

Five-way Smoking Status Classification using Text Hot-spot Identification and Error-Correcting Output Codes

A. M. Cohen, M.D. M.S.

Department of Medical Informatics and Clinical Epidemiology

School of Medicine

Oregon Health & Science University

3181 S.W. Sam Jackson Park Road, Mail Code: BICC

Portland, Oregon, USA 97239-3098

Correspondence:

Aaron M. Cohen

Phone: 1-503-494-0046

Fax: 1-503-494-4551

Email: cohenaa@ohsu.edu

Abstract

Objective: We participated in the i2b2 smoking status classification challenge task. The purpose of this task was to evaluate the ability of systems to automatically identify patient smoking status from discharge summaries.

Design: Our submission included several techniques that we compared and studied, including hot-spot identification, zero-vector filtering, inverse class frequency weighting, error-correcting output codes, and post-processing rules.

Measurements: We evaluated our approaches using the same methods as the i2b2 task organizers, using micro- and macro-averaged F1 as the primary performance metric.

Results: Our best performing system achieved a micro-F1 of 0.9000 on the test collection, equivalent to the best performing system submitted to the i2b2 challenge. Hot-spot identification, zero-vector filtering, classifier weighting, and error correcting output coding contributed additively to increased performance, with hot-spot identification having by far the largest positive effect. The use of manually created post-processing rules did not improve performance.

Conclusions: High performance on automatic identification of patient smoking status from discharge summaries is achievable with the efficient and straightforward machine learning techniques studied here.

1 Introduction and Background

Automated document classification can be a powerful technique to aid biomedical researchers by reducing the human effort needed to make repeated decisions that categorize a sample of text. The samples of text are commonly called *documents* and may come from a large variety of sources, including but not limited to: abstracts, full text articles, medical record discharge summaries, and radiology or pathology reports. Automated document classification is useful when each of the individual samples from a given source need to be categorized into one or more of several pre-defined categories (also called *labels* or *classes*), and there already exists (or can easily be created) a set of documents that have been already categorized, known as *training data*, which is usually (but not always) created by human experts. In this situation, machine learning algorithms can be used to extract out a set of rules or procedures from the training data, and apply these rules to new, previously unseen documents, accurately predicting the labels that should be assigned to these documents.

Automated document classification tasks come in several types. The most elementary is the binary classification task where each document is assigned exactly one of two labels. Typically one of the classes is of interest and the other is not, and so in this case the labels are often simply termed “positive” and “negative”. The specific meaning of the labels is task-dependent. For example, “positive” may mean that a certain journal article is likely to be included in a systematic review of medical pain therapeutics.¹ Alternately, a “positive” label may indicate that the document should be sent to an expert biologist for annotation, who will then codify the new contributions to biomedical science contained in the paper by adding gene ontology (GO) terms² and protein-protein interactions³ to a database.⁴

Multiple classification tasks are also common. These can be either mutually exclusive or non-exclusive. With non-exclusive tasks, one or more of several labels can be assigned to each document. Commonly, this is treated as a set of independent binary classification tasks, one for each label. While this is often a good approach, the labels may in fact not be independent, and so better classification methods will exploit the relationships between the labels as well as the label-specific information. One example of this kind of task is automated assignment of ICD-9 codes to hospital admissions based on the text of the discharge summary and other items in the medical record. While each ICD-9 code can be treated as an independent label, the codes are certainly not independent and it may be possible to achieve better performance by taking into account the positive and negative correlations between labels such as by including information concerning which pairs of labels may and may not be assigned simultaneously to the same admission.⁵⁻⁷

Multiple classification tasks can also be mutually exclusive, meaning that exactly one of the predefined categories must be assigned to each sample. The i2b2 smoking status challenge task to be studied here is of this type. Samples may be somewhat compatible with the assignment of more than one label and therefore the automated system must determine the best, or most likely, or strongest supported label assignment for each sample. Alternatively, a system can produce a series of confidence measures for each possible label, and allow the user of the system or the next system in the pipeline (if the output of the classification system is used as input to another system) to determine how this information should be interpreted.

The accuracy with which these tasks can be done depends upon the size and quality of the training data, as well as specific domain features of the problem. Furthermore, whether or not current state-of-the-art approaches can provide an acceptable level of performance is dependent both upon the needs of the intended users of the automated document classification system, and

the accuracy to which human experts themselves can perform the task. For example, if the human experts can dependably assign consistent labels 80% of the time (determined, for example by inter-annotator agreement studies⁸), and automated methods can achieve, say 75%, then it remains to be decided by the users what level of performance is required, and whether the cost of paying the human annotators is worth the performance improvement over the automated system.

