DeepPUNs : Deep Positive Unlabeled Networks

Gowthami Somepalli[*]
Department of Computer Science
University of Maryland
College Park, MD 20740
gowthami@umd.edu

Neha Kalibhat[*]
Department of Computer Science
University of Maryland
College Park, MD 20740
nehamk@umd.edu

Pulkit Kumar[*]
Department of Computer Science
University of Maryland
College Park, MD 20740
pulkit@umd.edu

Abstract

Many applications in healthcare and web datasets such as disease diagnosis, ad clicks, or product reviews, labeled data is available only for few positive examples. Majority of the examples, positive or negative, are left unlabeled. This class of problems is commonly known as One-class classification or Postive Unlabeled (PU) Learning. Given absence of labeled negatives in either training or validation data, a standard setup to simulate PU Learning is to take a labeled dataset with two classes and create an unlabeled class from all the negatives and randomly sampled positives. Traditional approaches to perform PU learning involve cost-sensitive classification while treating unlabeled class as negative class. In this work, we propose three different methods for PU Learning addressing some of the issues in the existing approaches.

1 Introduction

Various approaches have been proposed to solve the PU learning problem. Generally, the classification is done by adding pseudo negatives to the data and classifying the unlabeled data into either of the classes. Depending on how the positive labeled examples are sampled, various discriminative techniques have been proposed as mentioned in [1]. It has been found in [2] and [3] that we can achieve PU learning with cost-sensitive learning between positive and unlabeled data using a non-negative risk estimator. The existing techniques work moderately well under "selected completely at random" (SCAR) assumption of positive labels. While most techniques tackle unbiased PU learning problems, there has been little work on learning from biased data [4]. Some generative models like GenPU [5] have also been proposed to recover both positive and negative data distributions.

We have used GenPU as our baseline. We have trained and validated our models using MNIST dataset which has approximately 6000 samples of each digit in training and approximately 1000 samples in validation set. We have run 2 sets of experiments primarily, classification of 3 vs 5(Negative class is unimodal) case and classification of 3 vs Rest(Negative class is Multi-modal) of the digits. Negative class is marked as *Unlabeled* in this case. We have defined *Spill ratio* as percentage as Positive samples marked as *Unlabeled* and rest of the samples. Our selection of positive labeled data follows SCAR, an unbiased scenario, under spill ratios ranging from 0% (all 3's are labeled as positive) to 99% (64 3's are labeled as positive).

[*]Contributed equally on the project

Preprint. Under review.

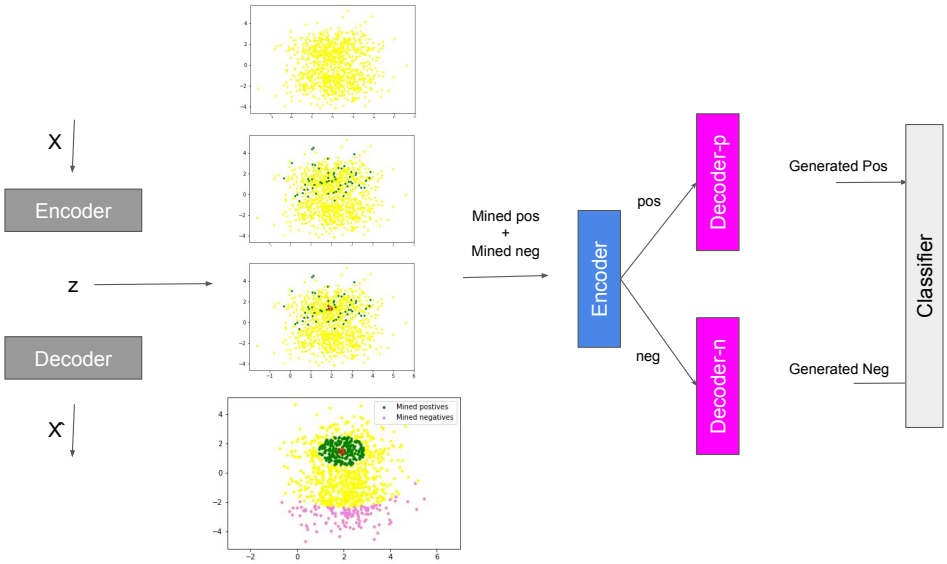

Figure 1: PU VAE Layout. Yellow represent all data points. Red point is the centroid of known positives. Green are mined postives and Violet are mined negatives. Once we have the mined positives and negatives, we train a non-traditional VAE, to generate more positives and negatives which we use to build a Positive-Negative classifier.

