Learning from Label Proportions: Bootstrapping Supervised Learners via Belief Propagation

Shreyas Havaldar *
Google Research India
shreyasjh@google.com

Navodita Sharma *
Google Research India
navoditasharma@google.com

Shubhi Sareen
Google India
shubhisareen@google.com

Karthikeyan Shanmugam
Google Research India
karthikeyanvs@google.com

Aravindan Raghuveer
Google Research India
araghuveer@google.com

Abstract

Learning from Label Proportions (LLP) is a learning problem where only aggregate level labels are available for groups of instances, called bags, during training, and the aim is to get the best performance at the instance-level on the test data. This setting arises in domains like advertising and medicine due to regulatory guidelines concerning privacy. We propose a novel algorithmic framework for this problem that iteratively performs two main steps. For the first step (Pseudo Labeling) in every iteration, we define a Gibbs distribution over binary instance labels that incorporates a) covariate information through the constraint that instances with similar covariates should have similar labels and b) the bag level aggregated label. We then use Belief Propagation (BP) to marginalize the Gibbs distribution to obtain pseudo labels. In the second step (Embedding Refinement), we use the pseudo labels to provide supervision for a learner that yields a better embedding. Further, we iterate on the two steps again by using the second step's embeddings as new covariates for the next iteration. In the final iteration, a classifier is trained using the pseudo labels. Our algorithm displays strong gains against several SOTA baselines (up to 15% AUROC) for the LLP Binary Classification problem on various dataset types - tabular and image. We achieve these improvements with minimal computational overhead above standard supervised learning due to Belief Propagation, for large bag sizes, even for a million samples.

1 Introduction

Learning from Label Proportions (henceforth LLP) has seen renewed interest in recent times due to the rising concerns of privacy and leakage of sensitive information [3, 5, 39, 13, 37]. In the LLP binary classification setting, all the training instances are aggregated into *bags* and only the aggregated label count for a bag is available, i.e. proportion of 1's in a bag. Features of all instances are available. This can be seen as a form of weak supervision compared to providing instance-level labels. The main goal is learn an instance wise predictor that performs very well on the test distribution.

There are two sources of information that can help in predicting the instance wise label on the training set. One is the bag level label proportions that are provided. The other source of information is indirect and comes from the fact that any smooth true classifier would likely assign similar labels to instances with similar covariates or feature vectors. Covariate information of instances belonging

*Equal Contribution

Workshop on Regulatable Machine Learning at the 37th Conference on Neural Information Processing Systems (RegML @ NeurIPS 2023).

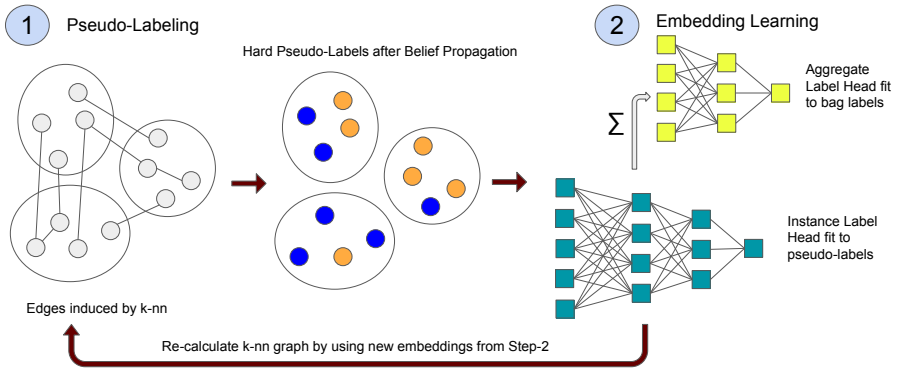

Figure 1: 12 instances placed equally into 3 bags. Step-1: On the k-nn graph induced by the covariate embeddings we perform belief propagation to obtain pseudo-labels that respect edge constraints and bag constraints. Then in Step-2 we fit a MLP to the instance pseudo-labels and bag aggregate label. Embedding learned in an intermediate layer is used to further refine the k-nn graph in Step-1. (Figures best viewed in colour)

to the bags are explicitly given. Some of the current methods [37, 3] propose to fit the average soft scores over a bag, predicted by a deep neural network (DNN) classifier, to the given bag label proportion. There is another class of approaches that seek to train a classifier on instance level loss obtained using some form of *pseudo labeling* [18, 39].

Our work builds on the idea of forming pseudo labels per instance. Our key observation is that one could utilize *two* types of information: *bag level constraints* and *covariate similarity information* in an explicit way. To realize this, we take inspiration from coding theory for communication systems [19, 16], where one of the fundamental problems is to decode an unknown message string sent by the encoder using only parity checks over groups of bits from the message. Parity checks provide *redundancy* that battles against noise/corruptions in the channel. The state of the art codes [19] are decoded using Belief Propagation [24] on a factor graph where message bits are variable nodes (whose "label" is to be learnt) and parity checks form factor nodes (that constrain the sum of bits in the parity checks to be odd or even). Sum-product belief propagation is used to learn the marginal soft score on the each of the message bits.

