Towards Online PU-Learning

Carter Blair

April 2022

1 Abstract

A significant part of PU-learning is mixture proportion estimation (MPE). MPE is the task of estimating the proportion of positives in the unlabeled data. This paper proposes a new method (TOM) for MPE that relies on training and using a non-traditional classifier on positive and unlabeled data. The method is shown to outperform previous algorithms on MPE for Gaussian data for various values of α and β. In addition to this, a method (TOM-ON) for MPE, that runs in constant time, is proposed and is shown to successfully iteratively estimate the mixture proportion of unlabeled samples from a stream of data.

2 Introduction

Learning a PvN classifier from positive and unlabeled data is known as PU-learning. It is often stated that PU-learning can be split up into two parts: (1) Mixture proportion estimation, which is characterized by estimating the proportion of positives in the unlabeled data (α), and (2) using this estimate to learn a PvN classifier [1]. Much attention has been paid to the latter task, but often the former task, MPE, gets less attention. This is especially apparent in the emerging niche of online PU-learning, and, in fact, there are no known methods for iteratively estimating the mixture proportion.

Previous papers that have addressed online PU-learning typically assume a known mixture proportion and rely on this to update their PvN classifier incrementally, given a stream of positive and unlabeled data [2]. However, in real-world settings, the mixture proportion of the unlabeled data is often unknown. This presents a significant challenge for methods that rely on the assumption that the mixture proportion is known. In the face of this problem, such methods would have to guess the mixture proportion, which could have detrimental effects on the PvN classifier if the guess is inaccurate. To address this issue, I present a method (TOM-ON) that can iteratively estimate the mixture proportion given a stream of positive and unlabeled data, given a non-traditional classifier (NTC). The method, TOM-ON, does not update the NTC or the PvN classifier, but in conjunction with other methods such as OPU [2], fully online PU-learning could be achieved.

TOM-ON is based upon another method developed in this paper named TOM for Threshold Optimization MPE. The general intuition behind TOM is that the probabilities produced by the NTC for unlabeled samples that are truly positive will be similar to the probabilities produced by the NTC for positive samples. And on the other hand, probabilities produced by the NTC for unlabeled examples that are truly negative will look different than probabilities for positive examples. With this in mind, there should be some threshold such that probabilities above the threshold correspond to truly positive examples and probabilities below the threshold correspond to truly negative samples. TOM finds the optimal threshold for this task and uses it to classify unlabeled samples as positive or negative, then uses the counts of unlabeled data points classified as positive and negative to estimate the mixture proportion.

To find the optimal threshold, TOM iterates through a finite set of thresholds and finds the percent of positive probabilities above the threshold and the percent of unlabeled probabilities above the threshold. The threshold that produces the biggest difference between the percent of positive examples above the threshold and unlabeled examples above the threshold is chosen. If this method were used to update the mixture proportion estimation iteratively, the resulting method would be $O(n^2)$ because, for each of the n data points in the stream, TOM would need to be run to find an updated mixture proportion. TOM itself runs in $O(n)$ time due to the fact that it iterates through all of the previously seen probabilities.

TOM-ON is similar to TOM but runs in constant time at the expense of some accuracy. Specifically, instead of iterating through all of the previously seen probabilities to find the optimal threshold as TOM does, TOM-ON places the probabilities into bins as they come and estimates the optimal threshold using the bins of probabilities. TOM-ON then uses the optimal threshold estimate to estimate the mixture proportion just as TOM does with the true optimal threshold. TOM-ON processes the data as it comes and runs in constant time, thus making it possible to do mixture proportion estimation in $O(n)$ time, where n is the size of the stream.

3 My Batch Method (TOM)

This section will introduce TOM, a method for estimating the mixture proportion and is well suited to the batch setting. TOM relies on a pre-trained NTC, and thus it is assumed that we have access to this. In the setting focused on in this paper, namely classifying one-dimensional Gaussian data points, something as simple as a logistic regression classifier trained on a set of positive examples and a set of unlabeled examples is sufficient for the NTC.

The idea behind TOM is as follows: Given a test set of positive examples, a test set of unlabeled examples, and a classifier f, begin by running the samples through the classifier and keep track of the resulting probabilities for the positive set and the unlabeled set. The result of this can be visualized as in figure 1.