For some tasks, such as conducting research based on the information contained in patient medical records, a substantial benefit in human effort savings is possible by using automated systems if the automated systems can identify the records that the investigators are interested in with good accuracy and with a relatively small amount of random error (as opposed to some systematic bias that may affect the research). In this situation, researchers may, if they deem it necessary, manually review records identified as appropriate by the automated classification system, so records labeled of interest that are really not of interest will be identified and removed. For some tasks a low percentage of errors may be tolerable and no human review of the label assignments necessary. On the other hand, errors that mislabel records of interest as not of interest will not be detected by human inspection, and therefore these errors need to be randomly distributed across the area under study by the investigators. For a specific task, given enough training data, it is straightforward to confirm that this is the case by performing cross-validation experiments on the training data and examining the distribution of errors.

The i2b2 challenge included a five-way, mutually exclusive classification task meant to model the task of identifying patient records of interest in terms of smoking status (cigarettes, cigars, etc.). The challenge task was organized to evaluate the ability of automated systems to identify the smoking status of a given patient from a hospital discharge summary. For the purpose of this task, smoking status was defined as one of five mutually exclusive categories:

- UNKNOWN - there is insufficient information to determine any of the other classes.
- NON-SMOKER – the patient has never smoked.
- SMOKER – the patients smokes or has smoked, but there is insufficient information to determine which of these two is correct.
- PAST SMOKER – the patient was a smoker in the past, but does not currently smoke.
- CURRENT SMOKER – the patient is an active smoker.

The challenge task was intended to evaluate how well automated systems could perform this task given a modest amount of manually labeled de-identified training data. Training and test collections were created using human-experts to manually assign the labels. The task organizers studied the performance of the human annotators during the creation of the test and training sets. They found that two human annotators agreed about 81.5% of the time on which one of the five labels to assign to each summary. If automated systems can perform at this level, it is easy to see how such systems could be useful in a clinical research setting, for example, to select study cohorts from a large data warehouse of de-identified data, dramatically reducing the number of documents that need to be manually reviewed by the scientists conducting the studies.

2 Methods

2.1 Classifier system approach

Our approach to this multi-classification problem is shown in Figure 1. Training proceeds in a sequence of five steps. The classification of unknown samples uses the same sequence of five steps except that the final step makes a prediction based on the models created during training instead of using the training data to create those models. Each step in the process will be described in detail in the following sections.

2.1.1 Hot-spot passage isolation

On examination of the training data we found that the discharge summaries contained a wide variation of information on a wide range of aspects of patient medical history. We hypothesized that words would have different effects on the classification depending upon where they occurred in the discharge summary, and that words occurring near text describing the smoking status of the individual would be the most important. Conversely, we thought that words occurring far away from text describing smoking status would most likely contribute only noise to the machine learning algorithms. Since our basic classifier approach was to treat features extracted from the text without specific position information (often called a “bag of words”⁹), we needed a simple way to incorporate our hypothesis into the algorithm.

We call the approach that we came up with “hot-spot passage isolation”. By examining the training data we found that there were a small number of text patterns that indicated that the nearby text pertained to the patient’s smoking status. We used these text patterns to identify the “hot spots” in the discharge summaries from which we would extract the word based features. While more complex NLP and parsing methods are certainly possible, by experimentation using cross-validation on the training set we found that simply taking a window of text up to 100 characters before and after an identified hot-spot lead to good performance. Cross-validation micro-averaged F1 was a few percentage points lower for windows of both 50 and 200 characters.

The hot-spot identified passages were then isolated and passed on to the next step in the process, tokenization and vectorization. For the purposes of classifying smoking status, the rest of the discharge summary was ignored. The hotspot identifying text patterns that we used are shown in Table 1.

2.1.2 Tokenization and vectorization

Once the hot-spot identified passages were isolated, these were tokenized into individual words and symbols using the *StandardAnalyzer* module available in the Apache Lucene search engine library (available at <http://jakarta.apache.org/lucene>). We chose this tokenizer because it handles non-space delimiters well and we have had good success with it in the past. It also implements a small stop list, filtering out words such as “a”, “are”, “for”, and “into”. The *StandardAnalyzer* also filters out the stop words “no” and “not”, which we thought might be important for determining smoking status. However, cross-validation experiments on the training data found no differences in performance when these negation words were included or excluded.

After tokenization, the word list was converted into a binary vector. Each position in the vector provides information on the status of a specific token within the text sample, a value of one indicating the presence of the token in the text sample, and a value of zero indicating its absence. We did not use token counts within a sample, or other feature weighting approaches such as TF*IDF.¹⁰ We also did not perform any feature selection. All token based features that were identified by the hot spot and tokenization process were included in the feature vectors.

2.1.3 Zero vector filtering

After passing a discharge summary through the hot spot and tokenization processes, some documents result in a zero feature vector, that is, there are no identified tokens for that sample. With multiple-classification problems, it is likely that these samples will not add useful decision making information to the machine learning algorithms, and conversely may bias the algorithms by contributing to class prior-probabilities.