Motivated by the above connection, we propose a novel iterative algorithm that has two stages: We use an existing embedding to learn pseudo-labels and then use the learnt pseudo labels to refine embeddings and we iterate this procedure with new embeddings. Our central approach is described in Figure 1. We outline our contributions in detail below:

1) We draw a parallel between the LLP problem and the parity recovery problem for *pseudo labeling*. The labels of instances are bits of the message and parity checks are the bag level counts (more general than just parity). We adapt this analogy naturally to form a *Gibbs distribution* that enforces the bag constraints. To obtain further redundancy, we exploit covariate information, where, for every pair of instances that are in the *K-nearest neighborhood* of each other, we force their binary *labels* to be *similar*, i.e. another "*parity constraint*" is added to the Gibbs distribution. As can be seen in a cartoon depiction in step 1 of Fig. 1, the nearest neighbors induce similar labels (colors). Then, we do a *sum product* Belief Propagation (*BP*) to obtain marginal pseudo labels. To the best of our knowledge, ours is the first work at making this connection between the information theoretic approach of recovering messages from parity and the LLP problem.

2) Our novelty in the second stage is to utilize thresholded soft pseudo labels to provide full supervision to learn a new embedding. However, we observe that after marginalization thresholded soft labels may violate bag constraints. Therefore, we use a novel Deep Neural Network architecture that has 1) *instance head* giving rise to an instance level loss involving the thresholded soft pseudo label and 2) *bag level head* formed by pooling the penultimate layers's representations of instances within a bag giving rise to a loss between bag proportions and predicted bag level proportion. We call the final loss as *Aggregate Embedding loss*. This loss is used to train the penultimate layer embedding.

3) We iterate the above two steps by using the embedding obtained in the previous step as features. We show that our iterative two stage algorithm, finally produces instance level predictions that outperform a number of LLP baselines including DLLP [3] by wide margins. Improvements obtained are upto **15%** on standard **UCI** classification datasets. We outperform the baselines by upto **0.8**% on a large,

challenging **Criteo** dataset and upto **7%** on **image** datasets. Our methods mostly outperform most baselines in the large bag regime (upto **15%** gains for bag size ≥ 512) where supervision is very weak. It is worth noting that on bag size of 2048 there are only a maximum of ~20 bags on the datasets and yet our method does not display significant degradation in performance. Our ablations show that *remarkably* 1-NN achieves most of the gains relative to using k-NNs in the pseudo labeling step (Section A.1.1). We find that reducing percentage of KNN constraints (to 50%) registers a significant drop in test metrics underscoring the importance of 'parity checks' from covariate nearness (Section 6.2) We provide many other ablations that validate different components of our approach (Section 6 and Section A.1).

2 Related Work

Learning from Label Proportion (LLP): Quadrianto et al. [28] use kernel methods under the assumption of class conditioned independence of bags. Yu et al. [36] were one of the first to tackle the problem and present theoretically backed $\propto$SVM, a non-convex integer programming solution. This is computationally infeasible on large sized datasets we use. Patrini et al. [23] introduced a fast learning algorithm that estimates the mean operator using a manifold regularizer while providing guarantees on the approximation bounds. Scott and Zhang [32] provide an approach that use Mutual Contamination Models which provides some form of weak supervision. The method does not scale to large number of training instances. Several recent methods have been introduced to learn from bagged data. Poyiadzi et al. [26] propose an algorithm that uses label propagation, i.e. iterative damped averaging of neighbors labels where neighbors are decided based on some similarity measure. Every node is set to the proportion of ones from the bags they participate in at the beginning. This is closest in spirit to ours. However, we always impose bag level constraints and covariate information through a joint Gibbs distribution and use BP to marginalize it followed by an embedding refining step. Poyiadzi et al. [26] provide results on Size-3 Bags and Size-100 Datasets and their algorithm requires extensive compute for inversion of a kernel matrix. Ardehaly and Culotta [3] use deep neural networks to tackle the LLP problem show very good empirical performance. When bag size is large their method degrades significantly. Tsai and Lin [34] use covariate information in the form of a consistency regularizer that modifies the decision boundary. The work does not perform well at larger bag sizes and is computationally expensive for large datasets. Zhang et al. [39] use forward correction loss to draw parallels to learning from label noise. The method does not converge well for smaller bags. Busa-Fekete et al. [5] is a very recent work approaching the problem by providing a surrogate loss. We further elaborate on our baselines in section A.4.

Belief propagation in decoding error correcting codes has a long history. There is a related problem of learning from pooled data which has a group testing flavor. We review these in the supplement.

