorithms based on Inductive Logic Programming (ILP) and natural language processing techniques. The algorithms have been used to automatically assign documents to a pre-existing set of categories by correlating text within the documents to text relating to classifications/categories. Once text documents are categorised according to their content, then fusion can begin. The results of initial experiments indicate that the ILP approach performs at least as well as the WFV technique, and outperforms it in one set of experiments. Both techniques have the scope for further improvements which we outline in this paper.

II. THE FUSION PROBLEM DOMAIN

Given the huge scope for information and intelligence fusion within the defence domain, the requirements capture was initially limited to reviewing text fusion processes within a joint operational environment [1] through available documentation and interviews with the user community. The aim was to identify current text fusion processes undertaken within military operations and intelligence domains. The result of this phase was the identification of domain-specific intelligence processes, the significant business events that drive these processes and the information used and produced during process enactment. This was followed by the production of high level business process models to aid the identification of opportunities for a fusion prototype within the operational intelligence domain. These opportunities correspond to manual fusion activities or where the novel application of automated fusion would offer benefit. Within operational intelligence the following definitions are used (see [1]):

- **Intelligence.** “The product resulting from the processing of information concerning foreign nations, hostile or potentially hostile forces or elements or areas of actual or potential operations.”
- **Information.** “Unprocessed data of every description which may be used in the production of intelligence.”

where information (intelligence sources) and intelligence products are processed to meet specific requirements. There are three principal sources of intelligence (i.e. unprocessed or processed information): controlled (e.g. military intelligence, specialist agencies); uncontrolled (e.g. media, technical journals); casual (e.g. defectors, refugees, Non-Government Organizations).

The high-level operational intelligence cycle consists of the following four steps:

1) direction - determine intelligence requirements and collection management;

2) collection - the exploitation of sources of information and the delivery of this for processing;
Fig. 1. The intelligence processing lifecycle

3) processing - production of intelligence (outlined below);
4) dissemination - timely delivery of appropriate intelligence product to those who need it.

Within this research the identification of the high-level requirements for integrated text analysis involved the mapping of text fusion capabilities against the processing phase of the intelligence cycle as illustrated in Figure 1, which shows the typical steps undertaken by a military officer. The activities comprising the intelligence processing are:

- **Collation** - The grouping together of related items of information or intelligence to provide a record of events which facilitates further processing. In practice, this step is made up of the procedures for receiving, grouping and recording all reports at any level. This constitutes a categorisation task.
- **Evaluation** - An appraisal of an item of information in respect of the reliability of the source and the credibility of the information.
- **Analysis** - Subjecting information to review in order to identify significant facts for subsequent interpretation.
- **Integration** - Analysed information or intelligence is selected and combined into a pattern in the course of the production of further intelligence. In essence, this is a summarisation task.
- **Interpretation** - The significance of information or intelligence is judged in relation to the current body of knowledge.

As indicated in Figure 1 there are two points identified within this process where the application of information fusion would provide benefit - during the collation and integration steps. In the collation step, textual (and other) items need to be categorised according to which text and non-text items should be grouped. This is a text classification problem. During the integration step, the items of interest need to be fused into a coherent report that can be easily interpreted. This is essentially a summarisation task. The work described here aims to identify which text and non-text information should be collated for fusion. This paper concentrates on this content-based categorisation problem that occurs at the collation phase of the intelligence processing cycle.

### III. TEXT CLASSIFICATION

Text classification is the task of assigning text objects (typically electronic documents) to categories or classes according to the subject of the topic of the text and is a relatively mature research area. The assignment of text to the different classes is typically achieved through the use of predefined rules or by learning statistical models of the text describing topics associated with the classes.

The majority of text classification tools automatically create a statistical model for each of the classes. The models are then used to predict whether new documents fit in a category or class. In order to form the models of the classes, a set of training examples, consisting of a number documents labelled with the class(es) to which they belong, is used. A model is created for a category by extracting the statistics of the text, or features derived from the text, in the documents labelled with that class. The models obtained with these techniques are highly effective at text classification and are produced in a fraction of the time taken to manually produce a set of rules.

