Extensions to the Active Testing Algorithm*

John Holodnak, Jensen Dempsey, Carmen Stowe, Adam Tse
MIT Lincoln Laboratory

January 22, 2020

1 Introduction

For many classification problems, high quality labeled datasets do not exist. As a result, to train and test classifiers, we are forced to use automated or crowdsourced methods to obtain labeled datasets. It is well known that crowdsourced labels are often low quality. As a result, evaluations performed with these "noisy" labels may be correspondingly inaccurate. The work by Nguyen et al. [2] seeks to mitigate this problem by updating ("vetting") the noisy test set labels on only the most informative items in the test data set, so as to get improved performance estimates with as small a labeling budget as possible. They refer to their approach as "active testing." The authors discuss several methodologies to select the most informative items.

We provide an update to the active testing algorithm to address the issue that the vetter may not be completely accurate either. The existing work on active testing uses the vetted labels to learn a model of the noisy test set labels and the classifier predictions. We propose using a label aggregation technique by Dawid and Skene [1] to estimate the correct label for each item using the noisy labels, the classifier labels, and the vetted labels. This aggregation technique does not assume that any of the label sources is completely accurate.

The following sections explain active testing and our extension in more detail and provide comparisons of different active testing methodologies on simulated and real datasets.

2 Overview of Active Testing

In active testing, we are given a trained classifier with the goal of estimating the performance of the classifier on some fixed test set X, by actively querying a "vetter," who provides a label for the queried items. We assume that each item $x_i \in X$, $1 \leq i \leq N$, has at least one "noisy" label (meaning

This material is based upon work supported under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force.

we do not trust that it is correct) and a prediction from the classifier. We denote the noisy labels as $y_i^k, 1 \leq i \leq N, 1 \leq k \leq K$, where K is the number of "experts" providing labels. We assume that the classifier prediction is in the form of a score distribution $s_i = (s_{i1}, \ldots, s_{iL}), 1 \leq i \leq N$, over the L possible labels. We want to obtain an accurate estimate of classifier performance, while querying the vetter as few times as possible.

Active testing has two basic steps, querying items for the vetter to label and then estimating the true label for each item in the test set. Depending on the query strategy, we may either query and label all items prior to the estimation step, or iteratively query items, update our estimated true labels and then use this to query more items. We notate our current estimate of the ground truth for the labels of the items in the test set as $p(z|\mathcal{P})$, where $\mathcal{P}$ represents all the information that we use to build our estimate. For example, we may use the noisy labels y, the vetted label (if available) v, and the classifier score distribution s. When we are finished querying items, we use the classifier score distributions and our estimate of the ground truth to approximate the performance metrics of interest. For example, to estimate accuracy, we simply average the probabilities associated with the classifier's predictions:

$$\frac{1}{N}\sum_{i=1}^{N} p(z_i = \mathsf{argmax}\{s_i\}|\mathcal{P}).$$

More generally, we can repeatedly sample z_i from $p(z|\mathcal{P})$ and compute the metric of interest using the classifier predictions and z_i.

Nguyen et al. [2] defined a variety of estimation strategies to learn $P(z|\mathcal{P})$, such as doing the evaluation only with the vetted items or using noisy labels for the items that are not vetted. In this paper, we will work with a general form of the so-called "learned" estimator, which typically performed the best. The learned estimator uses the noisy labels, the vetted labels and the classifier scores to estimate the true labels. To see how, note the factorization below, assuming the classifier scores are conditionally independent of the noisy labels,

$$\begin{aligned} p(z|y,s) &\propto p(y,s|z)p(z) \\ &\propto p(y|z)p(s|z)p(z) \\ &\propto p(y|z)p(z|s). \end{aligned}$$

This factorization conveniently separates the categorical noisy labels from the continuous scores. If we assume that the vetted labels are correct, we can estimate the entries of the confusion matrix $p(y = \ell_1|z = \ell_2),\ 1 \leq \ell_1 \leq L,\ 1 \leq \ell_2 \leq L$, and estimate $p(z|s)$ via logistic regression. If there is more than one expert and we assume the experts are conditionally independent, given the true label, then the factorization can be written as

$$p(z|y,s) \propto p(y^1|z) \ldots p(y^K|z)p(z|s).$$

Note that when there are a large number of classes, estimating the full confusion matrix for each expert will introduce many parameters. In this case, we can consider simpler models, such as assuming a symmetric confusion matrix where the diagonal elements are modeled and we assume the remaining probability mass is distributed evenly across the other classes. Even more simply, we can further assume that all diagonal elements are equal and so model each expert with a single parameter. (To see examples of these confusion matrices in the case where there are three classes, see Figure 1.)

$$\begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{bmatrix}, \begin{bmatrix} \theta_1 & (1-\theta_1)/2 & (1-\theta_1)/2 \\ (1-\theta_2)/2 & \theta_2 & (1-\theta_2)/2 \\ (1-\theta_3)/2 & (1-\theta_3)/2 & \theta_3 \end{bmatrix}, \begin{bmatrix} \theta & (1-\theta)/2 & (1-\theta)/2 \\ (1-\theta)/2 & \theta & (1-\theta)/2 \\ (1-\theta)/2 & (1-\theta)/2 & \theta \end{bmatrix}.$$

Figure 1: Full confusion matrix, symmetric confusion matrix, and one parameter confusion matrix for three classes.

