
Guiding Class Selection for an Artificial Nose

Rachel Lomasky
Computer Science Department
Tufts University
Medford, MA 02155
rachel.lomasky@tufts.edu

Carla E. Brodley
Computer Science Department
Tufts University
Medford, MA 02155
brodley@cs.tufts.edu

Matthew Aernecke
Chemistry Department
Tufts University
Medford, MA 02155
matthew.aernecke@tufts.edu

Sandra Bencic
Chemistry Department
Tufts University
Medford, MA 02155
sandra.bencic@tufts.edu

David Walt
Chemistry Department
Tufts University
Medford, MA 02155
david.walt@tufts.edu

Abstract

For some supervised learning tasks, researchers can control the data generation process. In such cases, it would be beneficial to have feedback during learning to guide future data collection. Our research is motivated by a real-world problem: discrimination of vapors with an “artificial nose”. The nose’s accuracy is vital, because it will be deployed to detect harmful gases in critical situations, such as an airport or a subway. We address how to improve accuracy if insufficient examples have been observed to accurately define the class’s decision boundaries. This problem differs from situations in which active learning is applicable. Active learning either requests labels for existing data or explicitly queries the feature space. In contrast, our task allows us to ask for additional examples from specific classes. In this paper we propose an adaptive heuristic to identify from which classes instances should be added during the learning process. We evaluate our methods on the artificial nose data and show significant improvement over random sampling.

1 Introduction

Our research is motivated by a real-world problem: discrimination of vapors with an “artificial nose.” The goal is to build a classifier to distinguish vapors with an array of sensors. For example, a nose could be installed in a subway system and sound an alarm if it detects a nerve agent. No fixed set of classes exist. Rather the nose is a general purpose device that

can be trained to discriminate the k vapors of interest for the location in which it is to be placed. The accuracy is a function of how well the sensors are able to differentiate between the substances of interest. Because of the cost of creating data, we seek to optimize the selection of training data.

The training process is coordinated by software we have developed, called the Nose Data Analyzer. The priorities regarding which groups of vapors for which to increase the accuracy are dictated by real-world constraints, such as the prevalence and toxicity of various poisonous gases. During data collection, the chemists can control the classes from which samples are generated. The software assesses which classes are confused due to volatile boundaries in the current model, and uses heuristic methods to suggest which vapors should be experimented with next. The goal is to maximize accuracy for a target group of vapors.

Since we have begun collecting data, sensors have been invented and newer training examples may contain readings from these new sensors as additional features. Similarly, older sensors may have been retired. Therefore, we have a dynamic feature set. Our system handles the dynamic dataset behind-the-scenes, freeing up the chemists to gather data, and reducing the possibility of human error. We do this by employing a relational database with procedural code that generates datasets in machine learning format.

This paper examines the opportunities and challenges available when a class distribution can be specified for data collection. Selecting training data by class resembles active learning, however, they are significantly different. Although active learning [3] is an incremental learning approach, it addresses requesting labels for data [5] or explicitly querying the feature space [4]. For the nose, querying the feature space is not possible because no pool of unlabeled examples exists. Training data for the artificial nose comes from measurements of the vapors emanating from particular substances. Each measurement is labeled with the corresponding vapor being tested. One cannot separate the process of gathering data from the process of obtaining labels. Thus, rather than guiding which instances to label, our method indicates the classes would most benefit from additional training instances.

2 Generating Training Data for the Artificial Nose

The artificial nose dataset consists of a series of physical experiments in which vapors were passed over an array of sensors. A vapor is pulsed at the sensors, and the sensors react by glowing. Ideally, each sensor type will have a unique response to each vapor. The reactions of each sensor are recorded as a time series of each sensor's reading as the vapor passes over it. For redundancy, each sensor is replicated and the results of sensors of the same type are averaged to create a time-series. An array is re-usable, so there will be hundreds of samples with the same feature set.

When we started gathering data, there were four sensor types [2]. Currently, there are twenty-five types of sensors. Several hundred sensors (beads) are arranged in an array. The beads fluoresce, and wave length of the light with which this is detected is a parameter of the experimental protocol. Additionally, the experiments are conducted with various pulse lengths, which is the length of time the vapor is wafted over the sensors. Because training data was obtained with a variety of sets of sensors, different samples contain different feature values. We currently only train on beads for which we have data for all of the selected vapors. This has the effect of limiting our training data. We hope in the future to explore ways of allowing all of the instances to be considered.

The Nose Data Analyzer (NDA) serves many purposes, including organizing and indexing the large corpus of training data. The front-end is a user-friendly GUI that allows data to be added to the system quickly. For example, an experiment of about 12 million sensor readings takes under an hour to import. The back-end is implemented in Microsoft SQL

Server 2000 to allow for a scalable solution.

To create a nose, the chemist selects which sensors and which vapors are of interest. The NDA allows the selected data to be exported in a format which can be analyzed by the Weka suite of machine learning tools [6]. Additional filters on the features, wave length and pulse length, are also available. The NDA provides the chemists with quick feedback on models built from the training data. Previously data assembly took place in Microsoft Excel, so this is a significant improvement.