2 PU learning using VAE and psuedo-negatives

Gen-PU model was heavily dependent on hyper-parameters and is going into mode-collapse in one-vs-rest case, so we have tried a similar approach with Variational Auto-Encoders(VAEs). We wanted to build a more robust model which can work in any setting, no matter what the spill is and whether the unlabeled data-set has one or multiple classes.

2.1 Proposed Model

As you can see in Figure 1, for this model we propose to mine the negatives in 20 dimensional Gaussian space as encoded by conventional VAE [6]. Unlike other models which train only on Positives, we have trained the VAE with both Positive and Unlabeled data.

Since we have some known positives, we calculate the centroid and then calculate the distances of all the data-points from the centroid. We take the closest 1000 as positives and farthest 1000 points in the space as negatives. Once we have mined positives and mined negatives, we train a VAE with one Encoder and two Decoders (one for positive data and one for negative). The parameters of Encoder and Decoders are initilized to be the same as that of the first VAE which encoded all the Positive and Unlabeled data in the beginning. For training this VAE, we can backprop just through the decoders(scenario marked w/o ebp) or backprop all the way through encoder(scenario marked w. ebp in results table).

Once we generate the positives and Negatives with our PU-VAE, we can build a regular Positive-Negative(PN) classifier with the generated data.

The distance plot, as you see in the Figure 2, we use euclidean distance which represents negative log-likelihood of a point assuming data is normally distributed in latent space, and as you can see, the separation is quite good both in 3 vs 5 and 3 vs rest case even though we were working only with 64 positive samples. One issue with this approach is, this model assumes the centroid of the known positives have the highest likelihood, which might not be the case in reality. Also it does not account for the fact that there might be multiple modes in unlabeled data-set. A much bigger problem is that the negatives closer to the positives are not used for generation of positive data which in turn results in poor performance of the PN classifier.

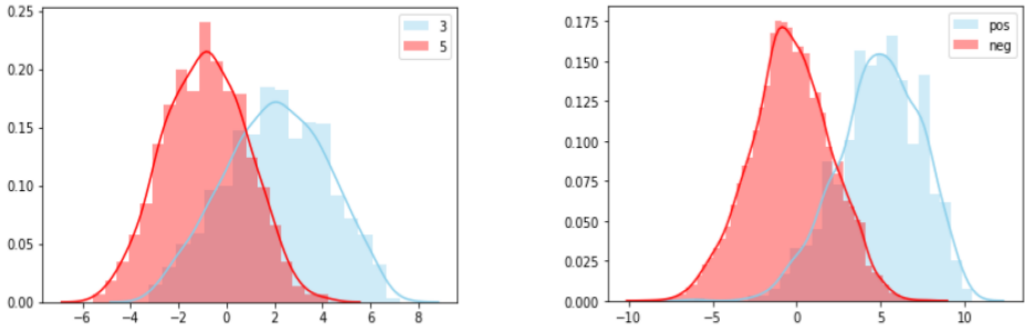

Figure 2: Euclidean distances of the points from the centroid. Left - 3 vs 5 with 99% spill, Right - 3 vs Rest with 99% spill

Table 1: PU-VAE: AuPR scores

3 vs 5			3 vs Rest		
Spill	AuPR w. ebp	AuPR w/o ebp	Spill	AuPR w. ebp	AuPR w/o ebp
0	0.88	0.87	0	0.74	0.76
0.99	0.86	0.86	0.99	0.76	0.77

2.2 Results

As you can see in Table 1, the AUPR values are very close to 0.88 in all 3 vs 5 scenarios and are close to 0.76 in 3 vs Rest scenarios, irrespective of Spill ratio. This seems to be a robust model but the AUPR values are not up to the state of the art.

One possible way to improve this is, train the PU-VAE and run the classification simultaneously instead of doing it in multiple steps right now (effectively removing the first VAE in this model).

3 PU learning using GLOW

3.1 Motivation

In the recent years, flow based generative models [7] have become popular due to their ability of having tractable log-likelihoods. Glow [8], proposed by Kingma et al., is a type of flow based generative model which uses 1×1 convolution and optimises on the plain log-likelihood.