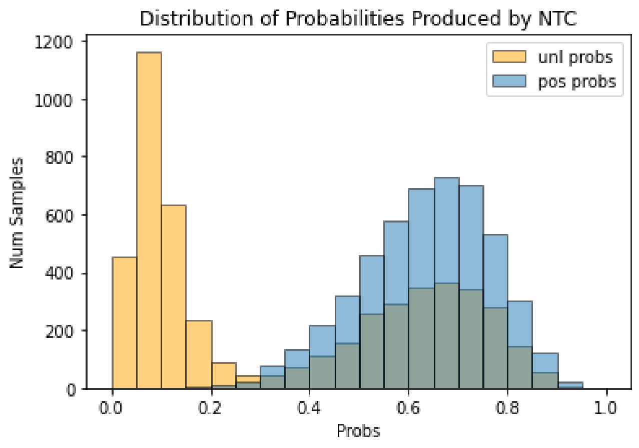


Figure 1: Probabilities produced by NTC where $\alpha, \beta = 0.5$, $\mu_p = 8$, $\mu_n = 3$, and $n_p, n_u = 5000$.

The histogram in figure 1 is a rough estimate of the distribution of the probabilities obtained from running the samples through the NTC. This example shows that the positive probabilities are unimodally distributed, and the unlabeled probabilities are bimodally distributed. The right mode of the unlabeled distribution is the same as the mode from the distribution of positive probabilities, and the left mode is vastly different.

Since the data set used in this experiment was contrived, we can look at the probability distributions of the positive unlabeled samples and the negative unlabeled samples. This is visualized below in figure 2.

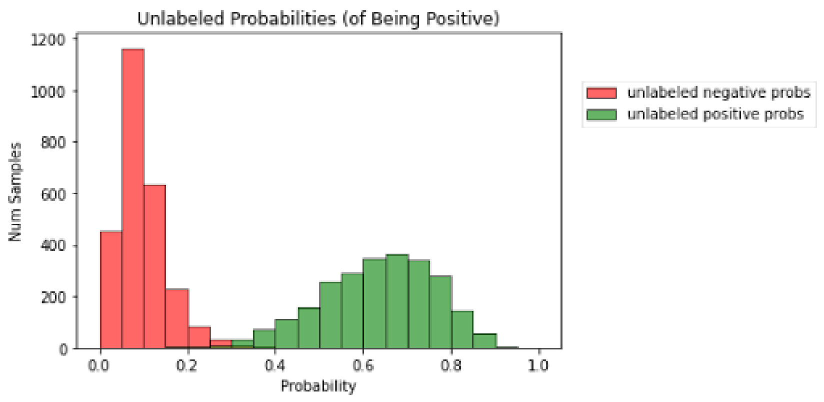


Figure 2: Unlabeled probabilities, same data parameters as described in figure 1.

The visualization in figure 2 allows us to gain some intuition regarding the probabilities produced by the NTC for the unlabeled data set. In particular, the above visualization indicates that unlabeled examples that are truly positive have a higher probability of being positive than unlabeled examples that are truly negative do, even though the NTC is PvU.

With this in mind, the TOM method seeks to find a threshold for the NTC that maximizes the number of positive examples above said threshold while minimizing the number of unlabeled examples above said threshold. To do so, the method searches through a discretized interval of thresholds between 0 and 1 and selects the threshold that maximizes the difference between the percent of positives probabilities above it and the percent of unlabeled probabilities above it. This can be visualized as in figure 3.

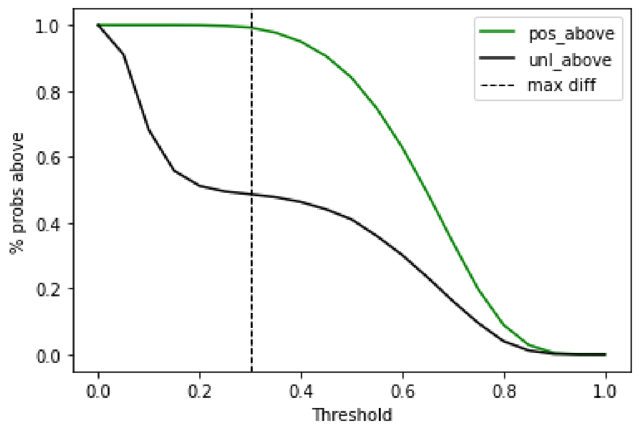
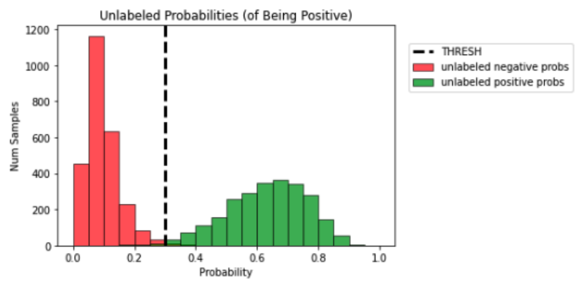


Figure 3: The percent of probabilities above the threshold. Optimal threshold, which maximizes the difference, is represented by the dotted line.