One problem with not passing zero vector samples to the machine learning algorithm is that the filtering has to be done both during training and classification of unknown samples. That means that the system has to assign a class to these zero vector samples based on some problem

domain specific criteria. The most common such criteria is probably simply assigning the zero vector samples to the most common class. However in the case of smoking status classification, there is a more logical way to make this choice. Since the classification task specifically includes an UNKNOWN category, it makes the most sense to classify samples without pertinent features into this class. In effect the system uses the lack of non-zero classification features as the rule for classifying a sample as UNKNOWN. Cross-validation experiments on the training set supported the use of this rule for the UNKNOWN class. Experiments showed a micro-averaged F1 improvement of about 0.10 compared to the same system simply passing the zero vectors on to the machine learning algorithm.

2.1.4 Multiple classification using error correcting output codes and support vector machines

2.1.4.1 Error correcting output codes

Error-correcting output codes (ECOC) is a technique used in communications based on information theory that adds redundancy to an encoded bit stream in order to allow detection and correction of incorrectly received bits. The application of this technique to multi-classification problems was first described by Dietterich and Bakiri¹¹, and is a powerful way of handling mutually exclusive text multi-classification problems.¹² While it is well known in the machine learning community, it has not been frequently applied in the biomedical informatics domain. For example, Yeang et al. describe the technique in their paper on classification of tumor types, but then use a hierarchical code instead of the error correcting technique.¹³ This may be due to some of the limitations inherent in the use of ECOC for problems with large numbers of classes. Nevertheless, it is a powerful approach to multi-classification problems with smaller number of mutually exclusive classes, perhaps between four and eight categories. The practical reasons for this will become clear as we describe the generation and application of the codes.

In the ECOC technique a multiple classification problem is reduced to solving multiple binary classification problems by partitioning the original classes into several positive and negative sets using the ECOC approach to guide the partitioning. An *ensemble*¹⁴ of classifiers is then trained on the training data, one per partition. When an unknown sample is to be classified, each classifier in the ensemble makes a binary prediction, and then the set of binary predictions is compared to the original partitioning in order to find the original class with the closest binary partitioning. The idea here is to create an error-correcting set of partitions where making a few binary classification mistakes does not prevent the classification ensemble from correctly categorizing a sample into the correct one of several mutually exclusive classes. The partitioning is determined automatically by the algorithm, is a function solely of the number of classes, and is independent of the training collection.

We will illustrate this approach using the smoking status challenge task as a specific example. This is a five-way mutually exclusive classification problem. The globally maximized separated set of codes for $k=5$ is shown in Table 2. Fifteen separate binary classifiers are created and trained, which we have labeled A through O for convenience. Each smoking status class is shown on the left hand column, and how that class is included in a particular binary classifier is shown as a plus sign (“+”) for classes that are included as positive samples for that binary classifier, and a minus sign (“-”) for negative samples. For example, the UNKNOWN class is considered a positive sample when training all 15 of the binary classifiers. The NON-SMOKER class is treated as a negative sample for binary classifiers A through H, and as a positive sample for classifiers I through O. It is important to note that all of the binary classifiers are trained on the same training set, only the class labels fed to the supervised learning algorithm are changed.

After the 15 binary classifiers are trained, by using the coding table we can apply these classifiers to categorize an unknown sample. Each of the binary classifiers is in turn applied to the sample to make a binary prediction. We construct a binary vector of these positive and negative predictions, maintaining the column correspondence between the binary classifier and the resulting prediction. We then find the row in the coding table that has the minimum Hamming distance from the binary vector. The predicted class is then the label for that row in the coding table. For example, if the prediction vector for an unknown sample was found to be “[- - - + - + - + - + - + + -]”, the predicted class would be CURRENT SMOKER because that row is only one bit change away from the prediction vector, and there is no closer row.

For the ECOC technique to be effective, it is important that the generated partitions be as far apart as possible in terms of Hamming distance. The Hamming distance between two binary bit vectors is the minimum number of bits that need to be changed to transform one vector into the other. The error correction capability of these codes is equal to one-half the minimum between-vector Hamming distance minus one. The number of correctable errors in the code is equal to the number of incorrect binary classifications that the system can handle and still predict the correct mutually exclusive category. Dietterch and Bakiri discuss code table generation in depth and provide exact algorithms for the generation of globally maximized separated codes when k , the number of mutually exclusive categories is between 3 and 11, and locally maximized separated codes for $k > 11$.

Because of the Hamming distance separation between the rows of the table, the 5 class ECOC method can detect and correct up to three incorrect binary predictions out of the 15. To see why this is the case, examine Table 3. This table shows the code length and inter-row Hamming distance for multi-class problems with a number of classes between three and 10. As

we have seen in Table 2, the code length (or equivalently, number of binary classifiers) for a 5 class problem is 15, and the minimum inter-row Hamming distance is 8. A code such as this can detect and correct for a number of errors of up to half the minimum inter-row Hamming distance minus one. This is because samples are classified into the closest (in terms of Hamming distance) row, and no mistakes will be made if the number of binary classification errors is less than that number. For $k=5$, half the minimum inter-row Hamming distance minus one is equal to 3.