Having discussed the estimation step of the active testing algorithm, we now turn towards the item query step. There are a variety of ways to query the items to vet. We can, of course, query items at random, or use a variety of active learning inspired strategies, such as querying the items according to a distribution related to the classifier's uncertainty, quantified as the entropy of the score distribution

$$p_i \propto -\sum_{\ell=1}^{L} s_{i\ell} \ln(s_{i\ell}).$$

In the case where there is more than one noisy label per item, we can also compute the entropy of the expert vote distribution. Specifically,

$$p_i \propto -\sum_{\ell=1}^{L} \frac{V_{i\ell}}{T_i} \ln\left(\frac{V_{i\ell}}{T_i}\right)$$

where $V_{i\ell}$ is the number of votes for label ℓ on item i and T_i is the total number of votes for item i. The idea behind these strategies is that we update the labels of items about which either the experts or the classifier were very uncertain. In theory, these are the instances that are hardest to label.

Nguyen et al. [2] consider two other query strategies that they term Most Confident Mistake and Maximum Expected Estimator Change. In Most Confident Mistake, we query items that have the largest discrepancy between the classifier predicted scores and the ground truth estimates. This query strategy requires updating the estimated ground truth distribution $p(z|\mathcal{P})$ periodically and is thus more computationally expensive than the strategies above. In Maximum Expected Estimator Change, we query the item that, in expectation, will most affect the performance metric estimate. The authors derive a formula for the item in the case of a few metrics, but in the absence of a specific formula, this is very computationally expensive, as it requires iterating through the dataset and retraining the classifier for each unvetted item to determine which item provides the largest expected change to the performance metric. Due to its lack of ability to be easily generalized, we do not consider it further in this work.

2.1 Simulations to compare query strategies

We now briefly compare the query strategies on simulated datasets. We generate a dataset of 10,000 items in $\mathcal{R}^{10}$ with four classes, each of which consists of four distinct clusters.[1] We use 75% of the dataset to train a simple feed-forward neural network. Our goal is to estimate the performance of the classifier, using active testing, on the remaining 2500 instances. We generate three noisy

[1] We used the make_classification function from scikit-learn `https://scikit-learn.org`.

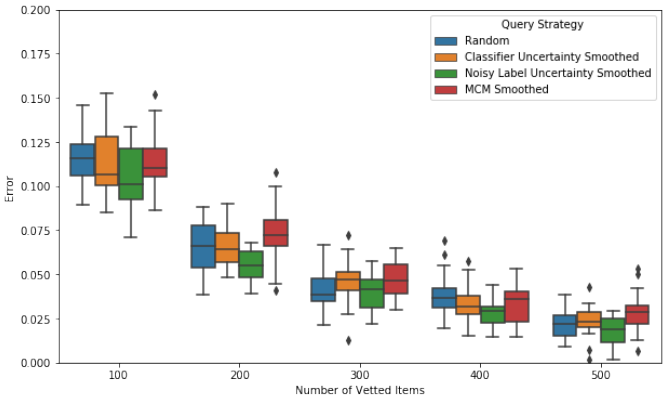

Figure 2: Box plots of the estimator error against the number of vetted items on a simulated dataset with three experts for several query strategies, over 25 trials.

labels for each data item, using the following expert confusion matrices, where the (i, j) entry is the probability that the expert chooses label j when the true class is i:

$$\begin{bmatrix} 0.3 & 0.3 & 0.2 & 0.2 \\ 0.1 & 0.4 & 0.1 & 0.4 \\ 0.3 & 0.1 & 0.5 & 0.1 \\ 0.3 & 0.1 & 0.0 & 0.6 \end{bmatrix}, \begin{bmatrix} 0.3 & 0.0 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.1 & 0.2 \\ 0.1 & 0.2 & 0.5 & 0.2 \\ 0.1 & 0.2 & 0.1 & 0.6 \end{bmatrix}, \begin{bmatrix} 0.3 & 0.3 & 0.3 & 0.1 \\ 0.2 & 0.4 & 0.2 & 0.2 \\ 0.2 & 0.3 & 0.5 & 0.0 \\ 0.2 & 0.2 & 0.0 & 0.6 \end{bmatrix}.$$

Note that the experts make asymmetric errors. Each expert has accuracy 0.3 on class 1, 0.4 on class 2, 0.5 on class 3, and 0.6 on class 4. However, their errors are distributed differently to the remaining classes.