However, the overwhelming benefit of the software lies in its ability to recommend for a given task, the set of vapors for the next experiment. Our method, Redistricting, predicts the proportion in which to collect additional instances of the selected classes and displays the results for the chemists. Because of physical limitations with the vapor equipment, we can only sample seven vapors at a time. Therefore, the predictions for the next experiment consist of seven vapors. Which seven vapors are of the highest priority may change over time as more data is collected.

3 Redistricting

Intelligent sample generation will allow the optimal use of training resources. Experiments have a high cost in both time and money. Approximately one hundred examples of seven vapor types can be gathered in a given day. Incorrect suggestions could actually hinder the nose training process, in addition to wasting time. Our method, Redistricting, determines online which classes (and in what proportions) should be sampled from next. There are no limitations in generating instances of a particular class, and we assume that the costs for generating instances are identical for each class. We start with the training data that the chemists generated previously. Training sets for each vapor group will be of variable size and class distributions, depending on the class distribution of previous data collection.

We split the available data into a training set, tuning set, and a “hold-out set” that is uniformly distributed with respect to class. Then, we run K-Nearest Neighbors with $k=3$ on the training data, and note how each instance from the tuning set is classified. We retrain the classifier using the training data and the hold-out set, and examine the tuning instances’ new classifications. If the classification boundaries are volatile, the tuning instances classifications will have changed. This entire process is repeated with five-fold cross-validation.

The number of instances added from each class will be proportional to how many tuning instances were redistricted into or out of it, and the proportion of the tuning set that it represents. We consider the classes’ proportions in the tuning set to keep small classes from being ignored and large classes from being overemphasized. Instances are added according to the following formula, where c is a class from the set of all classes in the dataset, a_c is the number of instances of c to add, p_c is the proportion of c in tuning set, n is the total number of instances from the tuning set that were redistricted, r_c is the number of instances of c that were redistricted, and b is the number of new training instances we wish to gather in that batch;

$$\forall c \in \text{classes}(a_c = p_c * (r_c/n) * b) \tag{1}$$

For example, in Figure 1, the decision boundary between “triangles” and “stars” in the tuning set moved after adding the hold-out set. As a result, the circled triangle is no longer classified as a star, thus, it has been “redistricted.” Although information may be gleaned from examining whether the redistricted instance’s new class prediction is correct or incorrect our method only determines if it is different than the first prediction. If the classification of a redistricted instance changed with the addition of more data, it must be near a relatively volatile decision boundary. Therefore, we would like more instances of triangles

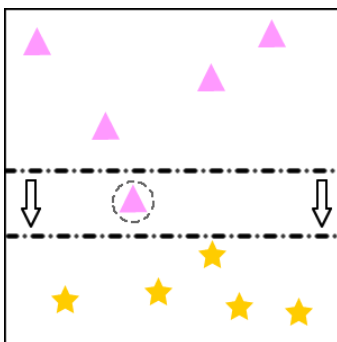


Figure 1: After adding data, the decision boundary has shifted in the tuning set causing the predicted classification of the circled instance to change from a star to a triangle. The circled instance is a “redistricted” instance.

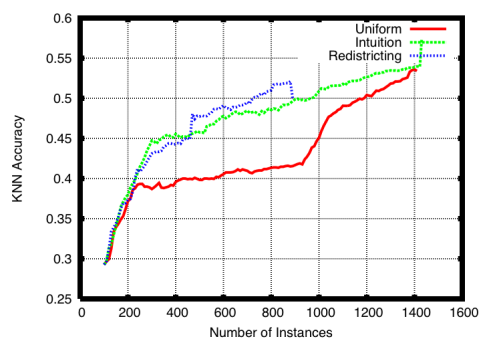
and stars, to further refine the boundary between them. Note that when we generate data for a particular class, we have no control over whether it is actually near class boundaries.

4 Experiments

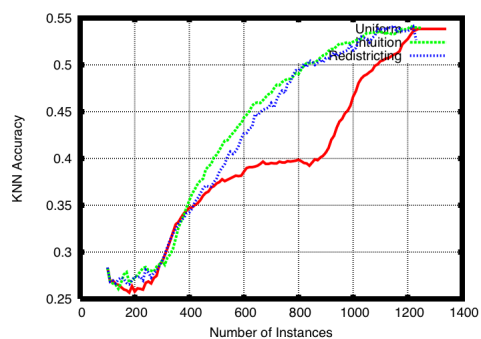
Currently, experiments are being conducted using Redistricting. Before deploying the Nose Data Analyzer to guide the training data generation, we performed the following experiments to validate our method. The chemists on our team generated three datasets that “made sense chemically,” had poor accuracies, and a relatively large number of training examples. Each of these data sets had a total of sixteen classes. Nose Dataset 1 (ND1) consists of 4-nitrotoluene, 2-Chloroethylether, Decanol, Methyl salicylate, Benzene, Benzaldehyde, Propanol, combinations of these vapors, and air. Nose Dataset 2 (ND2) consists of Toluene, Dimethylmethlphosponate, ethanol, heptane, p-cymene, Isopropenyl acetate, combinations of these vapors, water, and air. Nose Dataset 3 (ND3) consists of Diethylmalonate, Decylaldehyde, 2-propanol, Hexane, Benzyl benzoate, Ethyl benzene, n-butanol, combinations of these vapors, and air.