The main advantage of the ECOC technique is that it can recover from errors in the individual binary classifiers more gracefully than other popular multi-classification methods such as one-against-all-others. The one-against-all-others method creates a single classifier for each of k classes and uses all samples from the other classes as negatives. Therefore a single classifier error in the classifier of the sample's true class will result in an incorrectly classified sample. For example, assuming that the average probability of error for any single binary classifier in a 5-way problem is equal to p , then the probability of an error using the one-against-all-others method on a sample of a given class is proportional to p . The probability of an error using ECOC is proportional to p^4 since four errors have to be made before the ECOC classifier ensemble makes an error. Even if the average probability of one of the ECOC ensemble classifiers is larger than p , the ECOC method is likely to show improvement until the average classifier error rate approaches the fourth root of p .

The most obvious limitation of the ECOC technique is that it requires the training and application of more binary classifiers than methods such as the one-against-all-others method. Whereas the one-against-all method creates k binary classifiers for a problem with k classes, the ECOC method creates $2^{k-1}-1$ binary classifiers. Therefore, in general the ECOC method will be $(2^{k-1}-1)/k$ times slower than the one-against-all-others method. For small k , this is usually

acceptable, but for large k it may be prohibitive. For example, with $k=16$, the ECOC method is predicted to be almost 2,000 times slower than one-against-all. For problems with large k , Dietterch and Bakiri show how to create good codes that are not as long, at the expense of losing some error correcting ability. It is necessary to take task-specific considerations into account to determine when it is productive to make this trade off. Dietterch and Bakiri have also shown that the ECOC technique is helpful with small sample sizes, such as with the i2b2 smoking challenge training data.

For the smoking status classification problem, the ECOC is a good fit. There are not too many mutually exclusive classes, the runtime penalty is only 3x as compared to one-against-all, and the sample sizes are relatively small. Therefore for this problem we expected ECOC to provide some benefit.

2.1.4.2 Weighted Support Vector Machines

In practice, any binary classification learning algorithm could be used to implement the 15 individual binary classifiers. However, we choose to use the libSVM implementation of the support vector machine (SVM) technique first proposed by Vapnik.^{15, 16} SVM has been shown to perform well on text classification problems with sparse vectors such as we have here, and we have had good results with it in the past.¹⁷⁻¹⁹

One issue with SVM-based classifiers that has sometimes been troublesome for biomedical text classification is that the optimization function attempts to minimize errors. Implicitly, if one class is more common than another, then this approach will favor the more frequent class. To address this we used the weight parameter available in libSVM. The unweighted approach just has all the weights set to 1.0, for all of the binary classifiers. In the weighted approach, the

positive and negative classes for each binary classifier are weighted by their relative rarity using the following formula:

$$w_{class} = (N - N_{class}) / N \quad (1)$$

Where N is the total number of samples and N_{class} is the number of samples of the given class. When using the ECOC method, N_{class} is either N_+ or N_- , for the positive and negative classes, respectively. N_+ is the number of samples in the positive partition for a given binary classifier, and N_- is similarly the number of samples in the negative partition.

During initial development we used cross-validation on the training data to explore different SVM kernels. We did not find any advantage to using polynomial or radial basis-function kernels for this problem. Therefore, all of our results shown here used libSVM with the linear kernel, either weighted or un-weighted as described above.

2.1.5 Post-processing rules

Since the training collection only included a few hundred samples, we hypothesized that there might not be enough data for the algorithm to identify all important combinations of features. Using cross-validation on the training set, we examined the errors that our system made and created post-processing transformation rules based on the predicted class of the ECOC ensemble and text patterns that we found during the error analysis. Our post-processing rules are shown in Table 4.

The rules were applied in the following manner. If the sample had a prediction given in the second column of the table, and matched the text pattern given by the regular expression in the first column, then the predicted class was modified to be the one given in the third column of the table. While the patterns appeared fairly general to us, since they were derived by manual

inspection of the training data we could not accurately estimate their effect on performance before using them on the test data.

2.1.6 Official submissions

We submitted three runs in our official submission to the i2b2 challenge task. The first run consisted of the weighted SVM ECOC system with the post-processing rules. The second run included the weighted SVM ECOC system without the post-processing rules. The third run submitted used the un-weighted SVM ECOC system with the post-processing rules. All three systems include the hot-spot passage isolation and mapping zero vectors to the UNKNOWN class since we found these techniques to consistently contribute a strong improvement in performance during cross-validation testing on the training set.

3 Evaluation

The data for the i2b2 challenge was divided into training and test sets. The training data consisted of 398 discharge summaries with accompanying smoking status information that specified which of the five classes were assigned by expert annotators to that summary. The test collection consisted of 104 discharge summaries similar to those in the training data. Both the test and training collections were assembled from samples on which the human annotators had 100% agreement. The number of samples assigned to each class in the training and testing collections is shown in Table 5.