Because of the tractability of the exact log-likelihood, we tried to model the distribution of the the positive class from the labelled positive samples. We hypothesised that the distribution of the negative class would be distinctively separate from the positive class.

3.2 Experiment setup

To test out the hypothesis, we considered class **0** of MNIST as the positive class and the class **1** as the negative (0 vs 1). Using the same hyperparamters as Kingma et al. [8], we trained GLOW models with different spill ratios of the positive class to simulate the Positive unlabeled setup. At inference time, the log-likelihood of each sample in the validation of both the classes was estimated by the model and a histogram was plotted. Figure 4 shows these histograms at different spill ratios. We repeated the same experiment with class **3** as the positive class and class **5** has the negative class (3 vs 5). Histograms for this case can e seen in figure 4.

In figure 3, it can be seen that histograms with two different peaks are formed for 0 vs 1 case and hence the two classes can be classified from the log-likelihood itself. But in figure 4, it can be observed that the histogram of 3 vs 5 overlap with each other, hence the two classes couldn't be classified accurately.

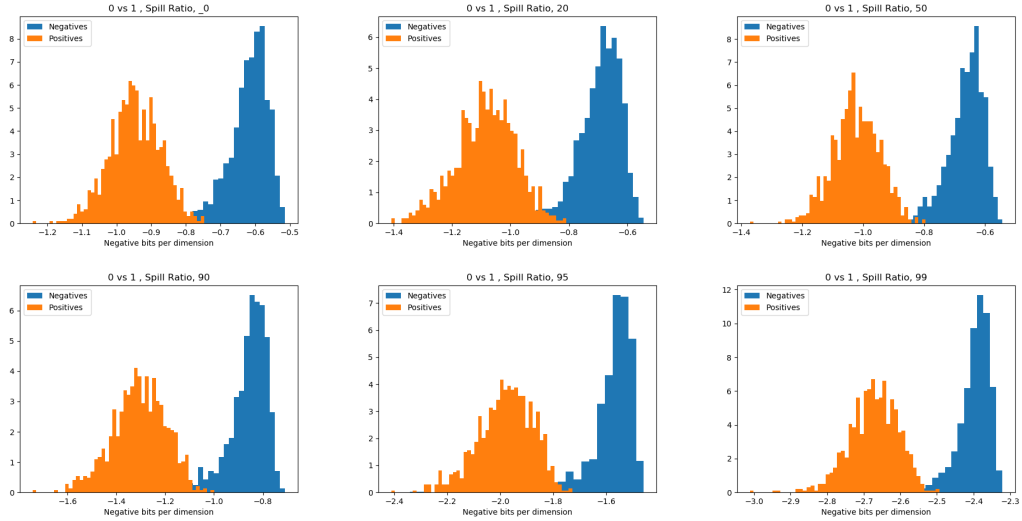

Figure 3: Histogram plots of log-likelihood using GLOW at different spill ratios with class 0 of MNIST as positive and class 1 as negative

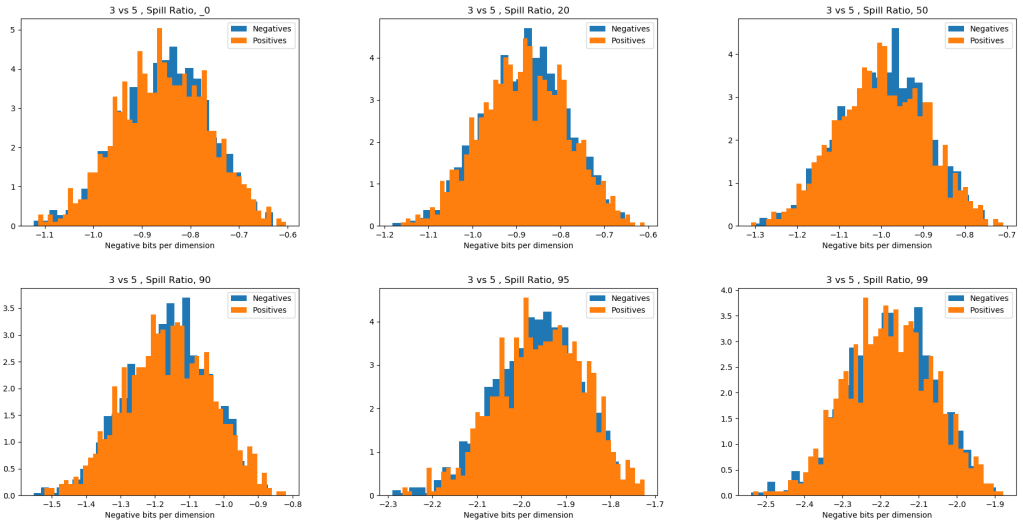

Figure 4: Histogram plots of log-likelihood using GLOW at different spill ratios with class 3 of MNIST as positive and class 5 as negative