The Vector-Space (VS) model and Naïve Bayes (NB) classifiers are two well known techniques in the text analysis domain that build statistical models in order to perform text classification. Both techniques treat each text item as a set of words and disregard other attributes such as grammatical structure or relationships between words, so that the order in which the words appear does not influence the item’s classification.

The VS model represents each piece of text as a feature vector where the features are the words in the text (typically after all stop words - frequently occurring words such as a, the, and, of, and so on - have been removed). A statistical model for each class is built up by weighting the features according to how well they characterise the class, using a statistical metric such as $\chi^2$ [2]. The top scoring $n$ features can be selected (and their scores normalised) to create an initial feature vector. The weights can then be updated using learning (see [3]). The classification of a new instance (i.e. piece of text) is determined by the value of the sum of the weights of the $n$ features (i.e. words) that are present in the new instance. We refer to this technique as Weighted Feature Vector (WFV) classification.

The NB classifier is a simple but efficient classification algorithm based on Bayes’ theorem. It counts both the number of times each word occurs within each class in the training texts and the number of training examples belonging to each class. It then builds a density model that is used to identify the classes to which new text items belong. In order to simplify the calculations involved, it makes the assumption that the word positions in the pieces of text are independent of each other. Whilst this assumption seems inappropriate for text, NB classifiers can work well in practice.

### IV. INDUCTIVE LOGIC PROGRAMMING FOR TEXT CLASSIFICATION

Machine learning (ML) techniques are particularly good at automatically identifying patterns in data and have recently been applied to text classification problems (see [4]). Inductive logic programming (ILP) is a technique that has a number of benefits that other machine learning methods do not [5]:

- the output is comprehensible (in the form of IF-THEN rules) which provides an audit trail for decisions made and, potentially, insight into the problem being solved;
- it can learn complex structural and relational concepts;
- it can make use of existing expert knowledge.
ILP has been developed to solve non-text classification problems: for example it has been successfully used to characterise mutagenic chemical compounds [6] and to classify aircraft types from their outline extracted from an image [7]. Text processing is a hard problem due to the complexities of natural language. However the significant success of ILP in complex non-text domains makes it a viable candidate for the text classification problem. A small number of text classification experiments have investigated the use of academic implementations of ILP algorithms. Some notable examples are discussed later in this section.

ILP can be used to learn complex concept descriptions from training examples within a logical framework. It expresses a concept description as a set of first-order logic IF-THEN rules, normally written in Prolog. The IF part of a rule states the conditions that need to be satisfied in order for the predicate in the THEN part to be true. The ability to include expert domain knowledge in the ILP learning process makes it easy to exploit known general features of natural language such as relationships between words (e.g. synonyms, hypernyms) and grammatical structure in the learnt rules. A set of rules for the same predicate states the different sets of conditions that could be satisfied for the concept to apply (just one rule needs to be satisfied). Inducing (or learning) a rule set involves exploring the search space of all possible rule sets in a logical fashion. This is obviously preferable to an exhaustive search, which would be computationally expensive or infeasible.

As a supervised ML technique, ILP takes pre-classified training examples as input. Most ILP systems accept training examples classified as positive and negative instances of a single concept. To illustrate these issues, consider the case of predicting students’ grades based on their school reports. This will require a set of classification rules that characterise the general properties of a school report belonging to students that are to get particular grades (e.g. A to G and Fail) in particular subjects. In this case the training set will consist of school reports that have been labelled with the grades achieved in the subjects that have been taken by the students. A student’s school report will be made up of symbolic information such as the age, and sex of the student and a number of free text fields, one for each subject taken by the student containing descriptions of the student’s progress in that subject written by the teacher of that subject. The report may also contain a field of free text written by the student’s head teacher, noting their general observations of the student’s behaviour.