The smoking status assigned by the expert annotators to the summaries in the test collection were not available to i2b2 participants until after all teams had submitted all of their official runs. Each team was allowed to submit up to three runs. The i2b2 challenge task was evaluated using the standard classification measures of precision, recall, and F1-measure, extended for multiple classification problems.⁹ The primary measure of comparison was the F1-measure, with

precision and recall weighted equally ($\beta = 1.0$). Micro- and macro-averaged F1 values were computed. The calculation of both of these types of average F1 values begins with counting the true positive, false positive and false negative predictions for each label. From these counts the precision, recall, and F1 can be computed for each label, treating each label as a binary classification task. The micro-averaged F1 value is computed by weighting each class-specific F1 by the fraction of overall samples in the gold standard that belong to that class. Macro-averaged F1 is computed by simply averaging the class-specific F1 values without any weighting factor. The result of this is that micro-averaged F1 is more sensitive to the performance on classes with a relatively large number of members, and macro-averaged F1 is more sensitive to the performance on rare classes. Which measure is “better” or more appropriate for a given task is dependent upon the requirements of the users of the system. For the i2b2 task, micro-averaged F1 was taken as the primary measure.

For the i2b2 smoking status classification task, the organizers scored the submissions in two ways in addition to applying micro- and macro- averaged F1. Submissions were scored as 5-way tasks as given in the training data, with each sample being classified into one of the five smoking status categories. They also created a 3-way task post-hoc, grouping SMOKER, PAST SMOKER, and CURRENT SMOKER into one SMOKER class, and keeping NON-SMOKER and UNKNOWN separate. Participants were not told in advance that these 3-way measures would be applied, so it is likely that most systems were optimized only for the 5-way task.

4 Results

In addition to the officially submitted and scored runs, we have conducted a series of experiments to make it possible to determine which features of our system lead to improved performance. Table 6 shows the results of these experiments, using the same 5-way scoring

method used by the i2b2 task organizers. The systems labeled Run1, Run2, and Run3 were our officially submitted runs, the systems labeled A through I are additional system configurations that we trained on the training set and scored on the test collection using methods equivalent to those used by the task organizers. The system labeled “i2B2 best” was the best performing system run submitted to the task organizers.

Configurations A through I systematically vary properties of our classification approach and make it possible to determine which properties contributed positively to the results and by how much. Therefore we varied the use of hot-spotting, tokenization, zero vector filtering, SVM weighting, the ECOC method, and inclusion of the post processing rules. The columns labeled “Zero vector filtering”, and “Post-processing rules” should be self explanatory as these simply represent whether the system configuration included these features as described in the Methods section above, or not. The column “Hot-spot” represents whether the system configurations used text host-spot identification or simply the entire text of the discharge summary. The columns “Tokenization”, “SVM weighting”, and “ECOC” require some further explanation.

The “Tokenization” column varies the tokenization method between two approaches. The “Lucene” approach uses the *StandardAnalyzer* from Lucene as described above. Because we were interested in whether filtering out negation stop words (as done by the Lucene tokenizer) might have a negative effect, we also tested runs that used a very simple tokenizer that split text only on white space and punctuation, and did not filter out any stop words.

In the column labeled “ECOC”, “Yes” denotes whether the system included use of the ECOC multi-classification method described above. An entry of “No” indicates that the system used the multi-classification approach built into libSVM. This is a one-against-all-others approach as previously described above.

Lastly, the column labeled “SVM weighting” indicates whether the inverse-frequency class weighting was used as the prior probabilities given to the individual binary classifiers. If weighting was not used, that means that the prior probabilities were all set equal to one.

As can be seen in Table 6, Run2 was our overall best performing submitted run. This system used Lucene tokenization, zero-vector filtering, SVM weighting, and ECOC, but not the post processing rules. In fact, this run scored second out of all runs submitted. Interestingly our concerns about stop-word filtering out negations were well founded. Even though we did not see this effect on our cross-validation experiments before submitting our official runs, on the actual test data there is a noticeable degradation in performance. Comparing system A to Run2, the only difference is the tokenization and lack of stop word filtering in system A. System A has a micro-F1 of 0.9000, which is 0.0140 better than Run1, and actually minutely higher than the best performing system submitted to the i2b2 challenge task.

Of the techniques proposed here, by far the largest effect was due to hot-spotting. The value of hot-spotting for this task can be seen by comparing the results of System A with System I, the only difference between these two approaches being the use of hot-spotting. The difference in micro-F1 is very large, 0.2960, with the hot-spotting system achieving a micro-F1 of 0.9000 as compared to the non-hot-spotting system score of 0.6040. The single change of removing hot-spotting transforms the best performing system configuration presented here into the worst.

Several other informative comparisons can be extracted from Table 6. The only difference between Run2 and E is the use of zero-vector filtering. Run2 outperforms E by 0.02. The use of SVM weighting is the difference between Run2 and B, and Run 2 outperforms B by 0.014. The ECOC technique is compared to the one-against-all-others by looking at Run1 versus system G. This time Run1 outperforms G by 0.020. Finally for the 5-way evaluations, the use of the post-

processing rules was counterproductive. The use of this technique is the only difference between Run1 and Run2, and Run2 does a little better without it. Finally system H uses the inferior choice for each of the three alternative techniques (zero-vector filtering, weighting, ECOC), using hot-spotting but no post-processing rules (as with our best system), and achieves a micro-F1 of 0.8360, which is 0.0640 less than the best performing system. Interestingly this is very close to the sum of the individual contributions of each of the left-out features: $0.0140 + 0.20 + 0.0140 + 0.20 = 0.0680$.