When the difference between the percentage of positive probabilities above the threshold and the percentage of unlabeled probabilities above the threshold is maximized, it maximizes the number of true positives classified as positive and minimizes the number of true negatives classified as positive. The threshold that maximizes this is taken to be the optimal threshold.

With this threshold in hand, TOM then runs through all of the probabilities outputted by the NTC from the unlabeled data set and maintains a count of the probabilities that exceed the optimal threshold determined above. The final count of probabilities that exceeded the threshold is divided by the total number of unlabeled probabilities, and this value is taken as the mixture proportion estimate.



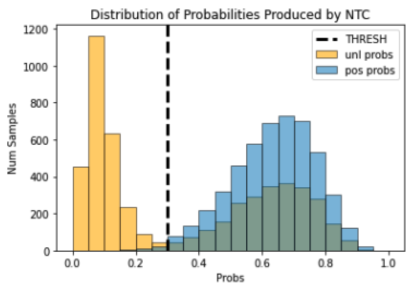


Figure 4: The optimal threshold from the visualization above overlaid on the NTC probability distributions.

With the intuition behind TOM in hand, I will now present the algorithm.

Algorithm 1 Threshold Optimization MPE (TOM)

Input: Validation positive (X_p) and unlabeled (X_u) samples, classifier $\hat{f} : \mathcal{X} \rightarrow [0, 1]$, set of possible thresholds T
Output: Mixture proportion estimate: $\hat{\alpha}$

```
 1: D = {}
 2: $Z_p, Z_u = \hat{f}(X_p), \hat{f}(X_u)$
 3: for $t_i \in T$ do
 4:     $a_p, a_u = \frac{\sum_{z_j \in Z_p} \mathbb{1}[z_j > t_i]}{n_p}, \frac{\sum_{z_j \in Z_u} \mathbb{1}[z_j > t_i]}{n_u}$
 5:     $d_i = a_p - a_u$
 6:     $D = D \cup \{d_i\}$
 7: end for
 8: $i^* = \arg\max_i(D)$
 9: $t^* = t_{i^*}$
10: $\hat{\alpha} = \frac{\sum_{z_j \in Z_u} \mathbb{1}[z_j > t^*]}{n_u}$
11: return $\hat{\alpha}$
```

4 O(n) Version of My Method (TOM-ON)

The TOM method presented above works well in the batch setting but updating the mixture proportion estimate runs in O(n) time due to the fact that for every update, all of the positive probabilities and all of the unlabeled probabilities must be looped through multiple times in order to find the optimal threshold. So, in a stream of data where the goal is to update the mixture proportion estimate after each new data point, the runtime would be O(n^2). The TOM-ON overcomes this issue by placing the probabilities into bins as they come in. The result of this is a histogram of probabilities akin to the one presented in figure 1. Then, instead of finding the optimal threshold by looping through all of the probabilities as happens in TOM, TOM-ON only has to loop through a finite number of bins. Since the probabilities are discretized into bins the resulting threshold is only an estimate of the optimal threshold.

With this update, TOM-ON runs in constant time because, upon receiving an example, a classifier predicts the probability, then the probability is placed into a bin. Then the bins are used to find an optimal threshold, and the probability that the incoming point is positive is compared to the optimal threshold; lastly, the mixture proportion is estimated. The algorithm is presented at the bottom of the paper.

5 Experiments

5.1 Batch Experiments

The TOM method was tested against various known MPE algorithms, such as BBE, DEDPUL, the Elkan and Noto estimator, and the Scott estimator [1].

5.1.1 Experiment 1: Varying α

In experiment 1, all five algorithms were tested for varying values of α, the true mixture proportion. The data set consisted of 2000 positive samples and 2000 unlabeled samples that were split according to the value of α. The true positives were normally distributed with $\mu = 8$ and $\sigma = 1$, and the true negatives were normally distributed with $\mu = 3$ and $\sigma = 1$. The classifier used was a logistic regression classifier trained on a separate data set with the same parameters as the test set. For each value of alpha, each algorithm was run 10 times, and the absolute error was recorded. The plot below in figure 5 shows the average absolute error of each algorithm for various values of α.