3 Problem Setting and Overview of Our Solution

Consider a supervised learning dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)_{i=1}^m\}$ where $x_i \in \mathbb{R}^d$, $y_i \in \{0, 1\}$. The instance wise labels y_i's are not explicitly revealed to the learning agent. There are a set of "bags" $\mathcal{B} = \{S_1 \dots S_n\}$ that contains subsets $S_i \subseteq [m]$ and for each S_i, bag level counts $y(S_i) = \sum_{j \in S_i} y_j$. In this work, we will consider the case of disjoint bags, i.e. $S_i \cap S_j = \emptyset$. Let the vector of bag level counts be $[y(S_1) \dots y(S_n)] = y(\mathcal{B})$. In this work, we consider the following problem:

Given the covariates ($\{x_i\}_{i=1}^m$), information about bag compositions ($\mathcal{B}$) and bag level counts of labels ($y(\mathcal{B})$), our aim is to learn a classifier $f : \mathbb{R}^d \to \{0, 1\}$, $f \in \mathcal{F}$ such that the loss $\ell(\cdot)$ on the test distribution $\mathbb{E}_{(x,y)\sim\mathbb{P}}[\ell(f(x), y)]$ is minimized. Here, $\mathcal{F}$ is the set of classifiers we would like to optimize over. We term this as *learning from label proportions* problem.

Our main contribution to this above problem is an iterative procedure that repeats two steps: a) Pseudo Labeling obtained through Sum-Product Belief Propagation that uses a Gibbs distribution capturing covariate information and bag level constraints and b) Learning a better embedding that uses training signals from an instance level predictor on top of the embedding that fits pseudo labels while using the same embedding over multiple instances to simultaneously predict bag level proportions. Then, in the next iteration we use the embedding learnt in the second step instead of original covariates and perform the two steps again. In the last iteration, we simply obtain an instance level predictor to score on the test dataset. Our complete approach is given in Algorithm 1.

Algorithm 1 Iterative Embedding Refinement with BP and Aggregate Embedding

Input: Covariates $\mathcal{D} = \{x_i\}$, Bag Information $\mathcal{B} = \{S_1, S_2 \dots S_n\}$, Bag Label Counts $\{Y(S)\}_{S\in\mathcal{B}}$. Parameters $k, \lambda_s, \lambda_b, T, L, L', \tau, \delta_d, k(\cdot,\cdot), d(\cdot,\cdot)$
Output: Soft Classifier $f_L(\cdot)$
Set Covariates $\{z_i\} \leftarrow \{x_i\}$
for $r \leftarrow 0$ **to** R **do**
  (`Pseudo Labeling Step`)
  Initialize node and Pairwise potentials $h_i, J_{i,j}$ for Belief Propagation as in (3) using $\{z_i\}$.
  $\{\mathbb{P}^r(y_i)\} \leftarrow$`SUM-PRODUCT-BP`$(\{h_i\}, \{J_{ij}\})$.
  Obtain Hard Labels: $y_i^r \leftarrow \mathbf{1}_{\mathbb{P}^r(y_i)>\tau}$.

  (`Train embedding with instance and bag losses`)
  Train the DNNs $f_L(\cdot), g_L(\cdot)$(instance and bag loss heads) using hard labels $\{y_i^r\}$ and bag level labels using $\mathcal{L}_{\text{Agg}-\text{Emb}}$ loss as in (9).

  Set Covariates $\{z_i\} \leftarrow \{f_{L'}(x_i)\}$ (`Update Embedding`)
end for
Output: Return the function $f_L(\cdot)$. (`Obtain instance label predictor`)

4 Details of our Algorithm

4.1 Step 1: Obtaining Pseudo-Labels through Belief Propagation (BP)

Now, we describe the first step that involves obtaining soft labels from the bag constraints and covariate information. To this end, we form a Gibbs distribution whose energy function captures two-fold information: a) bag level constraints and b) label similarity when two points are nearby.

Gibbs Distribution from Bag level constraints and covariate information: The bag level constraints are penalized using a least squares loss between sum of all labels in the bag and the count given by $(\sum_{j\in S_i} y_j - y(S_i))^2$. When two points are close in some distance measure, we would like to make sure their labels are close. In order to capture this, for every point x_i we form k-nearest neighbor set $N_k(x_i)$ with respect to a given distance function $d(x, x')$, with x_j added to $N_k(x_i)$ if $d(x_j, x_i) \leq \delta_d$. If $x_j \in N_k(x_i)$, we use the least squares penalization $(y_i - y_j)^2$. We also use a kernel function $k(x, x')$ that scales the least square penalization due to nearness. In most of experiments, we fix $d(x, x')$ to be the cosine distance or euclidean distance. Choices of $k(\cdot)$ are Matern Kernel and RBF kernels.

We define the Gibbs distribution below for the entire dataset $\mathcal{D}$ given by:

$$\mathbb{P}_{\lambda_b,\lambda_s}(y_1..y_m) \propto \exp\left(-\lambda_b \sum_{S_i\in\mathcal{B}} (\sum_{j\in S_i} y_j - y(S_i))^2 - \lambda_s \sum_{x_i,x_j\in\mathcal{D}, x_j\in N_k(x_i)} k(x_i,x_j)(y_i - y_j)^2\right) \tag{1}$$

This is an Ising model with pairwise potentials and node potentials (external field). We note that both the terms are invariant upto a shift in y_i by a constant. Therefore, transformation to $\{+1, -1\}$ through $y_i' = 2y_i - 1$ would only scale λ_b, λ_s by a factor of 4. Therefore, the energy function in terms of $\{+1, -1\}$ upto a universal scaling is identical to that of $\{0, 1\}$. Hence, we will remain in $\{0, 1\}$ and state the pairwise and node potentials.