Table 7 shows submitted system scores for the 3-way evaluation of smoking status. Run3 was one of the two top performing systems, achieving a micro-F1 of 0.9713, and a second-best macro-F1 of 0.9493. However, all three of our submitted runs score similarly. The scores on the three-way task are higher than the scores on the five-way task. The best 3-way micro-F1 is greater than the best 5-way score by about 0.08. Furthermore, the best macro- and micro-F1 scores are much closer in the three-way task than the five-way.

5 Discussion

By far, the biggest difference in performance was due to hot-spot filtering of the discharge summary text. It is clear that the value of word based features for classifying smoking status is dependent upon the text region in the discharge summary in which these words are found. The hot-spotting technique was a simple way to determine what areas of text to focus on, and to filter out a large amount of noise. Furthermore, the hot-spotting technique was completely reliable for this task using only the six identifying phrases shown in Table 1. Since the zero vector filtering method maps samples without non-zero features to the UNKNOWN class, performance on this class is a good indication of whether hot-spot based filtering is removing important features. For systems implementing hot-spotting and zero-vector filtering, in our experiments with the test

collection, precision and recall of the UNKNOWN class were both 100%. These means that hot-spotting did not remove any information that was necessary for discriminating between UNKNOWN samples and samples of other classes. While the well-known trade-off between precision and recall usually applies to procedures such as hot-spot filtering, in this case the technique greatly improved precision with no loss of recall.

Besides hot-spotting, the other techniques proposed and used here, including zero vector filtering, SVM inverse proportional weighting, and the use of ECOC for multi-classification problems all add a noticeable improvement to the system performance. While it was unexpected that these techniques would additively combine to produce an overall linear increase in performance, this does appear to be approximately the case. These three techniques must therefore improve performance in independent ways, which is both good for the smoking status challenge task and also provides some support that these methods may be useful when applied to other text classification problems.

The post-processing rules did not help performance. Nevertheless, one of our systems that included these rules, Run3, was the top performing submitted system when evaluated on the Micro-F1 on the 3-way task. Since systems were not optimized for the 3-way problem, it is not clear that these results are meaningful for comparison purposes across submissions. However, the results are meaningful for understanding how well systems can do when training on 5-way data for tasks where the 3-way labeling is important. At micro-F1 measures of over 0.97, it seems clear that our system, as well as other top performing submissions, can assign 3-way smoking status at a very high level of performance, a level that is likely to be more than adequate for many purposes.

The difference scoring between the 5-way and 3-way tasks is both interesting and informative. At first, the exceptional 3-way performance seemed a little surprising since systems were not specifically tuned for this task. However, the 3-way task is essentially a subset of the 5-way task, and it appears clear from these results that the greatest difficulty in the 5-way task is discriminating between the three classes of smokers. When these categories are collapsed into one, the task of discriminating between smokers, non-smokers, and people of unknown smoking status is much more straightforward, and current state-of-the-art systems can do very well on assigning smoking status in situations when the difference between current and past smokers is not important. Furthermore, collapsing 5 classes into 3 classes eliminated rare classes from the problem space, which naturally lead to higher macro-F1 scores that were similar to micro-F1 values, since macro-F1 emphasizes rare categories and the 3-way problem doesn't contain any of these.

In contrast to the 5-way task, the inclusion of the inverse-class frequency weighting did not improve performance on the 3-way task. In fact, the highest scoring submission was our Run3, which did not use the inverse-class frequency weighting. Since the 3-way task did not include relatively rare classes, it is to be expected that the inverse-class frequency weighting, which specifically makes the machine learning algorithm more sensitive to rare classes, would not have a positive effect. This can be seen by comparing Run1 to Run3, where the only difference is the use of weighting in Run1, which scores lower than Run2. The post processing rules were not helpful for the 3-way task, as can be seen by comparing Run1 to Run2, where Run1 includes the post-processing rules and scores lower than Run2.

All of the techniques used by our systems are of low computational complexity, in fact, since the hot-spotting method filters out most of the text of the discharge summary, the training and

classification steps run faster than without hot-spotting. The best 5-way performing system submitted to i2b2, “Clark3”, achieved a micro-F1 improvement of only 0.009 over our Run2, and actually scored a bit lower than our system A presented here. Macro-F1 was improved by a larger amount, 0.7568 up from 0.6504 for our Run2. As noted above, Macro-F1 treats each class as equal, and so it is likely that the improvement was due to better performance on the “SMOKER” category, which was an uncommon label in the training data and never predicted by our Run2 system.