A particular learning task in this domain could be to use ILP to learn to characterise school reports of students that are likely to obtain an A grade in the subject of physics. For this task the positive examples would be all those reports in the training set labelled with A-Physics. All the other reports in the training set would be negative examples.

Before learning it is typical to pre-process the data to convert all words to lower case and to remove punctuation and stop words. It is also common to perform stemming to convert words into their root form (e.g. studying and studies into study). At this point the reports would be converted into a Prolog database, where the fields of the data set are kept separate, and each record labelled with the appropriate class.

The conditions that can appear in a rule for a particular ILP application must be defined in advance of learning because these are the constructs for rule building. This set of pre-defined conditions is known as background knowledge. These can be either expressions of application independent general relationships (such as numerical or geometric relationships) or could be application-specific concepts such as those described below:

- **RepContains(Subject,ReportId,Word):** This is a relation that returns true if the word Word is contained in the particular Subject field of the report with id ReportId. For example the condition RepContains(math,ck1234,well) would return true if the word well is contained in the mathematics field of the report with id ck1234. Note that Word can be a constant value or a variable.
- **isWord(Word,Const):** This relation returns true if the word referred to by the variable Word has the constant value Const. See the example rule below for an illustration of its use.
- **synonym(Word1,Word2):** This relation returns true if Word1 is a synonym of Word2.
- **near(Word1,Word2):** This relation returns true if the words Word1 and Word2 are near to each other (e.g. within three words of each other) in the text.

In addition to the relations in the background knowledge, the constants that may appear in the relations in the rule need to be specified. These may be specified by a domain expert and/or they may be extracted from the text in the training examples.

The main goals of an ILP engine are to derive a rule set that:

- (a) covers the training examples correctly, i.e. classifies the positive examples as positive and the negative examples as negative and (b) is general enough to correctly classify unseen (i.e. not contained within the training example set) positive and negative examples. Achieving the second goal means that future examples (e.g. school reports) can be categorised reliably by the rule set as being positive (e.g. potentially belonging to a student that will achieve a particular grade in a particular subject) or not. Typically some level of assurance is provided by executing the rules on a test set of examples (none of which appear in the training set). The pre-classification of each test example (allocated in the same way as the training examples) is known, so the classification granted by the learnt rule set can be assessed as being correct or not.

The rules are formed by iteratively adding new relations from the background knowledge and refining the variables to take the same value or to become a constant value. At each stage of the rule construction, candidate rule refinements are tested on the training data so that only the most promising rules refinements are taken forward to the next iteration. This process continues until a rule exceeds the maximum specified length or the performance of the rule on the training set has reached the desired level.

An example of a simple rule that might be learnt by an ILP system to characterise the school reports of students that are likely to obtain an A grade in the subject of physics is given below.

Note if a student was not studying mathematics and did not have a mathematics field within their school report, then the condition would fail.
In English this rule states that a student is likely to obtain an A grade in the subject of physics if their school report contains a synonym of the word good in the text of the mathematics field, and a value that is greater than 90 in the text of the physics field. In addition the headteacher’s field contains the words hard and work near to each other within the text.

Examples of the use of ILP in text classification problems include Cohen’s use of the ILP system FOIL to learn text classifiers that depend on word orders [8]. The background knowledge consisted of the words’ positions in the documents and various relationships based on word position such as: successor to express that one word is the successor of another in the document; near<1,2,3> to express that a word is within 1, 2, or 3 word positions of another; after to express that a word appears at some point after another in the document. This representation was compared to a propositional representation, where just the words and their positions were provided to FOIL. Both the propositional and relational representations had low performance for some problems, but the relational representation was statistically significantly better overall. The relations associated with the frequently occurring words were then discarded from the background knowledge, significantly improving the results obtained, outperforming existing propositional systems.