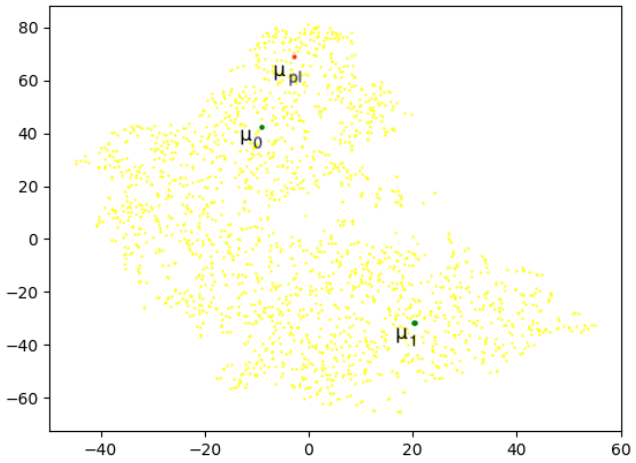

Figure 5: Clusters of positive and unlabeled data formed in latent space with their centroids

4 PU learning using Autoencoder with clustering

4.1 Motivation

Building upon the failure case of GLOW, we wanted to explore clustering methods for solving PU classification. By using the mean embedding of the labeled positive samples (μ_{pl}), we aimed to bring the centroid of one of the clusters (μ_0) closer to the μ_{pl} and the centroid of the other cluster (μ_1) farther from μ_{pl}.

4.2 Experiment Setup

To achieve this we first train an auto-encoder on the entire dataset so as to learn some representation of the data. Next, we discard the decoder of the autoencoder and from the encoder we extract the embedding for each of the data samples. On all the embeddings, K-Means clustering algorithm is used to cluster them into two clusters. The cluster ID (either 0 or 1) assigned to each sample, is used as the *pseudo label* (y_i) for that sample. Using the *pseudo labels*, corresponding centroid is calculated using,

$$\mu_k = \frac{1}{n_k} \sum_{y_i=k} \mathbf{z}_i \tag{1}$$

where k corresponds to cluster ID and is in $\{0, 1\}$, $\mathbf{z}_i$ corresponds to the embedding of the sample i, and n_k is the number of samples in cluster k. We also calculate μ_{pl} using,

$$\mu_{pl} = \frac{1}{n_{pl}} \sum \mathbf{z}_{pl} \tag{2}$$

where n_{pl} corresponds to the number of positive labelled samples and $\mathbf{z}_{pl}$ corresponds to embedding of a positive labelled sample.

Using these, the encoder is fine tuned according to the loss function,

$$loss = \min_{\mu_0, \mu_1} \|z_{pl} - \mu_{pl}\|^2 + \lambda_1 \|\mu_0 - \mu_{pl}\|^2 - \lambda_2 \|\mu_1 - \mu_{pl}\|^2 \tag{3}$$

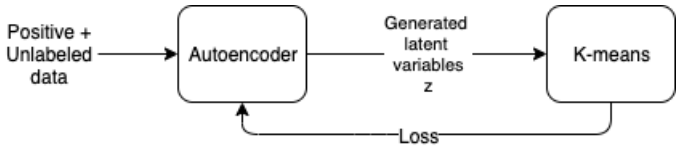

Figure 6: Autoencoder with K-means architecture

where λ_1 and λ_2 are hyperparamters used to balance the terms in the loss function. Gradient decent was used for optimising over the loss. Figure 6 shows a figurative representation of the training procedure. The variables and the steps are summarized in Algorithm 1.

