COMP4046 Statistical Natural Language Processing Assignment 1

Enoch Lau
SID 200415765

17 April 2007

Abstract

Text categorisation is the problem of assigning a topic or a theme to a document. The task at hand was to attempt the Computational Medicine Center's 2007 Medical Natural Language Processing Challenge[1], where participants classified radiology reports using ICD-9-CM codes. A baseline system was developed, and the effect of using additional features in the vector representing each document was investigated. In particular, we examined the effect of improved tokenisation, normalisation, word weightings, separation of word counts into the documents' constituent parts, and the inclusion of bigrams. These experiments were conducted using the C 4.5 decision tree in the Weka machine learning software[2], and investigations into the effect of some of the parameters of the classifier were also conducted. The highest F-score via ten-fold cross-validation obtained was 0.840. Finally, a comparison with the results obtained by using a Naive Bayes classifier is performed.

1 Introduction

In an increasingly computerised age, documents of all kinds are being stored in digital information systems, and it is important for these documents to be retrieved accurately, because it is impossible for a human to manually perform this process due to the volume of information. This is none more so than in hospital information systems, and we look at a particular area, namely, radiology reports.

In text categorisation, there is a training set of objects, which are typically assigned one class. Each object is typically associated with a vector of measurements, and this contains counts of such things as words in the texts. In this particular problem, we use the training data from the Computational Medicine Center's 2007 Medical Natural Language Processing Challenge, which consists of 978 radiology reports. The problem is non-typical, because each report may be tagged with more than one of the categories, which consists of ICD-9-CM codes. Each report consists of two sections of text, the "clinical history" and the "impression". For the class assignment, the documents were classified by three independent companies, and the gold standard was chosen (by the organisers of the challenge) to be the majority annotation.

In the following sections, we first discuss the series of features used in the experiments, and provide descriptions of how they are produced and the reasoning behind their inclusion. We then discuss the machine learners used, namely C 4.5 and Naive Bayes, and discuss some of the options available in the C 4.5 implementation in Weka. We then discuss the evaluation methodology, and tabulate results. We conclude with a discussion of results, which includes an analysis of the areas of the system with potential to improve.

2 Types of features

A vector of word counts is typically used to represent a document in text categorisation. However, there are varying ways of including the word counts in the vector, and there are also different things to count apart from just words.

[1] http://www.computationalmedicine.org/challenge

[2] http://www.cs.waikato.ac.nz/ml/weka/

2.1 Baseline

The baseline system constructed consists of counts of tokens, created by splitting the text on whitespace. Three ways of including word/token count or the presence of a word/token in the document were investigated throughout the experiments:

- Relative frequency: each element in the vector is calculated as the relative frequency of the corresponding token; that is, it is the number of times the token appears in the document divided by the number of tokens in the document. It is a relative frequency to avoid the dominance of longer documents, although an inspection of the medical documents in the training data shows that it does not vary much in length.

- Log weighting: each element, s_{ij}, in the vector $x_j = (s_{1j}, \ldots, s_{Kj})$ (where K is the number of features) is calculated by way of the following formula:

$$s_{ij} = 10 \times \frac{1 + \log(tf_{ij})}{1 + \log(l_j)}$$

  This is identical to the presentation in the textbook[3], with the exception that rounding is not necessary, as the numbers are not designed to be presented on the written page. The justification for the inclusion of this measure is that more as words become more frequent in a document, their importance does not increase linearly with the number of times they appear.

- Binary: a 1 is recorded if the word is present in the text, otherwise a 0 is recorded. This was chosen because binary features improve computational efficiency, and we wanted to see if we could achieve the same or better performance in less time.

Each of these weighting schemes was carried through each of the following experiments, as types of features were added, thus resulting in a matrix of results.

[3] *Foundations of Statistical Natural Language Processing*, C. Manning and H. Schütze, page 580

2.2 Changes in tokenisation and the removal of stop words

After the baseline was constructed, the next experiment was to improve the words that were included in the feature vector. Firstly, tokenisation was changed from simply splitting the text on whitespace, which is undesirable because punctuation is often remains attached to words. The tokenisation was changed to use the regular expression "$(\backslash w\backslash.) + |(\backslash w + \backslash'\backslash w)|\backslash w+$"; this captures abbreviations, words with apostrophes and regular words. The words were then all converted into lower case, on the assumption that words have the same meaning regardless of capitalisation (this is not always true, for example, "AIDS" and "aids", but there are few such abbreviations). Finally, stop words were removed; a list of English stop words was obtained from the Internet[4].

2.3 Stemming

The next experiment altered the features such that words are stemmed using the Porter stemmer. This is to improve the distributional characteristics of the words, because the doctor may refer to the same concept but in a different part of speech, for example. Although the Porter stemmer often does not, by empirical observation, produce sensible English words, this is not a problem; the words are not designed to be read by a human and are only used for compiling statistics on the documents.

