A Rule-Based Scheme for Filtering Examples from Majority Class in an Imbalanced Training Set

Jamshid Dehmeshki, Mustafa Karaköy, and Manlio Valdivieso Casique

Medicsight Plc., 46 Berkeley Square, Mayfair
London, England W1J 5AT
jamshid.dehmeshki@medicsight.co.uk

Abstract. Developing a Computer-Assisted Detection (CAD) system for automatic diagnosis of pulmonary nodules in thoracic CT is a highly challenging research area in the medical domain. It requires a successful application of quite sophisticated, state-of-the-art image processing and pattern recognition technologies. The object recognition and feature extraction phase of such a system generates a huge imbalanced training set, as is the case in many learning problems in medical domain. The performance of concept learning systems is traditionally assessed with the percentage of testing examples classified correctly, termed as accuracy. This accuracy measurement becomes inappropriate for imbalanced training sets like in this case, where the non-nodules (negative) examples outnumber nodule (positive) examples. This paper introduces the mechanism developed for filtering negative examples in the training so as to remove 'obvious' ones, and discusses alternative evaluation criteria.

1 Introduction

Early detection of lung cancer is crucial in its treatment. Conventionally, radiologists try to diagnose the disease, by examining computed tomography (CT) images of the subject's lung and then deciding if each suspicious object, i.e., region of interest (ROI) is a nodule or a normal tissue. The manual radiological analysis of CT images is a time consuming process. Therefore, developing a Computer-Assisted Detection (CAD) system for automatic diagnosis of pulmonary nodules in thoracic CT is a highly challenging research area in the medical domain [1].

Achievement of this task requires the successful application of state-of-the-art image processing and pattern recognition techniques. The image processing tasks are followed by nodule detection and feature extraction processes. This often results in large, imbalanced data sets with too many non-nodule examples, since it is important to avoid missing any nodules in the images. Constructing an accurate classification system requires a training data set that represents different aspects of nodule features. This paper assumes an object has been detected and deals with the subsequent object learning. It focuses on the problem of extremely imbalanced training sets, (i.e. the

P. Perner and A. Rosenfeld (Eds.): MLDM 2003, LNAI 2734, pp. 215–223, 2003.

relatively high number of negative non-nodule results, compared to the low number of positive nodule results.)

Informally, a good performance on positive examples and negative examples is expected rather than one at the cost of the other. However, the classification performance in this kind of task cannot be expressed in terms of the average accuracy since the training set is extremely imbalanced in that the non-nodules (negative) examples heavily outnumber the nodule (positive) examples, and classifiers tend to over-fit non-nodule examples. Another problem with the training set is the training time due to the huge size of the data set. Filtering/eliminating some negative examples would help solve both problems so long as it does not deteriorate the learning performance.

2 Discovering Safe Regions in the Feature Space

Training sets for concept learning problems are denoted by pairs [x, c(x)], where x is a vector of attribute values of an example and c(x) is the corresponding concept label. In our case, c(x) is either positive or negative. Nevertheless, there is always a huge difference between the prior probabilities of the positive and negative examples. In other words, negative examples are represented by a much greater number of examples than the positive ones in the training data set, as is often the case in many learning problems of medical domain.

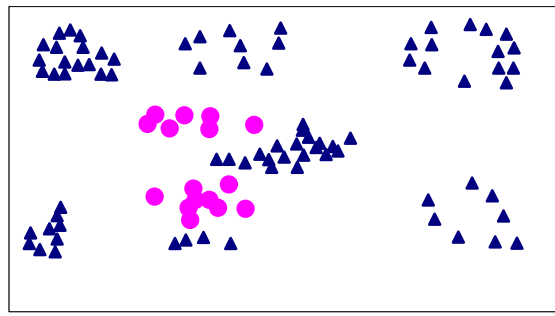
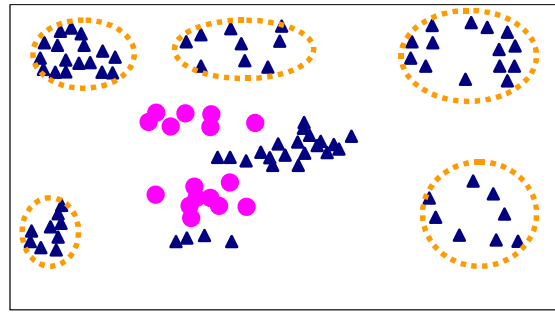