Junker et al. (see [9]) use ILP for both information extraction and text classification. They have their own ILP system that has been biased to the task of learning from text. It makes use of three relations: wordpos is used to describe words and their position in the text, fragment is used to provide the location of word sequences within a larger text, and next provides a mechanism for comparing the distances between words within the text. For text classification, an example is a list of words labelled as positive or negative. When the wordpos relation is used, the subset of the words in the text having minimum positive correlation to the positive examples are selected to be used as its arguments. Refinement operators also look at the occurrence of words to the left or right of a word that is already in the current rule. The WordNet thesaurus has been used as background knowledge in learning classification models for the Reuters data set based only on the text in the titles of the documents.

QinetiQ has developed an ILP system called IndicEz that is robust to noisy data and readily applicable to a wide range of real-world learning applications (including object recognition, control rule learning). It can learn from numeric, symbolic and text data and can automatically apply range constraints to attribute values in rules, if appropriate. Unlike other ILP systems, IndicEz can efficiently deal with large datasets through the use of a number of mechanisms to reduce the search space. IndicEz also makes use of special constant handling technology and is not reliant on the order in which the examples are presented. In this paper we investigate the potential of ILP, through the application of IndicEz, for the problem of text classification to support information fusion. Throughout the remainder of this paper the use of ILP refers to the use of IndicEz.

V. EXPERIMENTS AND RESULTS

In order to evaluate the potential of the proposed techniques for the text classification problem to be addressed, a number of experiments were carried out. The particular data sets used during the experiments are real-world data sets consisting of a large number of hand-labelled records about people. Each record consists of a number of different fields containing symbolic (non-text) values and paragraphs of free text. The data sets contain a large number of classes, with each record being labelled with a single class. The data sets were noisy in that they contained spelling mistakes, duplicated records and missing values/paragraphs.

Although the data sets used here have been taken from a non-military domain, they exhibit a number of similarities to the target military problem:

- The reports are written using a specialist vocabulary.
- The fields of text that form the reports are written by different people using different styles and language.

A. COMPARISON OF TECHNIQUES

For an initial proof-of-concept, the three proposed techniques (WFV, NB and IndicEz) were each used to learn a model to distinguish the category with the largest number of records belonging to it from the next three largest categories in the first data set. This led to a challenging learning problem with a highly skewed data set, with a small number of reports belonging to the category of interest, as compared to the number of reports that did not.

In order to learn the category models the records were pre-processed into a format that could be used as training examples by the techniques. First, the stop words and punctuation symbols were removed from the text. For the WFV and NB techniques, the free text in the different fields of a report were combined into a single vector of words, which were labelled with their associated class. For IndicEz, the reports were converted into Prolog database entries, where the fields of the data were kept separate, and each record labelled with the appropriate class. The \( \chi^2 \) metric was used to identify the top 50 scoring words in the training text to be placed in the background knowledge to be used as constants in the learnt rules. These words were augmented by a small number of words identified as related to the target class. In addition, a public-domain thesaurus containing synonym and parent/child relationships between words related to the domain was included as background knowledge for IndicEz.

The WFV technique takes the number of features to be used in its feature vector as an argument. This argument was allowed

\[^2\text{This could be referring to high test results.}\]
to take the values of 1, 3, 5, 10, 30 and 50 in all experiments. The results of the best performing configurations have been reported.

In order to assess the performance of the algorithms on the problem, a well-established machine learning evaluation technique, \textit{n-fold cross validation}, was used. \textit{N}-fold cross validation involves splitting the training examples into \textit{n} sets of a similar size. Each of the \textit{n} sets is removed in turn, to be used as a test set for the model learnt from the remaining \textit{n-1} sets. The results of each of the \textit{n} folds are then averaged to give an indication of the performance of the algorithm on future unseen data. In the initial experiments WFV and IndicEz achieved very promising results with an accuracy of 93\% and 94\% respectively using 10-fold cross-validation. The results for NB were disappointing scoring only 55\%. It is thought that the poor performance of NB can be attributed to the fact that NB models the distribution of all the words in the training text and some of the less important words may have been dominating the probability calculations. It is possible that improved results may be achieved by extending the set of stop words removed during the pre-processing stage. However, at this point NB was abandoned allowing us to focus on comparing the performance of the WFV and ILP techniques.