2.4 Weighting medical terms

The next experimented weighted medical terms more heavily than other words. The rationale for this is that medical words are more important than other words, because we are classifying medical documents. A list of medical terms was obtained from the Internet[5]. The weighting was performed after the text has been tokenised and made lowercase; any tokens identified as being in the list of medical terms was duplicated.

[4] `http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words`

[5] `http://users.ugent.be/~rvdstich/eugloss/EN/lijst.html`

2.5 Separating the text sections

The documents contain two sections: "clinical history" and "impressions". Previously, we treated these two sections as one document, performing word statistics without regard to which section the words came from. However, the words used in these two sections may differ and provide additional information that was lost by combining the two sections. This experiment includes word statistics separately for the two sections.

2.6 Bigrams

Often, medical terminology is not expressed in terms of a single word. For example, "lymph node" is a single concept, and the feature vector may wish to somehow acknowledge the existence of these bigrams as being of particular importance. Bigrams were formed after tokenisation, changing to lowercase and stemming, but before the removal of stop words. This was to prevent the creation of bigrams that did not actually exist in the original document. However, no bigrams including stop words were included. The bigrams that were used as features were those that were statistically significant. This was determined by using a t-test with a 99.5% confidence:

$$\frac{p(w_1, w_2) - p(w_1)p(w_2)}{\sigma/\sqrt{n}} > 2.576$$

The standard deviation, σ, was approximated by $\sigma = p$, because we have a binomial distribution and $\sigma = p(1-p) \approx p$ for small p. In the vector, the bigrams were evaluated according to relative frequency, log weighting or binary $\{0, 1\}$ as per the unigrams.

3 Machine learners

The Weka machine learning software contains a variety of classifiers, and the two that were used were the C 4.5 decision tree and the Naive Bayes method. However, for the experiments determining which features were useful, only the C 4.5 decision tree was used. The Naive Bayes method was used only for comparison purposes, and was run on the features that produced the highest F-score with the C 4.5 classifier with default settings (which turned out to be taking the binary existence of words, excluding bigrams).

Again taking the features that produced the highest F-score, we hold this constant while experimenting with various options in the classifier. The following options, all to do with how pruning is performed, were experimented with:

- Reduced-error pruning: the tree was pruned using alternative algorithm for pruning (using the -R flag)
- No pruning: the tree was not pruned (using the -U flag)
- Pruning with different confidence levels: the lower the confidence level, the more pruning nodes that are pruned; the default value is 0.25, and we tested 0.05, 0.1, 0.2 and 0.3 (using the -C flag)

Pruning is an important step in the creation of a decision tree, to avoid over-fitting or under-fitting the data. Although the Weka implementation uses a hold-out set to prune the tree [check], the options allow one to tweak how the process is performed for optimum results.

4 Method

4.1 Procedure

As outlined above, a series of experiments involving the addition of new types of features were performed. At each stage, we perform a ten-fold cross-validation with the training data provided, and obtain an F-score as outlined below. If at any stage, the F-score is lower, the type of feature just introduced is not carried through to following experiments; otherwise, the types of features accumulated in the order of description.

To investigate the effect that the choice of classifier has on the result, the best performing set of features from above was selected and run through the Naive

Bayes classifier and the C 4.5 classifier with changes to pruning options.

4.2 Gold standard

As described in the challenge specification, the gold standard was the majority annotation. This is justified because a priori, there is no reason to trust the annotation of one company over the others, and thus, a democratic solution was chosen.

4.3 Micro averaged F-score

Typically, in the single-label case, the F-score is the harmonic mean of precision (P) and recall (R), and is used to balance the presence or absence of a binary feature. However, because this is a multi-label situation, the F-score is adapted, as per the challenge specification, to be based off the micro averaged contingency table, which is the element-wise sum of the contingency tables over all the ICD-9-CM codes available (Figure 1).

The precision, recall and F-score are then calculated by:

$$\begin{aligned} P &= \frac{A}{A+B} \\ R &= \frac{A}{A+C} \\ F &= \frac{2PR}{P+R} \end{aligned}$$

Micro averaged F-scores give equal weighting to each code assignment event, unlike macro averaged F-scores, which give equal weighting to each code. If we make the assumption that the distribution of codes in the training data is representative of real-world data, then the micro averaged F-score gives a better indication as to how the classifier will perform with live data.

5 Results

5.1 Variation of features

We first present the results of the experiments involving different features using the C 4.5 classifier, and with the varying measures of word count (Figure 2). The best F-score achieved is 0.826 by taking the text in separate sections (along with previous feature types), and by taking the existence of the features as a binary $\{0, 1\}$.