Algorithm 1 Autoencoder with Clustering

procedure CAE($X_{train}, Y_{train}, b, l, e, \lambda1, \lambda2, f_{\theta1}, g_{\theta2}, X_{test}, Y_{test}$)
  X_{train}, Y_{train}: Training data and corresponding Postive or Unlabelled Labels
  X_{test}, Y_{test}: Test data and corresponding true labels
  $\lambda1, \lambda2$: hyperparameters to tune the loss function
  b: batch size
  l: Learning rate
  e: Number of epochs
  $f_{\theta1}$: Encoder function
  $g_{\theta2}$: Decoder function
  function AUTOENCODER($X_{train}, f_{\theta1}, g_{\theta2}$)
    return $Z_{train} = f_{\theta1}(X_{train})$
  function ENCODER_TUNING(X_{train}, Y_{train})
    Train the Autoencoder on all the data and learn representations, Z_{train}
    Discard the Decoder
    while $k <= e$ **do**
      Calculate the centroid of known positives of Z_{train} as mentioned in equation 2.
      Perform K-Means clustering algorithm on all of Z_{train} as mentioned in equation 1.
      Assign $Y_{label} = [0, 1]$ depending on the distance from μ_{pl}
      Calculate the loss as follows using equation 3.
      Backprop the loss through the encoder.

$$\theta_1^{k+1} = \theta_1^k - l * \nabla_{\theta1} loss \tag{4}$$

$$Z_{train} = f_{\theta_1^{k+1}}(X_{train}) \tag{5}$$

    return θ_1
  Validation
  Using the θ_1, learn the representations of the X_{test} as Z_{test}
  Run 2-means clustering on Z_{test} and calculate the respective centroids.
  Classify the set of points whose centroid is closest to μ_{pl} as positives and the other set as negatives

5 Results

Table 2 summarizes all our models in comparison with the baseline GenPU. GenPU proposes an elegant approach of using an array of discriminators for positive, unlabeled and negative data and two generators, for positive and negative data. When we ran our experiments on GenPU, we noticed that the model is highly dependant on its hyperparameters. It shows good results in the 99% spill scenario, but it fails to perform for lower percentage spills and in the 3 vs Rest case. Our PU-VAE model is consistent across different spills and different input data. The Autoencoder with K-means approach shows promising results in all experiments.

Table 2: Summary of accuracy and AUPRC scores of three approaches compared with the baseline, GenPU

	99% spill ratio		**0% spill ratio**	
Model	3 vs 5	3 vs rest	3 vs 5	3 vs rest
GenPU (baseline)	0.99	0.56	0.77	0.16
PU-VAE	0.87	0.77	0.87	0.76
Glow	0.53	0.10	0.53	-
Autoencoder with K-means	0.88	-	0.93	-

6 Discussion

The PU classification model using Autoencoder with Clustering or the PU VAE model are very easy to train and AuPRC scores are reasonable. We believe by modifying these models a bit, we will be able to beat the SOTA generative model, Gen-PU. Gen PU model is highly sensitive to hyperparameters, and suffers from issues of mode collapse and mode oscillation, especially when data is sampled from data distribution with large number of modes (3 vs Rest case).

One of the other ways to improve these models is to use Normalized Wasserstein distance measure [9]. Another option is to use Soft K-means in the Autoencoder with Clustering model rather than the Hard clustering.

All the approaches we have tried so far are under the assumption that data is sampled completely at random (SCAR), but in real world, most of the PU data is biased, hence we need to extend our models to biased data.

References

[1] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *arXiv preprint arXiv:1811.04820*, 2018.

[2] Marthinus C du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 703–711. Curran Associates, Inc., 2014.

[3] Ryuichi Kiryo, Gang Niu, Marthinus Christoffel du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. *CoRR*, abs/1703.00593, 2017.

[4] Masahiro Kato, Takeshi Teshima, and Junya Honda. Learning from positive and unlabeled data with a selection bias. In *International Conference on Learning Representations*, 2019.

[5] Ming Hou, Qibin Zhao, Chao Li, and Brahim Chaib-draa. A generative adversarial framework for positive-unlabeled classification. *CoRR*, abs/1711.08054, 2017.

[6] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. *CoRR*, abs/1605.08803, 2016.

[8] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018.

[9] Yogesh Balaji, Rama Chellappa, and Soheil Feizi. Normalized wasserstein distance for mixture distributions with applications in adversarial learning and domain adaptation. *CoRR*, abs/1902.00415, 2019.