Further experiments comparing WFV with IndicEz were then carried out. The twenty most frequently occurring classes in the data were selected for characterisation. For each class to be characterised, the records labelled with that class were considered to be positive examples. All other records in the remainder of the data set were then the negative examples. The positive and negative examples were combined to form a data set for the problem. The data set was pre-processed for WFV and IndicEz as described above. The results, in terms of the accuracy, from these experiments are presented in Table I. As before, the results have been obtained using \textit{n}-fold cross-validation\(^3\).

The results are also presented in the form of Receiver Operating Characteristic (ROC) graphs in Figures 2 and 3. The ROC graphs plot the percentage of the positive test examples that were correctly classified, or True Positives (TPs), against the negative test examples that were incorrectly classified, or False Positives (FPs). Traditional measures of performance such as accuracy can be misleading when learning from a skewed distribution [10]. If a data set contains more instances belonging to one class than another class, and is used by a machine learning system to learn a model that maximises accuracy (or minimises the error), the resultant model will be skewed to avoid errors on the class with the larger number of examples associated with it. The effect on some learning algorithms can be so extreme that they will predict that all new instances should be classified as the majority class. Whilst such a model can be highly accurate, it is of no practical use at all. Plotting the TPs against FPs provides further insight into the practical use of the learnt models. Different problem scenarios will have different requirements in terms of the TP to FP ratio. ROCs allow the different ratios to be visualised, and an appropriate algorithm set-up for the current scenario to be identified.

The aim is to achieve a high TP rate with a low FP rate for each class, putting all the points in the top left-hand corner. Whilst some excellent results have been achieved by both techniques for many of the classes, clearly some of the classes

---

\(^3\)Where \textit{n} is selected to be an appropriate number between 2 and 10 according to the number of positive examples in the data set for each class.
are easier to characterise than others. Neither of the techniques clearly outperforms the other in terms of accuracy. But the results differ in that all of the points plotted in the ROC graph of the results obtained by IndicEz (in Figure 2) are clustered towards the left hand axis, whilst those obtained with WFV (in Figure 3) are spread out across the graph. IndicEz uses a rich language to construct its classification rules which has allowed it to minimise the FP rate, whilst the WFV technique uses a very simple feature vector and could benefit from both further minimising the FPs and maximising the TPs in some cases.

The ILP algorithm, however, has scope for improved performance through the use of expert domain knowledge. As a proof of concept, two of the classes with the worst classification performance by IndicEz, class 7 and class 14 (indicated with an arrow in Figure 2), were selected. The models learnt by IndicEz for each of these classes have good overall accuracy, but on inspection of the ROC plots, they are too specific so that whilst few negative test examples are wrongly classified as positive examples, some positive test examples are misclassified as negative. This is thought to be due to the variation in language used to convey similar concepts in the reports.

Therefore, additional natural language processing such as specialised stemming and improved background knowledge specifying additional synonym and parent concept relationships were used. The ROC graph with the new results for class 7 and class 14 is presented in Figure 4 (the new results are indicated with arrows). The additional background knowledge and preprocessing has boosted the number of reports correctly classified as each of the classes (TP rate), but not at the expense of also increasing the number of reports that are incorrectly classified (FP rate), moving the results up into the top left-hand corner of the ROC plot, where we would like them to be.

B. Verifying Initial Results

In order to verify the initial results obtained using IndicEz and WFV, and to identify how well the techniques would generalise to future data sets, they were both applied to a second data set. This data set was from the same domain as the first data set, and the records were labelled with classes from the same classification system, but they were from a different district and contained some different fields of symbolic values and free text.

In this experiment, the five most frequently occurring classes in the data were selected for characterisation. These are the same as the top five most frequently occurring classes in the first data set. As before, for each class to be characterised, the records labelled with that class were considered to be positive examples. All other records were then the negative examples.