Observing that $y_i^2 = y_i$ and the fact that constant terms in the energy function would not affect the distribution (due to normalization), we can rewrite (1) as:

$$\mathbb{P}_{\lambda_b,\lambda_s}(y_1 \dots y_m) \propto \exp\left(\sum_{i\in[m]} y_i \left[\sum_{S\in\mathcal{B}: i\in S} \lambda_b(2y(S) - 1) - \lambda_s \sum_{x\in N_k(x_i)} k(x_i, x)\right] + \right.$$
$$\left. \sum_{i\neq j} y_i y_j \left[2\lambda_s k(x_i, x_j)(\mathbf{1}_{x_j\in N_k(x_i)} + \mathbf{1}_{x_i\in N_k(x_j)}) - 2\lambda_b |S \in \mathcal{B} : (i, j) \in S|\right]\right) \tag{2}$$

Remark: We note that not all pairwise terms are present. If two points are not in K-NN neighborhood of each other and if they don't belong together in any bag, then there would be no pairwise term corresponding to it. In our experiments, we consider the case where Bags are disjoint and non overlapping. Therefore, every instance i participates in only one bag. Use of K-NN and small disjoint bags creates only linear number of terms in the Gibbs distribution.

Pairwise and Node potentials: This is an Ising model $\mathbb{P}(\mathbf{y}) \propto \exp(\sum y_i h_i + \sum_{i \neq j} y_i y_j J_{i,j})$ with node potentials and pairwise potentials given by:

$$h_i = \sum_{S \in \mathcal{B}: i \in S} \lambda_b (2y(S) - 1) - \lambda_s \sum_{x \in N_k(x_i)} k(x_i, x) \tag{3}$$

$$J_{i,j} = 2\lambda_s k(x_i, x_j)(\mathbf{1}_{x_j \in N_k(x_i)} + \mathbf{1}_{x_i \in N_k(x_j)}) - 2\lambda_b |S \in \mathcal{B} : (i,j) \in S| \tag{4}$$

Obtaining Pseudo Labels using sum-product Belief Propagation(BP): We use the classical sum-product Belief Propagation [19] on (2) to approximate the marginal distribution $\mathbb{P}_{\lambda_s, \lambda_b}(y_i)$. We briefly describe the algorithm, At every round t, node i passes the following message $m^t_{j \to i}(y_i),\ y_i \in \{0, 1\}$ to every node $j : J_{i,j} \neq 0$ given by:

$$m^t_{j \to i}(y_i) = \sum_{y_j \in \{0,1\}} \exp(y_i h_i) \exp(J_{i,j} y_i y_j) \prod_{k \neq i : J_{k,j} \neq 0} m^{t-1}_{k \to j}(y_j) \tag{5}$$

Here, $m^{t-1}(\cdot)$ represents the message passed in the previous iteration. After T rounds of message passing, we marginalize by using the following (and further normalizing it):

$$\mathbb{P}(y_i) \propto \exp(y_i h_i) \prod_{j : J_{i,j} \neq 0} m_{j \to i}(y_i) \tag{6}$$

Implementation: We denote `SUM-PRODUCT-BP`$(\{J_{i,j}\}, \{h_i\})$ to be the sum product belief propagation function that implements T rounds of (5) and (6). We use `PGMax` package [40] implemented in `JAX` [4] where we just need to specify the potentials $J_{i,j}$ and h_i.

4.2 Step 2: Embedding Refinement Leveraging Pseudo Labels

We observe that (1) uses covariate information by exploiting nearness using nearest neighbors induced by a distance function $d(x, x')$ and using a kernel $k(x, x')$. We now provide an iterative method to refine representation x_i such that points with same true labels are brought together progressively although only bag level labels are available. We start with the original features $\{x_i\}$ given to the algorithm (this could already be an embedding obtained from another self-supervised module or any other unsupervised training method like auto encoding).

Learn marginal Pseudo Labels: We first identify pseudo labels $\mathbb{P}_{\lambda_s, \lambda_b}(y_i)$ by applying `SUM-PRODUCT-BP` on h_i, J_{ij} obtained from $\{x_i\}$'s. We expect the pseudo labels not to be perfect since it operates on only bag level information and covariate similarity information.

Learning Embedding using Pseudo Labels: Let us consider a deep neural net (DNN) classifier (with L layers) of the form given below:

$$f_L(x) = \text{softmax}(W_L^T \sigma_{L-1}(W_{L-1} \sigma_{L-2}(\cdots \sigma_1(W_1^T x + b_1)) + b_{L-1}) + b_L) \tag{7}$$

where σ_ℓ represents a coordinate wise non-linearity (like `Relu` function), $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}},\ \ell \in [1 : L]$ represents a weight matrix at the ℓ-th layer that multiplies the representation from the $\ell - 1$ layer of dimension $d_{\ell-1}$ and b_ℓ represents the biases added coordinate wise to the output which is d_ℓ dimensional. In our work, we focus on binary classification where $d_L = 1$ and $d_0 = d$ (input feature dimension). We call $f_L(\cdot)$ the *instance loss* head.