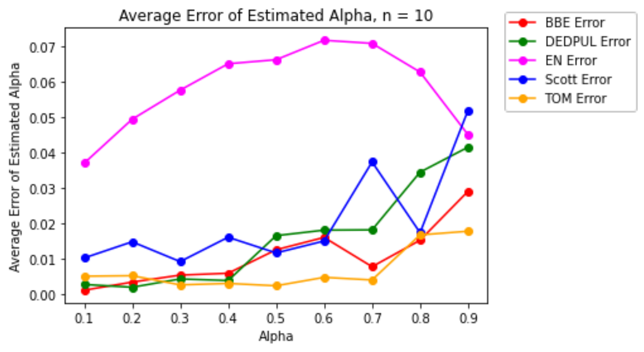


Figure 5: Batch experiment with various values of α

The TOM method was comparable to the best method for all values of α, and outperformed all other methods for 6 of the 9 values of α.

5.1.2 Experiment 2: Varying β

In the second experiment, the algorithms were tested for various values of β. The β value corresponds to the fraction of positive examples in the total data set. For example, for $\beta = 0.4$, 40% of the samples would be positive, and 60% would be unlabeled. For all values of β, α was set to 0.5. The total size of the test set was fixed at 4000 samples, including both positive and unlabeled. The true positives were normally distributed with $\mu = 8$ and $\sigma = 1$, and the true negatives were normally distributed with $\mu = 3$ and $\sigma = 1$. Once again, the classifier was a logistic regression classifier trained on a separate data set with the same parameters as the test set.

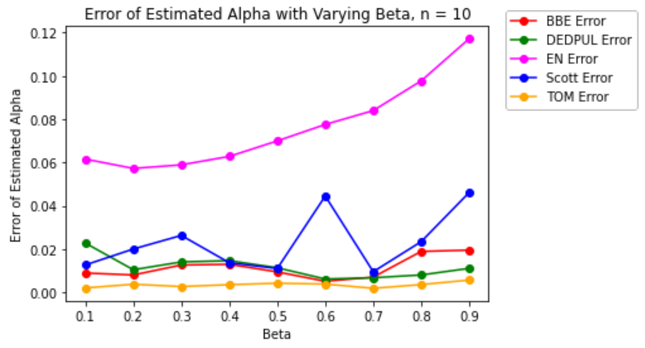


Figure 6: Batch experiment with various values of β

The TOM method outperformed all other methods for all values of β. In experiment 1, where various values of α were tested, TOM significantly outperformed all other methods when α was set to 0.5, so this experiment shows that the performance of TOM is preserved even in the face of varying values of β. However, more experiments are needed in order to determine whether this is true for all values of α.

Taken together, these experiments show that TOM is a method worth further testing. The performance on one-dimensional Gaussian data was shown to be good, but performance could degrade in the face of higher-dimensional data.

5.2 Online Experiments

The TOM-ON method was tested on a dataset consisting of 2000 positive samples and 2000 unlabeled samples where the unlabeled samples were split between positive and negative according to $\alpha = 0.7$. Further, β was set to 0.5. The true positives were normally distributed with $\mu = 8$ and $\sigma = 1$, and the true negatives were normally distributed with $\mu = 3$ and $\sigma = 1$.

The method was successful in predicting the mixture proportion estimate, and the estimate converged after around 750 unlabeled samples were encountered. This can be seen in figure 7 below.

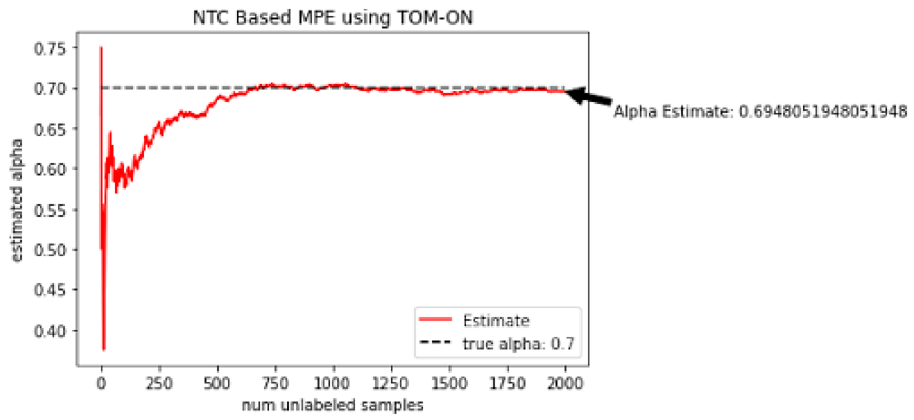


Figure 7: TOM-ON mixture proportion estimate per update (only updated on unlabeled examples).

Since the threshold is updated after each sample is encountered, the thresholds used in each iteration were plotted. The threshold converged after seeing about 200 examples (positive and unlabeled). This is shown below in figure 8.