5.2 Naive Bayes

As described above, we took the best performing set of features as determined above (up to taking separate sections with the binary existence of features) and compared the results from the default C 4.5 classifier and the default Naive Bayes classifier (Figure 3). The C 4.5 classifier, with default settings, far out-performs the Naive Bayes classifier, with default settings.

Classifier	F-score
C 4.5	0.826
Naive Bayes	0.497

Figure 3: The F-scores comparing the performance of C 4.5 and Naive Bayes on the feature set of separate sections with the binary existence of features

5.3 C 4.5 settings

We again took the best performing set of features with the default C 4.5 settings and attempted to improve settings by adjusting the settings of the classifier (Figure 4).

Flag	Effect	F-score
(default)		0.826
-R	Reduced-error pruning	0.813
-U	No pruning	0.811
-C 0.05	Confidence level = 0.05	0.840
-C 0.1	Confidence level = 0.1	0.839
-C 0.2	Confidence level = 0.2	0.833
-C 0.3	Confidence level = 0.3	0.827

Figure 4: The F-scores for different settings of the Weka C 4.5 classifier involving

	Code in Gold	Code not in Gold	Total
Code in Guess	$\sum_{Code} A$	$\sum_{Code} B$	$\sum_{Code}(A+B)$
Code not in Guess	$\sum_{Code} C$	$\sum_{Code} D$	$\sum_{Code}(C+D)$
Total	$\sum_{Code}(A+C)$	$\sum_{Code}(B+D)$	$\sum_{Code}(A+B+C+D)$

Figure 1: The micro averaged contingency table

	Relative frequency	Log weighting	Binary $\{0,1\}$
Baseline	0.640	0.643	0.763
Change of tokenisation	0.768	0.761	0.816
Stemming	0.771	0.770	0.818
Weighting medical terms	0.773	0.772	0.818
Separate sections	0.787	0.787	0.826
Bigrams	0.800	0.800	0.825

Figure 2: The F-scores for the results of varying the features used to represent each document. The rows correspond to types of features described in section 2.

6 Discussion of results

The results in Figure 2 indicate that for the relative frequency and log weighting measures, the addition of new feature types (change of tokenisation, normalisation, weighting of medical terms, and the use of bigrams) improves the F-score. With the binary $\{0,1\}$ measure, the final addition of significant bigrams does not improve the F-score. Although the gradual improvement of the features allows us to utilise all the improvements incrementally to achieve the best F-score, we cannot assume that the features listed always have the same effect on the F-score as in Figure 2. This is because the types of features are not independent and taken together, and different combinations of the types of features may affect the F-score negatively.

The notion of a word could be improved. Firstly, although bigrams are taken later on, time phrases (such as 9 months) were quite common in the text, and as diseases are sometimes age-specific, utilising this information better (for example, placing these phrases into bins) should be explored. Medical terminology often contains much jargon in the form of acronyms, and while we used a list of medical terms to boost their prominence, there is no correlation between the acronym and its expansion – both of which may appear in the text. Acronym expansion may assist in this case. In addition, the list of words used for stop words and medical words were simply downloaded from the Internet, and were not tailored for the task; some general stop words may be important in the medical field, while the language of radiology reports may have more specific language than medical words in general. A possible further experiment is to remove all words except for medical words. As for taking significant bigrams as a type of feature, different confidence levels in the t-test may allow for improvements in the final result.

Clearly, for this type of data, the Naive Bayes classifier performs significantly worse than the C 4.5 classifier, but the results are only valid for the default settings of these two classifiers. However, this result does show that the result is dependent on the choice of classifier.

The two options for the C 4.5 classifier, reduced-error pruning and no pruning, did not improve the result; in fact, it appears that they may be detrimental to the result. Obtaining the correct pruning is something that can be tweaked in the C 4.5 classifier, and the results show that heavier pruning tends to gain better results. This may show that the default confidence level in Weka results in an overfitted tree. Further experiments involving a finer-grained analysis of the confidence level may further improve the result.

In general, however, the main limitation of the re-

sults presented is that the F-score presented is likely to be higher than the F-score obtained with a test data, because, for instance, the values of the C 4.5 pruning confidence interval have been tuned to give the best results via ten-fold cross-validation on the training data.

7 Conclusion

We have taken the 2007 Medical Natural Language Processing Challenge training data and trained it using a variety of combinations of types of features, and with two different classifiers. We found that, in general, the chosen types of features affect the F-score positively, to varying degrees. The Naive Bayes classifier was found to perform significantly worse than the C 4.5 classifier. For the C 4.5 classifier, the reduced-error pruning algorithm and the lack of pruning had a negative impact on the F-score, but altering the pruning confidence level allowed us to further improve the F-score to 0.840 with a confidence level of 0.05.