One option is to just fit $f_L(x_i)$ to the information from Pseudo labels $\mathbb{P}_{\lambda_s, \lambda_b}(y_i)$ at the instance level. However, it may not be consistent with the bag level labels in the expected sense. So we impose bag level constraints by first average pooling third to last layer output $f_{L-2}(x)$ across instances $x \in S$ where S is a bag of instances. Then, we have one more hidden layer on top of this pooled representation to finally produce a soft score for the bag proportion. We call this the *bag loss* head and it produces the following soft score for a bag of instances $S \subseteq \mathcal{B}$:

$$g_L(S) = \sigma_L \left(V_L^T \left(\sigma_{L-1} \left(V_{L-1}^T \text{Avg Pooling}[\{f_{L-2}(x_i)\}_{i \in S}] + \tilde{b}_{L-1} \right) + \tilde{b}_L \right) \right) \tag{8}$$

We define two loss functions that learns $f_{L-2}(z)$ representation to simultaneously be consistent with 1) the bag label proportion through the average pooling operation in (8) and 2) the other one that makes it consistent with hard labels obtained by *thresholding* soft pseudo labels given by $y_i^0 = \mathbf{1}_{\mathbb{P}_{\lambda_s,\lambda_b}(y_i)>\tau}$, where $\mathbf{1}_E$ is the indicator function when event E holds.

The composite loss function is given by:

$$\mathcal{L}_{\text{Agg-Emb}}(S) = \sum_{i \in S} \text{CE}\left(f_L(x_i), \{y_i^0\}\right) + \lambda_a \text{CE}\left(g_L(S), \frac{y(S)}{|S|}\right) \tag{9}$$

Here, CE is the cross entropy loss. We call this composite loss function $\mathcal{L}_{\text{Agg-Emb}}$ the *aggregate embedding loss* function.

4.3 Iterative Refinement

We take the representation computed at some layer $L' < L$ (denoted by $f_{L'}(x)$) and apply the belief propagation (`SUM-PRODUCT-BP`) again where covariates are given by $\{z_i = f_{L'}(x_i)\}$. We use $L' = L - 2$. Then, let the new pseudo labels obtained be $\mathbb{P}^1_{\lambda_s,\lambda_b}(y_i)$ using $\{z_i\}$ as covariates. We obtain hard labels from thresholding pseudo labels by τ to obtain $y_i^1 = \mathbf{1}_{\mathbb{P}^1_{\lambda_s,\lambda_b}(y_i)>\tau}$ We again fit similar DNNs (f_L, g_L) by using $\mathcal{L}_{\text{Agg-Emb}}$ to the new hard labels $\{y_0^1\}$ and the bag level labels $\{y(S)\}$. In principle, we could iterate it several times to refine embeddings progressively but we stop when the new iteration does not clearly improve performance on validation set. Refer to the section A.2.2 for analysis on convergence of our method in 1-2 iterations. When we test on instances, we always remove the bag level head $g_L(\cdot)$ and test it with just the soft score $f_L(x)$. We describe the iterative procedure in Algorithm 1.

5 Experiments

We perform extensive experimentation on four datasets. We follow the standard procedure of creating disjoint *random* bags where we sample instances without replacement from the training set, and keep repeating this for each bag, *bag-size: k* number of times.

1. Adult Income [7] [14]: Classification task is to predict whether a person makes over $50K a year based on the provided census data of 14 features. The dataset is split 90-10 as train-test and 10% of train is used as a hold out validation following Yoon et al. [35]

2. Bank Marketing [7] [21]: The task here it to predict if the client will subscribe a term deposit from 16 features. Data is split as $\frac{2}{3}$-$\frac{1}{3}$ train-test split. We further use $\frac{1}{3}$ of the training set as a hold-out validation set.

3. Criteo [15]: 1 week of ad click data to predict CTR with 39 features. We sample non-overlapping sets of 1 million, 200k and 250k instances to form train, validation and test datasets. Note that Criteo is a very challenging benchmark with only +2% AUROC improvement shown in the last 7 years [1]

4. CIFAR-10 [11] 60K images with 10 classes. **CIFAR-B**: We assign label *1* to all *Machine* classes (0,1,8,9) and label *0* to all *Animal* classes (2,3,4,5,6,7). In this dataset, 40% of all instances belong to the positive class. **CIFAR-S**: All instances belonging to the class *Ship* are assigned label *1* and all other instances are assigned label *0*. This dataset has 10% positive instances.
We use the standard Train-Test splits for these 2 datasets. We use 10% of the data from the Train Set as Validation Set to tune our hyperparameters.