Fig. 1. An example of imbalanced training sets with two attributes

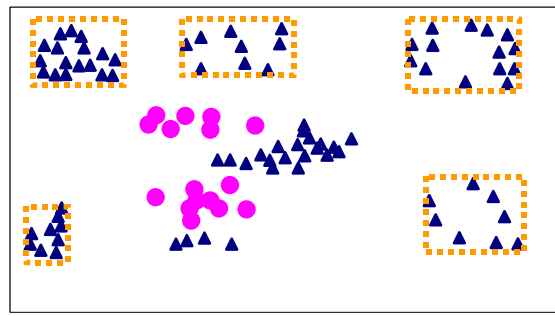
In an extremely imbalanced training data set (see Fig. 1), many sections of the feature space for x vectors, i.e., feature space, are likely to be comprised of those pairing with majority-class only and also quite far from the ones pairing with minority-class examples. Furthermore, radiologists believe that nodule cases have certain characteristics that would locate them in certain areas of feature space only. In other words, there should be many nodule free regions in the feature space. In summary, a learning model is constructed through following steps:

1. Discovering 'safe' regions in the feature space, where only negative examples exist.
2. Constructing filtering rules each of which defines the corresponding 'safe' region.
3. Eliminating training examples covered by these regions from the training data set.
4. Training a classifier or classifiers with the modified training set.

This paper focuses on reducing/filtering an imbalanced training set, which is part of our CAD system. Therefore, the last step is out of the scope of this paper.



(a)
Ellipse Regions



(b)
Rectangular Regions

Fig. 2. Determining safe regions

Figure 2 illustrates two examples of determining such regions for the imbalanced set with two attributes mentioned above. In a multidimensional space, those regions can be thought of as distinct hyper-ellipsoids or hyper-cuboids. Assuming the training data is a representative set of the problem it is plausible to construct rules for specification of those regions, and to label any test point satisfying any of these rules (i.e., being inside one of those regions) as non-nodule (i.e., negative). Such a rule not

only diminishes the training data set for the subsequent classifier but also could be a filtering and first-level classification mechanism for 'easy' non-nodule test examples[1].

First, a clustering (unsupervised learning) algorithm is applied to the whole (imbalanced) training set, which divides the data set into a specified number of distinct groups. Then, the 'pure' negative clusters, which consist of negative examples only, are marked. For each of these pure clusters, a hyper-ellipsoid or a hyper-cuboid is specified, and all examples of these clusters are removed from the training set of the subsequent classifier.

The algorithm for determining these safe regions in terms of hyper-ellipsoids is as follows:

1. Group the whole training set into a certain number of clusters using an appropriate clustering and mark the pure negative clusters. K-means clustering [2] and Gaussian mixture model (GMM) clustering [3] with expectation maximization (EM) [4] are used in this study.
2. For each cluster, set the center (c) of the corresponding hyper-ellipsoid to its mean vector (m) as defined below:

$$c_j = m_j . \tag{1}$$

3. For each cluster, set the initial radius values in all dimensions/attributes in terms of its standard deviation vector as follows[2].

$$r_j = 3 * s_j . \tag{2}$$

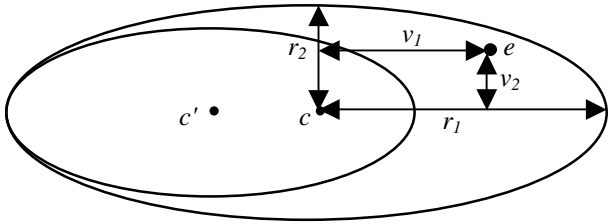


Fig. 3. Shrinking an ellipses to avoid a positive example

4. As shown in Fig. 3 on two-dimensional space for the sake of simplicity, for any positive training example (e) falling in a region, first determine the dimension (k) where the difference between example's value and the center is the biggest (v_k) as follows:

---

[1] Classifiers mostly fail on the examples close to the decision boundaries, hence these examples are difficult to classify. On the contrary, the examples far from the decision boundaries could be considered 'easy' examples.

[2] This formula makes sure that at least 99% (probably all) of the samples in the cluster are covered assuming they have a normal (i.e., Gaussian) distribution. Other heuristics could be applied instead. E.g., center and radius might be determined by minimum and maximum attribute values for the cluster.

$$v_k = \max(v_j) \qquad \text{where} \quad v_j = \left| e_j - c_j \right| \qquad \text{for all } j \,. \tag{3}$$

5. Then, update (i.e., shift) the center value in that dimension and modify the radius values in all dimensions (j) as follows:

$$\begin{aligned} c_k' &= (e_k + 3 * c_k + 2 * r_k)/4 && \text{if } e_k < c_k \\ c_k' &= (e_k + 3 * c_k - 2 * r_k)/4 && \text{otherwise} \end{aligned} \,. \tag{4}$$

$$r_k' = (v_k + 2 * r_k)/4 \,. \tag{5}$$

$$r_j' = r_j * r_k' / r_k \qquad \text{for all } j \text{ where } j \neq k \,. \tag{6}$$

Note the radiuses in all dimensions as well as the radius in the dimension with the maximum difference are recalculated, but also radiuses in all other dimensions are recalculated. Otherwise, the new region includes (even though relatively small but potentially not safe) areas that are outside the original region due to the shift of the center.