In Figure 2, we compare the error of random sampling, sampling according to the uncertainty of the classifier, sampling according to the expert labels, and using the Most Confident Mistake strategy. We see there is not much difference between the query strategies. As the number of samples increases, estimates using all query strategies improve. We interpret that the learned estimator does not strongly depend on the samples provided to it in this problem.

3 Supervised label estimation compared to unsupervised label estimation

The use of the vetted labels in the learned estimator implicitly assumes that the vetted labels are correct. In some situations this may be a reasonable assumption, but for difficult problems, the vetter may also not be correct. We propose a relatively simple extension to existing work on active testing that eliminates the assumption that the vetter provides the correct label. Instead of using the supervised approach described above, we will use an approach that estimates label probabilities using a set of categorical predictors (the expert, vetter, and classifier labels). To do this, we use a classic model proposed by Dawid and Skene [1]. The model assumes that each labeler independently labels each data point, given the true label, according to their own confusion matrix. As a result,

the vetter is not assumed to have perfect accuracy. The likelihood of the model is as follows:

$$P(y^c, y^v, y^1, \ldots, y^k | \Theta^c, \Theta^v, \Theta^1, \ldots, \Theta^m) = \prod_{i=1}^{N} \sum_{\ell=1}^{L} p_\ell \left(\theta^c_{\ell y_i^c} \theta^v_{\ell y_i^v} \prod_{k=1}^{K} \theta^k_{\ell y_i^k} \right).$$

In the above, p_ℓ is the prior probability of class ℓ and $\theta^k_{\ell y_i^k}$ represents the probability that expert k chooses label y_i^k when the true class is ℓ. Similarly, $\theta^c_{\ell y_i^c}$ and $\theta^v_{\ell y_i^v}$ represent the parallel quantities for the classifier and vetter. To be clear, the separation in notation is to conform to our setup; the labels of the classifier and vetter are not treated differently than the labels from the experts. Dawid and Skene derive the Expectation Maximization (EM) updates for this model. We will apply it assuming that each expert's confusion matrix is modeled by a single parameter.

3.1 Simulations

A natural comparison of these two approaches is to look at the error in estimating a classifier's performance as the error of the vetter increases. For the supervised method, we will also compare the three different models of the confusion matrix that we discussed above, full, diagonal, and one parameter.

In these next simulations, we generate a dataset and train a classifier as described in Section 2.1. We first consider the case where one expert label is available per data item and allow a budget of 500 queries. The confusion matrix of this expert is set to

$$\begin{bmatrix} 0.3 & 0.3 & 0.2 & 0.2 \\ 0.1 & 0.4 & 0.1 & 0.4 \\ 0.3 & 0.1 & 0.5 & 0.1 \\ 0.3 & 0.1 & 0.0 & 0.6 \end{bmatrix}.$$

Note that the expert's performance varies across the classes and that the errors are not evenly distributed. In Figure 3, we see the results of running the supervised method with the full, diagonal, and one parameter models for the expert confusion matrix, as well as the results for the unsupervised model. Except when the vetter makes no errors, the unsupervised method performs better. There appear to be only small differences between the supervised method using the different confusion matrix models.

We also compare the methods when we have three experts with the confusion matrices from Section 2.1. In Figure 4, we see that the unsupervised method outperforms the supervised methods for all vetter accuracies (apart from some large outliers). It is curious that even when the vetter makes no mistakes, the unsupervised method still performs better. One difference between the methods that could make a difference is that the unsupervised method creates a hard label from the classifier score, while the supervised method uses the score directly. If the model is highly accurate, but the scores are not all close to zero or one, then the hard label could conceivably perform better.

Note that the unsupervised method does however, exhibit much more variance when there is only a single expert and has some large outliers (not shown). These could likely be smoothed out by running the EM algorithm with several starting points and selecting the one with the highest likelihood. In addition, the EM algorithm performs better as more predictions per item are added to the model. This is clear from our two simulations, as the variance of the unsupervised method is much reduced in the case where there are three experts, as opposed to just one.