In part, the improvements of “Clark3” were achieved by using a medical entities extraction natural language processing (NLP) system along with a larger training set that they tagged themselves. We did not have run-time data on this method but based on our experience with named entity recognition^{20, 21} and other NLP-based techniques it is reasonable to assume that the Clark3 system has much higher computationally complexity than our approach. The classification time for our system was very short, less than 0.01 second for each sample. Since our system is implemented in pure Python with no attempt at optimization, it is very likely that it could be made to run much faster. Whether this level of efficiency is important or not is of course determined by the specific application to which the system will be applied.

The use of additional manually annotated training data by at least one of the participants makes comparison between systems of different participants somewhat difficult. More reliable comparisons between different machine learning algorithms can be made when the algorithms are trained and tested on the same data sets.⁹ As noted above, our system never predicted the “SMOKER” label since it was rare in the training data and presumably there were no strong features in the i2b2 supplied training data that supported discrimination between that class and the other smoking categories. Additional training data would be very useful in improving the

ability to correctly detect the “SMOKER” class by the machine learning methods we have proposed here. A full comparative evaluation of the different learning approaches would require having the manually tagged training data used by the Clark3 system made available to the other algorithms.

6 Conclusions

We have proposed and shown the effectiveness of several general text classification techniques that are particularly well suited to the 5- and 3-way smoking status multi-classification task applied to discharge summaries. The techniques are of modest implementation complexity, highly runtime efficient, and based on solid theory and straightforward assumptions. Empirical studies on independent test data show performance competitive with the best available techniques, including methods that require more complex NLP techniques and additional customized training and lexicon data. Further work will be required to study the level of performance required for specific applications of automated smoking status identification, and to study the effectiveness of these systems to support specific areas of biomedical research.

References

1. Cohen AM, Hersh WR, Peterson K, Yen PY. Reducing workload in systematic review preparation using automated citation classification. *JAMIA* 2006;13(2):206-219.
2. Camon EB, Barrell DG, Dimmer EC, Lee V, Magrane M, Maslen J, et al. An evaluation of GO annotation retrieval for BioCreAtIvE and GOA. *BMC Bioinformatics* 2005;6 Suppl 1:S17.
3. Cohen AM, Hersh W. The TREC 2004 Genomics Track Categorization Task: classifying full text biomedical documents. *Journal of Biomedical Discovery and Collaboration* 2006;1(4).
4. Donaldson I, Martin J, de Bruijn B, Wolting C, Lay V, Tuekam B, et al. PreBIND and Textomy--mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC Bioinformatics* 2003;4(1):11.
5. Pakhomov SV, Buntrock JD, Chute CG. Automating the assignment of diagnosis codes to patient encounters using example-based and machine learning techniques. *J Am Med Inform Assoc* 2006;13(5):516-25.
6. Pakhomov SV, Buntrock JD, Chute CG. Using compound codes for automatic classification of clinical diagnoses. *Medinfo* 2004;11(Pt 1):411-5.
7. Larkey LS, Croft WB. Combining classifiers in text categorization. In: *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval* 1996; Zurich, Switzerland ACM Press; 1996. p. 289-297.
8. Wilbur WJ, Rzhetsky A, Shatkay H. New directions in biomedical text annotation: definitions, guidelines and corpus construction. *BMC Bioinformatics* 2006;7:356.
9. Sebastiani F. Machine Learning in Automated Text Categorization. *ACM Computing Surveys (CSUR)* 2002;34(1):1-47.
10. Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. *Inform. Process. Man.* 1988;24(5):513-523.
11. Dietterich TG, Bakiri G. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research* 1995:263-286.
12. Ghani R. Using error-correcting codes for text classification. In: Langley P, editor. *Proceedings of {ICML}-00, 17th International Conference on Machine Learning*; 2000: Morgan Kaufmann Publishers, San Francisco, US; 2000. p. 303--310.
13. Yeang CH, Ramaswamy S, Tamayo P, Mukherjee S, Rifkin RM, Angelo M, et al. Molecular classification of multiple tumor types. *Bioinformatics* 2001;17 Suppl 1:S316-22.

14. Dietterich TG. Ensemble methods in machine learning. Lecture Notes in Computer Science 2000;1857:1-15.
15. Chang C-C, Lin C-J, LIBSVM : a library for support vector machines, 2001; Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, accessed March 20, 2006.
16. Vapnik VN. The nature of statistical learning theory. 2nd ed. New York: Springer; 2000.
17. Cohen AM. An effective general purpose approach for automated biomedical document classification. AMIA Annu Symp Proc 2006:161-5.
18. Cohen AM, Yang J, Hersh WR. A Comparison of Techniques for Classification and Ad Hoc Retrieval of Biomedical Documents. In: Proceedings of the Fourteenth Annual Text REtrieval Conference - TREC 2005; 2005; Gaithersburg, MD; 2005.
19. Joachims T. Text categorization with support vector machines: learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning; 1998; 1998. p. 137-142.
20. Cohen AM. Unsupervised gene/protein entity normalization using automatically extracted dictionaries. In: Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, Proceedings of the BioLINK2005 Workshop; 2005; Detroit, MI: Association for Computational Linguistics; 2005. p. 17-24.
21. Cohen AM, Hersh W. A survey of current work in biomedical text mining. Briefings in Bioinformatics 2005;6(1):57-71.