Similarly, the algorithm for determining these regions in terms of hyper-cuboids is as follows:

1. The first step is the same as in the previous algorithm.
2. Initially, define a hyper-cuboid for each cluster in terms of minimum and maximum values in each dimension as follows:

$$min_j = m_j - 3 * s_j \,. \tag{7}$$

$$max_j = m_j + 3 * s_j \,. \tag{8}$$

where m denotes mean vector of the cluster while s is the standard deviation vector.

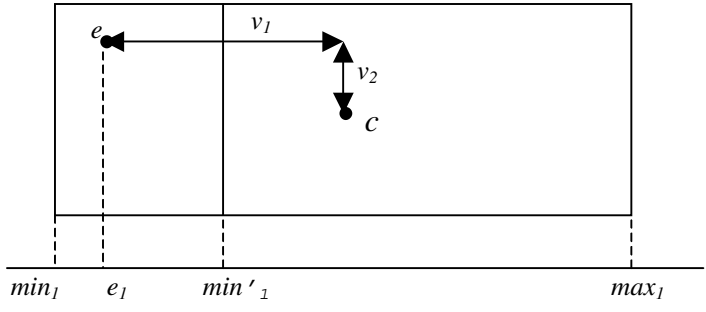


Fig. 4. Shrinking a rectangular to avoid a positive example

3. As shown in Fig. 4 on two-dimensional space for the sake of simplicity, for any positive training example (e) falling in a region, first determine the dimension (k) where the difference between example's value and the center is the biggest (v_k), and then either update the *min* value in that dimension if it is smaller than the example's value (as in Fig. 4), or update the *max* value otherwise as follows:

$$min'_k = (min_k + max_k + 2 * e_k)/4 . \tag{9}$$

$$max'_k = (min_k + max_k + 2 * e_k)/4 . \tag{10}$$

Once rules are constructed, the first part, i.e., the proposed filtering scheme for an overall classification system is complete. In learning phase of a two-stage classification system, this rule-based mechanism act as a filtering scheme for the training data to the classifier in the second stage, while it operates as a first-stage detection of 'easy' negative test cases in the test/classification phase.

3 Evaluation Criteria

Statisticians generally formulate the performance with a confusion matrix shown in Table 2 that characterizes the classification behavior of a concept learning system [5]. Based on this matrix, the traditional accuracy, i.e., the percentage of testing samples classified correctly, is calculated as follows:

Table 1. Confusion matrix

		Predicted	
		Negative	**Positive**
Real	**Negative**	*TN*	*FP*
	Positive	*FN*	*TP*

TN: the number of true negatives
FN: the number of false negatives
FP: the number of false positives
TP: the number of true positives

$$accuracy = \frac{TN + TP}{TN + FN + TP + FP} . \tag{11}$$

However, this bare accuracy measurement becomes inappropriate in the case of imbalanced training sets [6]. In this case, researchers choose different criteria for the performance. For instance, information retrieval community prefers to work with so called *precision* and *recall*. Below is the formulization of these measurements based on the confusion matrix:

$$precision = \frac{TP}{TP + FP}. \tag{12}$$

$$recall = \frac{TP}{FN + TP}. \tag{13}$$

These quantities are sometimes amalgamated into a single value called F-measure by giving them equivalent or different weights. When both precision and recall are considered equally important, the F-Measure (F) is computed as follows:

$$F = \frac{2 * precision * recall}{precision + recall}. \tag{14}$$

A common alternative for the combination is the geometric mean (g) of precision and recall values as given below [7]:

$$g = \sqrt{precision * recall}. \tag{15}$$

As in the F-measure formula in Equation **14**, this metric reaches high values only if both precision and recall are high and in equilibrium.

There are also other criteria such as Receiver Operating Characteristic (ROC) curve analysis[3], the one frequently used for the problems in medical domain [8]. All these measurements are more suitable than the simple accuracy value as a performance metric for the systems learning from highly imbalanced training set.

However, the scheme here is not a complete system for such a task. Rather, it will constitute part of such a system as a first-level detection of negative examples. Therefore, the criteria used for the filtering scheme consist of *error ratio* (*ER*) on the test set and *filtering ratios* (*FR*) on the test set and as well as the train set. Following are the formulae:

$$ER = \frac{FN}{FN + TP}. \tag{16}$$

$$FR = \frac{TN}{TN + FP}. \tag{17}$$

Error ratio indicates how reliable it is in terms of not missing any positive example, whereas the filtering ratio shows how useful it is with respect to detecting/eliminating as many negative examples as possible. It is aimed to get a low error ratio with as much a high filtering ratio as possible.