The positive and negative examples were combined to form a data set for the problem. The data sets were pre-processed for the WFV and ILP techniques as described above. The data set consisted of four months worth of data; it was decided to use the first two months worth of data to train and the second two months to test the algorithms. The results in terms of accuracy, TPs and FPs are presented in tables II and III.

As can be seen IndicEz has achieved some very good results across all classes (average 96% accuracy) compared to the WFV technique (average 65% accuracy). Both techniques achieved excellent results for class 4, but this required a very simple model to characterise the class. For all other classes, WFV performed poorly when compared to IndicEz, achieving very high false positive rates. With such a skewed data set this

4The WFV classifier consisted of just 3 features.

<table>
<thead>
<tr>
<th>Class</th>
<th>Accuracy</th>
<th>TP</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class1</td>
<td>97%</td>
<td>0.98</td>
<td>0.028</td>
</tr>
<tr>
<td>Class2</td>
<td>86%</td>
<td>0.97</td>
<td>0.241</td>
</tr>
<tr>
<td>Class3</td>
<td>97%</td>
<td>0.93</td>
<td>0.027</td>
</tr>
<tr>
<td>Class4</td>
<td>99.6%</td>
<td>0.98</td>
<td>0.003</td>
</tr>
<tr>
<td>Class5</td>
<td>98%</td>
<td>0.97</td>
<td>0.022</td>
</tr>
</tbody>
</table>
has impacted heavily on the accuracies achieved. WFV has been unable to find a weighted vector of words with which to create an accurate model of the class in these cases. The $\chi^2$ metric has been unable to identify some of the important words within the text. The higher performance of IndicEz is due to some of the important words being available as background knowledge. The WFV classification technique may therefore benefit from using an alternative to the $\chi^2$ metric.

Further investigation into the performance of IndicEz in characterising class 3 has revealed the possible misclassification of examples belonging to this class. On examination of the text in the records belonging to class 3, it is thought that some of the examples have been incorrectly hand-labelled as class 3, when in fact they should have been labelled with an alternative class. This issue is currently being investigated with the owner of the data. The transparency of the rules learnt by IndicEz is clearly a benefit here.

### VI. CONCLUSIONS

This paper has presented the initial findings of an investigation into automated support for fusing textual information and data from different sources. In particular, this paper has focused on the task of linking text items in preparation for fusion, so that a useful fusion product is produced. This is a text classification problem, where the aim is to classify documents into groups that should be fused to produce intelligence reports. During this work we have compared the performance of three candidate techniques to be applied to this task in a number of experiments.

The results of the initial experiments indicate that ILP performs at least as well as the WFV classifier, with both techniques outperforming the NB classifier for the first dataset. The ILP and WFV techniques have both produced excellent results for many of the individual text classification tasks, achieving an average accuracy of 95% and 93% respectively. These results were repeated by the ILP in further experiments on a second data set gaining an average accuracy of 96%. The results of the WFV techniques on the same data set were not as good, with an average accuracy of 65%.

The ILP technique has a number of features that have made it particularly suitable for this task, in particular, the ability to learn patterns in complex, relational information and the ability to exploit expert background knowledge such as word relationships (e.g. synonym and parent relationships) and grammatical information have boosted the performance of the technique in this application. Future work will continue to expand the set of semantic and grammatical relationships incorporated into the background knowledge and will further investigate the use of domain-specific stemming. Work will also continue to focus on the potential to boost the results achieved using the WFV classifier; other feature selection metrics and the use of phrases as features will be investigated. In addition, the development of the fusion prototype demonstrator system will continue, in order to assess the potential application of the techniques in the integration stage of the intelligence processing cycle.

### ACKNOWLEDGMENT

This work has been funded by UK MOD’s Data and Information Fusion (D&IF) Defence Technology Centre (DTC).

### REFERENCES