Tables

Table 1. Hot-spot identifying phrases for smoking status within discharge summaries.

“smok”	“cig”	“tobac”
“packs”	“tob ”	“nicotine”

Table 2. ECOC codes for the i2b2 5-way smoking status classification problem.

i2b2 Class	ECOC for 15 Binary Classifiers (A through O)														
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
UNKNOWN	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
NON-SMOKER	-	-	-	-	-	-	-	-	+	+	+	+	+	+	+
SMOKER	-	-	-	-	+	+	+	+	-	-	-	-	+	+	+
PAST SMOKER	-	-	+	+	-	-	+	+	-	-	+	+	-	-	+
CURRENT SMOKER	-	+	-	+	-	+	-	+	-	+	-	+	-	+	-

Table 3. ECOC code lengths and Hamming distances for various numbers of classes, k .

k	Code Length	Hamming distance
3	3	2
4	7	4
5	15	8
6	31	16
7	63	32
8	127	64
9	255	128
10	511	256

Table 4. Post-processing rules.

Text Pattern	Predicted Class	Revised Prediction
no history of(cigarette)? smoking	CURRENT SMOKER	NON-SMOKER
no smoking history	CURRENT SMOKER	NON-SMOKER
not a smoker	CURRENT SMOKER	NON-SMOKER
former smoker	CURRENT SMOKER	PAST SMOKER
remote smoking history	CURRENT SMOKER	PAST SMOKER
stopped smoking	CURRENT SMOKER	PAST SMOKER
chronic tobacco	NON-SMOKER	CURRENT SMOKER
current smoker	NON-SMOKER	CURRENT SMOKER
patient smokes	NON-SMOKER	CURRENT SMOKER
smoked in the past	NON-SMOKER	PAST SMOKER
former tobacco user	NON-SMOKER	PAST SMOKER
former smoker	NON-SMOKER	PAST SMOKER
past tobacco use	NON-SMOKER	PAST SMOKER
nicotine abuse	NON-SMOKER	PAST SMOKER
never smoked	PAST SMOKER	NON-SMOKER
not a smoker	PAST SMOKER	NON-SMOKER
no history of smoking	SMOKER	NON-SMOKER
current smoker	PAST SMOKER	CURRENT SMOKER
smokes (\w+\W+){1,3}(packs? per day ppd)	PAST SMOKER	CURRENT SMOKER
non-smoker	PAST SMOKER	NON-SMOKER
never smoked	PAST SMOKER	NON-SMOKER
not a current smoker	CURRENT SMOKER	PAST SMOKER
ex-smoker	CURRENT SMOKER	PAST SMOKER
quit(smoking)? tobacco	NON-SMOKER	PAST SMOKER
history of (tobacco cigarette smoking)	NON-SMOKER	SMOKER
ex-smoker	NON-SMOKER	PAST SMOKER
no history of(cigarette)? smoking	CURRENT SMOKER	NON-SMOKER

Table 5. Sample counts per class in the test and training collections.

Class	Training	Testing
UNKNOWN	252	63
NON-SMOKER	66	16
SMOKER	9	3
PAST SMOKER	36	11
CURRENT SMOKER	35	11
TOTAL	398	104

Table 6. Micro- and macro-averaged F1 results for 5-way classification by tested systems on test collection.

System	Hot-spot	Token-ization	Zero vector filtering	SVM weighting	ECOC	Post-processing rules	Micro-F1	Macro-F1
Run1	Yes	Lucene	Yes	Yes	Yes	Yes	0.8804	0.6430
Run2	Yes	Lucene	Yes	Yes	Yes	No	0.8860	0.6504
Run3	Yes	Lucene	Yes	No	Yes	Yes	0.8713	0.6190
A (not submitted)	Yes	Simple	Yes	Yes	Yes	No	0.9000	0.6780
B (not submitted)	Yes	Lucene	Yes	No	Yes	No	0.8720	0.6220
C (not submitted)	Yes	Lucene	No	No	Yes	No	0.8470	0.5960
D (not submitted)	Yes	Lucene	No	Yes	Yes	No	0.8340	0.5780
E (not submitted)	Yes	Simple	No	Yes	Yes	No	0.8660	0.6250
F (not submitted)	Yes	Lucene	Yes	No	No	No	0.8380	0.5640
G (not submitted)	Yes	Lucene	Yes	Yes	No	No	0.8660	0.6090
H (not submitted)	Yes	Lucene	No	No	No	No	0.8360	0.5670
I (not submitted)	No	Simple	Yes	Yes	Yes	No	0.6040	0.3050
i2b2 Best							0.8955	0.7568

Table 7. Micro- and macro-averaged F1 results for 3-way classification by tested systems on test collection.

System	Micro-F1	Macro-F1
Run1	0.9516	0.9136
Run2	0.9615	0.9317
Run3	0.9713	0.9493
Best	0.9713	0.9518

Figures

Figure 1. System depiction of the smoking status classification process.