We compare our method against several top LLP Baselines; namely DLLP [3], EasyLLP [5], GenBags [30], LLP-FC [39] and LLP-VAT [34] described in detail in appendix section A.4

5.1 Experimental Setup

We optimize our algorithm, using hyperparameters $\lambda_s, \lambda_b \in [10^{-4}, 200]$, $k \in [1, 30]$, $T = [50, 100, 200]$, $\tau \in (0, 1)$, $MLP_{LR} \in [10^{-6}, 1]$, $MLP_{WD} \in [10^{-12}, 10^{-1}]$, $\lambda_a \in [0, 10]$, $\delta_d \in [10^{-4}, 1]$, $BatchSize_{train} = [2, 4, 8, \ldots 4096, 8192]$, tuned using Vizier [33] to achieve the best Validation AUROC score. Illustrative values of best hyperparameters for various experiments are given in appendix section A.6. We then report the corresponding Test AUROC % averaged over 3

Table 1: Performance (Test AUROC) on UCI Tabular Datasets on Bag Sizes 8, 32, 128, 512, 1024, 2048 against major baselines. Instance-MLP performance on Adult is **90.30** (0.08) and on Marketing is **86.62** (0.06). **Ours-Itr-*n*** refers to our method run for *n* iterations.

Bag Size:	8	32	128	512	1024	2048
Dataset:	Adult					
DLLP	89.19 (0.32)	87.52 (0.46)	85.87 (0.91)	82.95 (1.37)	63.48 (2.41)	62.58 (2.18)
EasyLLP	88.68 (0.48)	87.51 (0.76)	75.59 (0.84)	66.02 (1.72)	61.65 (1.01)	63.21 (3.53)
GenBags	89.22 (0.32)	87.72 (0.34)	86.43 (0.28)	84.00 (0.26)	83.52 (0.91)	80.07 (0.90)
Ours-Itr-1	89.32 (0.26)	87.75 (0.32)	86.70 (0.31)	**84.97 (0.43)**	83.61 (0.49)	84.69 (0.76)
Ours-Itr-2	**89.47 (0.29)**	**87.82 (0.33)**	**86.87 (0.39)**	84.01 (0.34)	**83.88 (0.55)**	**84.95 (0.69)**
Dataset:	Marketing					
DLLP	84.49 (0.70)	82.65 (0.94)	79.69 (2.03)	70.36 (0.64)	66.39 (2.43)	65.60 (3.21)
EasyLLP	83.63 (0.34)	82.87 (0.72)	75.05 (3.29)	68.97 (2.76)	50.23 (1.21)	50.12 (0.55)
GenBags	85.26 (0.42)	83.15 (0.34)	79.74 (0.50)	69.29 (0.92)	64.82 (3.10)	58.43 (4.31)
Ours-Itr-1	85.76 (0.26)	84.18 (0.33)	**82.71 (0.44)**	77.71 (0.46)	80.56 (0.55)	78.63 (0.83)
Ours-Itr-2	**86.26 (0.31)**	**84.23 (0.45)**	82.46 (0.35)	**81.68 (0.77)**	**81.66 (0.61)**	**81.01 (0.92)**

trials and report the sample standard deviation in parenthesis throughout our tables. We perform the same setup for all our baselines. Best number is reported in **bold** and 2nd best is reported in underline

We also run the same MLP on true instance labels. This provides an upper bound to the performance that we can reach using aggregated labels. We report this number for each dataset at the table heading. We use an MLP with 5 Hidden Layers with relu activation and the following number of hidden units: $[5040, 1280, 320, 128, 64]$ for our 2nd Step. The final layer has sigmoid activation. We use Adam optimizer and Binary Cross Entropy Loss for all our datasets. We use the same MLP for all relevant baselines as well. We report main results using $k(\cdot, \cdot)$ = Matern and $d(\cdot, \cdot)$ = Cosine.

We perform experimentation on 6 bag sizes: 8, 32, 128, 512, 1024, 2048. All experiments were performed on a single NVIDIA V100 GPU. We provide further implementation details and the experimental details for the baselines in the supplementary section A.5

5.2 Performance Analysis

We compare the performance of our method with several baselines on all the datasets. **Ours-Itr-*n*** refers to our method run for *n* iterations.

We use the original features as is for the 2 UCI Datsets with 14 features for Adult Dataset, and 16 features for Marketing Dataset respectively.

Table 1 shows the performance of our method on the two UCI datasets across 6 bag sizes. We make four observations. First, for lower bag size (8, 32) our method almost bridges the gap with the instance level performance. Second, the second iteration of our method almost always improves performance over the first iteration across both datasets. Third, we are able to consistently outperform all the baselines across all bag sizes in both datasets. Finally, in large bag regime, our methods perform even better. For instance, with bag sizes 1024 and 2048, we are close to **15%** better compared to the nearest baseline DLLP.

In Table 2 we compare our method with other baselines on the Criteo dataset.[2] We use the self-supervised method MET [20] to generate embeddings for Criteo dataset to obtain better initial embeddings.This is because most of the features in Criteo are categorical and some of them have large number of categories rendering naive one-hot encoding very intractable. For fairness and consistency we use MET embeddings as input for all the baselines we compare against as well.

Our method scales really well for 1 million samples with negligible computational cost for the sum-product BP step. Criteo is an inherently harder dataset to work with due to high feature dimensionality and most of them being categorical. Over the last 7 years, the dataset has seen a 2% improvement in AUROC while our method is able to produce a **0.8%** improvement over DLLP. Similar to the previous tabular datasets, we also observe here that the iteration seem to help improve performance.