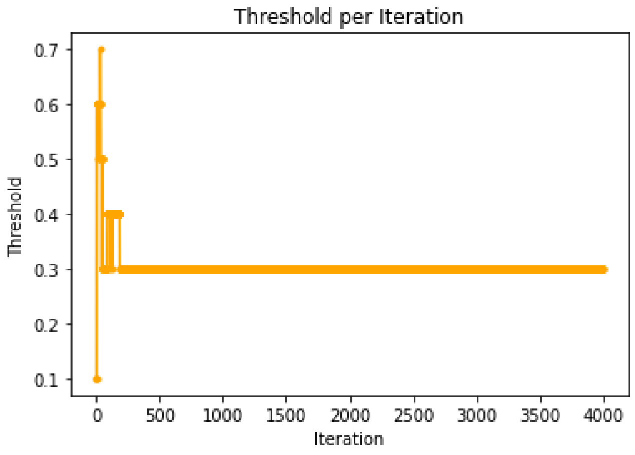


Figure 8: TOM-ON optimal threshold estimate per iteration.

Overall, TOM-ON didn't appear to suffer that much from the binning process, but more experiments are needed to compare this constant time method to its O(n) counterpart TOM.

6 Future Work

The methods developed in this paper, TOM and TOM-ON, performed well on one-dimensional data, but it is well known in the domain of PU-learning that a method's performance on low-dimensional data is not indicative of its performance on high dimensional data [1]. For this reason, future work should investigate the performance of this algorithm on higher-dimensional data.

In addition to this, TOM-ON assumed a pre-trained classifier, but in an online learning context where TOM-ON would likely be used, it is unlikely that there would be access to a pre-trained classifier. This could possibly be detrimental to the performance of TOM-ON since it relies on a relatively accurate classifier to make its mixture proportion estimate. On the other hand, training an online PU-learning algorithm requires access to a reliable mixture proportion estimate. It is unclear if this chicken-and-egg problem will be overcomeable. This question deserves future investigation.

7 Conclusion

In this paper, two new methods for MPE, TOM and TOM-ON, were proposed and experimentally validated. It was found that TOM could successfully estimate the mixture proportion, and further, TOM was found to have state-of-the-art performance on the task of PU-learning one-dimension Gaussian data. TOM-ON, a constant run-time algorithm for MPE, was proposed in order to move closer to a fully online PU-learning algorithm.

Algorithm 2 TOM-ON

Input: Training sample (x, y), classifier $\hat{f} : \mathcal{X} \to [0, 1]$, positive bins $B_p = \{b^p_1, ..., b^p_{10}\}$, unlabeled bins $B_u = \{b^u_1, ..., b^u_{10}\}$, c_p, c_n

Output: Mixture proportion estimate: $\hat{\alpha}$, positive bins $B_p = \{b^p_1, ..., b^p_{10}\}$, unlabeled bins $B_u = \{b^u_1, ..., b^u_{10}\}$, c_p, c_n

```
D = {}
$p = \hat{f}(x)$

if $0 \leq p < 0.1$ then
    $i \leftarrow 1$
else if $0.1 \leq p < 0.2$ then
    $i \leftarrow 2$
    ⋮
else
    $i \leftarrow 10$
end if

if $y = 0$ then
    $b^p_i = b^p_i + 1$
else
    $b^u_i = b^u_i + 1$
end if

for $j = 1, ..., 10$ do
    $a_p = \frac{\sum_{k=j}^{10} b^p_k}{\sum_{k=1}^{10} b^p_k}$

    $a_u = \frac{\sum_{k=j}^{10} b^u_k}{\sum_{k=1}^{10} b^u_k}$

    $d_i = a_p - a_u$
    $D = D \cup \{d_i\}$
end for

$i^* = \arg\max_i(D)$
$t^* = \frac{i^*}{10}$

if $y = 1$ then
    if $p > t^*$ then
        $c_p = c_p + 1$
    else
        $c_n = c_n + 1$
    end if
end if

$\hat{\alpha} = \frac{c_p}{c_p + c_n}$
return $\hat{\alpha}, B_p, B_u, c_p, c_n$
```

References

[1] Garg, Saurabh, Yifan Wu, Alexander J. Smola, Sivaraman Balakrishnan, and Zachary Lipton. Mixture Proportion Estimation and PU Learning: A Modern Approach. In *Advances in Neural Information Processing Systems* 2021.

[2] Zhang, Chuang, Chen Gong, Tengfei Liu, Xun Lu, Weiqiang Wang, and Jian Yang. Online positive and unlabeled learning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence,* 2021.