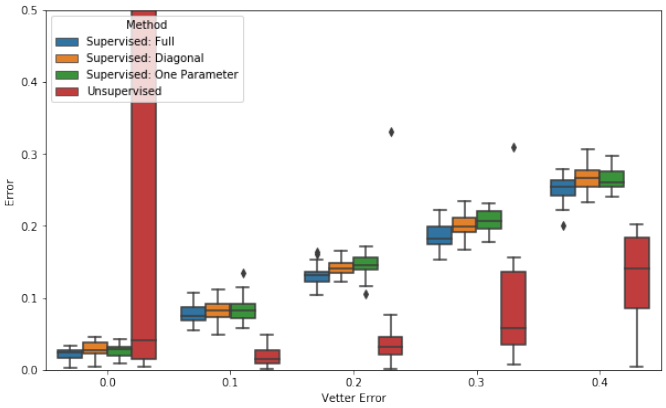

Figure 3: Boxplots of the estimator error against vetter error on a simulated dataset with one expert for the supervised and unsupervised label estimation methods, over 25 trials. The unsupervised estimator has some large outliers (error greater than 0.5) that are not shown so that the rest of the plot is more visible.

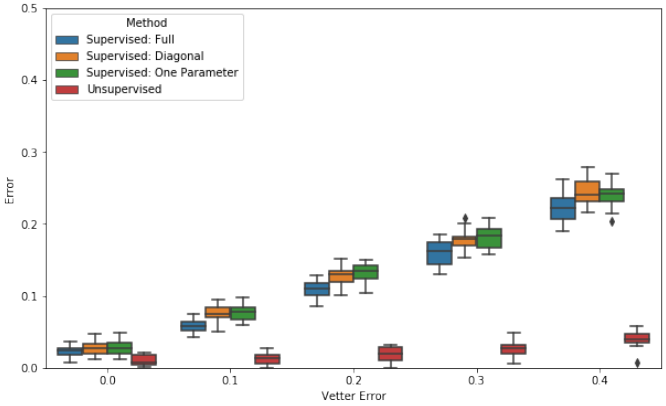

Figure 4: Boxplots of the estimator error against vetter error on a simulated dataset with three experts for the supervised and unsupervised label estimation methods, over 25 trials. The unsupervised estimator has some large outliers (error greater than 0.5) that are not shown so that the rest of the plot is more visible.

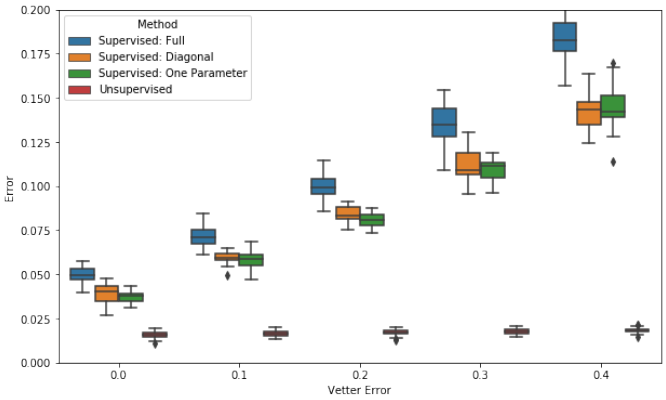

Figure 5: Boxplots of estimator error against vetter error on the MNIST Fashion dataset for the supervised and unsupervised label estimation methods, over 25 trials.

3.2 Experiments

In this section, we carry out a similar experiment using the MNIST Fashion dataset [3], which contains ten classes. Here we train a simple neural network as our classifier and generate simulated labels for three experts. The experts have one-parameter confusion matrices with parameters 0.75, 0.5, and 0.25. As before, we varied the performance of the vetter, which labeled 500 items. In Figure 5, we see that the performance of the unsupervised method is uniformly better than the supervised methods. As we might expect, the errors of the supervised method with the full confusion matrix are higher than with the diagonal or one parameter confusion matrices. This is not surprising, as the ten class dataset will have 90 free parameters for each confusion matrix, which are fit using the 500 labels from the vetter.

4 Conclusion and Future Work

In this paper, we have discussed the use of an unsupervised label estimation strategy within the active testing algorithm. This allows us to eliminate the assumption in active testing that the vetter is correct. On a mix of simulated and real datasets, we showed that the error of the supervised methods tend to noticeably increase as the error of the vetter increases, but that the error of the unsupervised methods increases much more slowly, or does not change at all.

In the future, we plan to continue to investigate the performance of the unsupervised method against the supervised methods, as the performance of the methods when the vetter is completely accurate did not match our initial expectations. In addition, we also plan to consider query strategies that make use of the feature space, in addition to the noisy labels and classifier predictions.

References

[1] P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied Statistics*, pages 20–28, 1979.

[2] Phuc Nguyen, Deva Ramanan, and Charless Fowlkes. Active testing: An efficient and robust framework for estimating accuracy. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3759–3768, 2018.

[3] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.