[2]We were not able to run BP on Criteo for large bag sizes since we ran into integer-overflow issues. It will take some time and perhaps even involved changes to the underlying PGMax library code to resolve them to accommodate large number of factors

Table 2: Performance (Test AUROC scores) on Criteo-1M on Bag Sizes 8, 32, 128 against major baselines. Instance-MLP performance on Criteo is **75.86** **(0.04)**

Bag Size:	8	32	128
Dataset:		Criteo	
DLLP	74.11 (0.09)	72.86 (0.01)	**70.99 (0.01)**
EasyLLP	70.77 (0.92)	68.42 (0.62)	62.87 (1.50)
GenBags	73.34 (0.01)	71.32 (0.77)	70.39 (0.46)
Ours-Itr-1	74.96 (0.23)	73.36 (0.33)	70.45 (0.51)
Ours-Itr-2	**74.97 (0.24)**	**73.43 (0.61)**	70.81 (0.44)

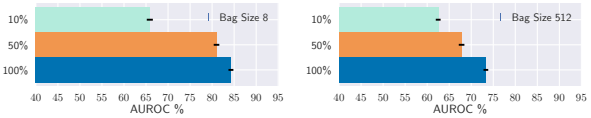

Figure 2: Comparison of the performance on adding different percentage of neighbours on Marketing on bag-size 8 and 512

Table 3: Performance (Test AUROC scores) on Image Datasets on Bag Sizes 8, 32, 128, 512, 1024, 2048 against major baselines. * denotes *SVD did not converge error* (while obtaining noisy labels), ! denotes *Out of Memory Error*. &: Validation performance of first iteration of our algorithm was clearly superior. Instance-MLP AUROC on CIFAR-B is **96.58** **(0.04)** and on CIFAR-S is **95.06** **(0.07)**

Bag Size:	8	32	128	512	1024	2048
Dataset:			CIFAR-B			
DLLP	95.49 (0.18)	**94.05 (0.09)**	**90.90 (0.06)**	**86.18 (0.67)**	76.65 (2.62)	68.19 (1.75)
GenBags	95.32 (0.05)	93.40 (0.17)	88.97 (0.33)	84.30 (0.60)	**82.73 (1.16)**	**75.22 (0.75)**
EasyLLP	91.33 (0.43)	84.67 (3.35)	81.37 (2.28)	58.07 (13.91)	65.34 (5.51)	55.37 (9.62)
LLP-FC	90.19 (0.32)	88.53 (0.31)	82.46 (0.49)	80.12 (1.34)	78.89 (0.51)	73.25 (2.26)
LLP-VAT	93.79 (0.29)	91.38 (0.15)	88.22 (0.12)	!	!	!
Ours-Itr-1	95.39 (0.21)	93.89 (0.18)	89.28 (0.35)	85.55 (0.85)	80.43 (1.51)	70.06 (1.03)
Ours-Itr-2	**95.54 (0.22)**	93.97 (0.21)	90.37 (0.27)	85.92 (0.45)	&	&
Dataset:			CIFAR-S			
DLLP	**93.87 (0.11)**	**92.12 (0.24)**	**88.63 (0.51)**	79.58 (1.34)	52.01 (8.56)	57.21 (6.50)
GenBags	92.36 (0.50)	90.10 (0.39)	86.78 (0.33)	82.69 (1.01)	68.45 (3.79)	60.43 (4.49)
EasyLLP	85.54 (1.006)	74.79 (2.17)	65.26 (3.51)	61.57 (9.88)	62.46 (5.21)	52.32 (3.04)
LLP-FC	*	85.58 (0.31)	80.59 (0.56)	75.62 (1.21)	65.75 (2.36)	63.76 (1.26)
LLP-VAT	90.10 (0.49)	83.20 (0.16)	64.76 (3.06)	!	!	!
Ours-Itr-1	93.53 (0.39)	91.29 (0.36)	88.17 (0.59)	83.49 (1.53)	**74.45 (2.58)**	71.01 (2.21)
Ours-Itr-2	93.64 (0.31)	91.31 (0.33)	88.31 (0.41)	**84.30 (1.28)**	&	**71.17 (2.13)**

In Table 3 we report performance on the CIFAR image dataset. We use the SimCLR [6] contrastive learning method to obtain unsupervised embeddings for our experiments on the Image Datasets. We use SimCLR embeddings as input for the relevant baselines we compare. Among the 2 image datasets, CIFAR-S has a large label skew. Our methods outperforms all baseline for CIFAR-S in the large bag size regime ($BagSize \geq 512$), where the performance of all other methods drop significantly. Specifically, we outperform DLLP by upto **20%** and GenBags by upto **7%**. For small bags, our method performs comparable to the best baseline.

Only for CIFAR-B that has label balance, GenBags is better than our method for larger bag sizes while DLLP outperforms slightly for lower bag sizes. However, even in this case, our method is competitive (close second mostly) with the best across bag sizes.