We compare two other sampling methods to Redistricting. With uniform sampling, we generate the same number of samples from each class. In the absence of additional knowledge about our data, this the best we can do. This gives us the best possible information on the distribution of each class so we can make further recommendations. The second sampling method uses the proportions the chemists collected using their intuition. The chemists use their knowledge about the chemical properties of the vapors to determine which ones likely need to be sampled at a higher rate because of inherent variability. Because they work with chemicals for which they have previous knowledge, they are able to make intelligent decisions at the outset of data collection, while Redistricting is still learning about the various classes. In contrast, uniform selection is not able to leverage any knowledge about the class boundaries.

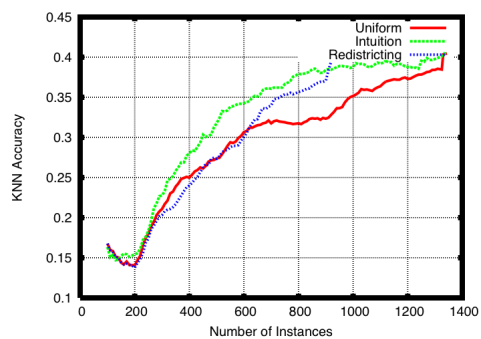
Figure 2 shows the results for each of the three data sets. Note that for some vapor classification problems, the artificial nose can achieve much higher accuracy than is shown in the figure (up to almost 100% [1]). The accuracy depends on how easy the chosen set of vapors are to discriminate and how much training data is available. In Figure 2, the x-axis represents the number of instances. The y-axis is the accuracy obtained with the nearest neighbor algorithm, with $k = 3$. We choose nearest neighbor because we assume that each vapor should be classified the same as those with the most similar readings for each point in the time series. We set $k = 3$ to counteract the noise in the feature values. We



(a) ND1



(b) ND2



(c) ND3

Figure 2: We show uniform sampling, Redistricting, and the data gathered according to the chemists' intuition. Each experiment began with 100 instances and had 10 instances added as suggested by Redistricting.

use one hundred instances as the “initial data collection” because the chemists believe it is sufficient to make the initial Redistricting decisions. We then add ten instances at a time, which is feasible in practice because of the chemists’ ability to alter the experiment while it’s running. As more data is added, all methods have seen the same data, and the results converge. Note that we are in the process of collecting more data. Each set of one hundred instances takes a full day to collect.

Results vary in the three datasets selected. In Figure 2(a) which illustrates the results from ND1, Redistricting performs equal to or better than the intuition of the chemists. This amounts to over 10% increase in accuracy over uniform sampling. Similarly, Figure 2(b) also shows how Redistricting rivals the scientists’ intuitions for ND2, and performs over 10% better than uniform sampling. In the final dataset, ND3, uniform sampling, Redistricting, and data gathered according the chemists’ intuitions all result in classifiers that perform comparably. This is because all three make similar choices.

5 Conclusion

Our system allows for a flexible, robust, and easy-to-use method of obtaining training data. We are able to solve the challenges presented by dynamic class and feature sets. The flexibility of the system allows an online assessment of the nose’s classification abilities and allows the chemists to direct their resources towards the area of largest benefit. Our training methodology was implemented recently. Therefore, there is much future work in assessing the efficacy of Redistricting. In addition, the artificial nose is still being trained in the laboratory, so we do not have feedback on its performance in a real-world situation.

A large open question is the best design of the sensor arrays and experimental protocols. Ideally, a sensor array would contain beads that would give a unique response for each vapor. We hope to determine which set of beads will optimize this goal. In addition, we would also like to determine in which proportion those beads should be present, based on their variability. Also, we would like to discover how short we can make the pulse length without sacrificing information. If the pulse length is shorter, more experiments can be conducted in a given time period. Lastly, we would like to experiment with the wave length to determine which gives the best training instances with the least variability.

References

- [1] S. Bencic, T. Sternfeld, and D. Walt. Microbead chemical switches: An approach to detection of reactive organophosphate chemical warfare agent vapors. *J. Am. Chem. Soc.*, 128:5041–5048, 2006.
- [2] S. Bencic-Nagale and D. R. Walt. Extending the longevity of fluorescence-based sensor arrays using adaptive exposure. *Analytical Chemistry*, 77(19):6155–6162, 2005.
- [3] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Mach. Learn.*, 15(2):201–221, 1994.
- [4] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Advances in Neural Information Processing Systems*, 7:705–712, 1995.
- [5] T. Luo, K. Kramer, D. B. Goldgof, L. O. Hall, S. Samson, A. Remsen, and T. Hopkins. Active learning to recognize multiple types of plankton. *JMLR*, 6:589–613, 2005.
- [6] I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham. *Weka: Practical machine learning tools and techniques with java implementations*, 1999.