---

[3] Originally, ROC curve analysis was developed during World War II for the analysis of radar images as a signal detection theory. It was used to measure the ability of radar receiver operators in deciding if a blip on the screen is an enemy target, a friendly ship, or just noise. However, it was recognized as useful for interpreting medical test results after the 1970's.

4 Experiments

The whole data used in the experiments consisted of 152382 examples such that only 739 were positive examples while 151643 were negative examples with 8 attributes. The data were normalized and randomized in a pre-processing task since the attributes were in diverse ranges. In addition, the positive and negative examples were separately split into 5 groups each so as to apply 5-fold cross-validation so that the prior probabilities of the classes are the same for each fold. More precisely, for a particular fold one fifth of positive examples and one fifth of negative examples formed a validation set while the rest of the whole data was the train set.

The program written to run experiments had 3 options: the clustering method (k-means or GMM), the number of clusters and the regions shape (hyper-ellipsoid or hyper-cuboid). Table 2 reports the filtering results when training sets in folds were clustered into groups of 250 using the k-means method. There was one problem with the fourth and fifth training sets. The method failed to cluster data into 250 groups and for this reason these two were clustered into 200 groups instead. On the other hand, clustering with GMM in place of k-means did not change these results much.

Table 2. Filtering results

Fold No	Filtering Ratio		Error on Test Set
	Train Set	Test Set	
1	49.83%	49.54%	4.08%
2	52.68%	52.76%	2.72%
3	52.62%	52.48%	8.84%
4	38.44%	38.11%	2.72%
5	43.11%	43.53%	2.04%
Average	47.34%	47.28%	4.08%

5 Discussion and Conclusion

At first glance, the filtering ratios might be considered low. However, remember that this scheme alone does not offer a complete classification system. Rather, its provides a first stage appraisal of test examples by the system as well as reducing the training data set to the classifier in the second-stage. Especially considering the fact that the error ratios in all cases are below 9% (i.e., much smaller than those of the classifiers trained with the same data set where they were above 15% in all cases), this mechanism proves to be useful in reducing the learning time for model-based algorithms, and the testing time for case-based algorithms.

The specification/definition of ‘safe’ regions in the feature space is important. In this study, these regions are specified as hyper-cuboids or hyper-ellipsoids for the

sake of simplicity of their mathematical definitions and less complexity requirements. However, distributions of examples inside clusters are not further investigated to better cover examples in the clusters based on the distribution. In this manner, some alternatives will be examined in future study.

In conclusion, this paper presented a scheme for filtering examples from the majority class in an imbalanced training set in general, and for filtering of non-nodule examples in particular, which is vital to improve the performance of our CAD system for nodule detection. As an initial evaluation of test examples in a classification process, this rule-based scheme also makes a contribution by eliminating easy negative examples, which bring about the reduction in learning time when a model-based classifier such as an Artificial Neural Network (ANN), or the reduction in decision making when an instance-based classifier such as k-nearest neighbor (kNN) is used, in addition to some improvement in performance. Hence, this mechanism also enables combination of rule-based and instance-based induction when a case-based algorithm is applied in the second stage, which differs from Domingos' RISE system that unifies these two induction strategies [9].

References

1. Lee, Y., Hara, A., Hara, T., Fujita, H., Itoh, S., Ishigaki, T.: Automated Detection of Pulmonary Nodules in Helical CT Images Based on an Improved Template-Matching Technique. In: IEEE Transactions on Medical Imaging, Vol. 20, No. 7. (2001) 595–604
2. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1. (1967) 281–297
3. Bishop, C..: Neural Networks for Pattern Recognition. Oxford University Press, UK (1995)
4. Dempster, A., Laird, N., Rubin, D.: Maximum Likelihood from Incomplete Data via the EM Algorithm. In: Journal of the Royal Statistical Society. B39 (1) (1977) 1–38
5. Nickerson, A., Japkowicz, N., Milios, E.: Using Unsupervised Learning to Guide Resampling in Imbalanced Data Sets. In: Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics. (2001)
6. Kubat, M., Holte, R., Matwin, S.,: Learning when Negative Examples Abound. In: Proceedings of ECML-97, Vol. 1224. Springer Verlag, (1997) 146–153
7. Kubat, M., Matwin, S.,: Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. In: Proceedings of 14th International Conference on Machine Learning, (1997) 179–186
8. Metz, C.: Fundamental ROC analysis. In: Beutel, J., Kundel, H., MetterHandbook, R. (eds.): Medical Imaging, Vol. 1. SPIE Press, Bellingham, WA (2000) 751–769
9. Domingos, P.: Unifying Instance-Based and Rule-Based Induction. In: Machine Learning, Vol. 24, No. 2. (1996) 141–168