6 Ablations

Here we provide ablations regarding the two most important ideas in our algorithm: 1) Nearest neighbor based nearness constraints and 2) Aggregate Embedding. Due to space constraints, we do provide a number of other ablations in the appendix regarding adding noise to embeddings (A.1.2), hard vs soft thresholding of BP labels (A.1.3), performance change with different choice of kernels (A.1.5) and distance functions (A.1.4) among others. In the supplement, we further report various performance metrics of our algorithm such as performance of BP pseudo labels in itself compared to ground truth (Section A.2.3) and convergence in very few iterations (typically 1-2) (Section A.2.2).

6.1 Time Complexity

For the problem we consider in the paper, we have two terms in the BP formulation: KNN based nearness constraints and Bag constraints. There are m/B bags each having B^2 pairwise terms giving rise to mB pairwise terms (here m is the dataset size and B is the bag size). Similar analysis

Table 4: Time for various parts of our algorithm compared to time taken by DLLP on Criteo Dataset. All time values are in seconds.

	Criteo ∼1m Samples				
Bag Size	DLLP Training	Ours - Data Setup	Ours - BP	Ours - MLP	Ours - Total
8	725.24 (142.59)	1993.47 (96.05)	695.34 (266.7)	679.26 (215.35)	3368.07 (466.34)
32	735.09 (207.50)	1970.05 (104.65)	1279.83 (278.75)	624.55 (187.30)	3874.43 (469.92)
128	568.00 (79.14)	2192.84 (189.90)	3590.73 (528.85)	588.00 (157.84)	6371.57 (727.31)

gives mk pairwise terms for the KNN constraints. So we have an Ising Model with $O(m(B+k))$ pairwise terms. Drawn as an undirected graph, the degree is linear in only $B+k$ BP message passing complexity per node per iteration is also $O(B+k)$. Any implementation will only have this much complexity per node per iteration of BP. `JAX` [4] implementation in `PGMax` [40] does an efficient update for all nodes. This is line with increase in complexity of the BP step (Column 3) for Criteo in Table 4. Additional wall clock time results and discussion (that shows linear scaling in bag size for Adult dataset) is in the supplement section A.2.1, Table 5. This establishes the feasibility of our method on larger datasets and larger bag sizes as well.

6.2 Importance of Nearest Neighbor constraints for BP

One of the main contributions in our algorithm is the usage of nearness of covariates to impose label similarity constraints in the Gibbs distribution is an unsupervised manner. We show how essential it is by removing a certain percentage of those constraints and studying degradation.

A good fraction of kNN constraints is necessary: As depicted in Figure 2, we show how for a good fraction of instances neighbourhood information is essential for good performance of the pseudo labeling step using the *Marketing* dataset. By simply retaining only 10% of the pairwise covariate similarity constraints, we lose around **18%** performance compared to when we use the entire set of pairwise covariate factors. This highlights the importance of the covariate information usage in our BP formulation.

6.3 Learning Aggregate Embeddings Helps

As we highlight in Figure 3, the addition of the additional *bag loss* head in our Aggregate embedding loss in (9) pipeline helps improve performance across both smaller and larger bags. Our choice of average pooling of different instances at bag level during the supervised learning provides the best performance. We also note that we experiment with a much more complex choice for aggregation of instances within a bag like using Multi-Head-Attention (*MHA*). This leads to slight degradation in the instance wise performance. We describe the architecture for this choice in the supplement.

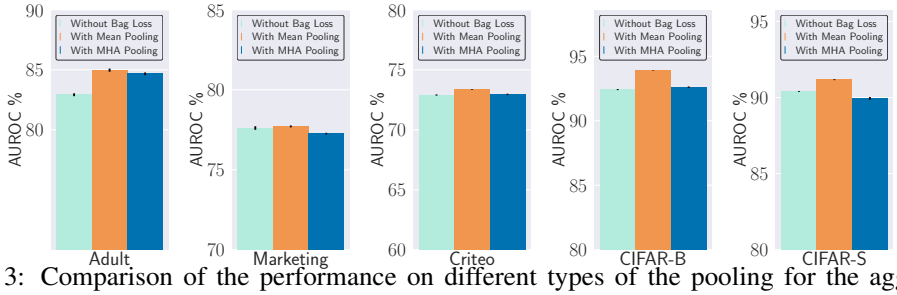

Figure 3: Comparison of the performance on different types of the pooling for the aggregate-embedding loss across different datasets for bag-size 512 for Adult and Marketing and bag-size 32 for Criteo, CIFAR-B and CIFAR-S at the end of iteration 1.

7 Conclusion

Thus we have provided a highly generalizable algorithm to perform efficient learning from label proportions. We utilised Belief Propagation on parity like constraints derived from covariate information and bag level constraints to obtain pseudo labels. Our unique Aggregate Embedding loss used instance wise pseudo labels and bag level constraints to output a final predictor. We have also provided an theoretical insight into why our approach works through varied ablations on different components and extensive experimental comparisons against several SOTA baselines across various datasets of different types. We will publicly release the source code after the review process.

References

[1] Criteo leaderboard, 2023. URL https://paperswithcode.com/sota/click-through-rate-prediction-on-criteo.

[2] Early stopping keras, 2023. URL https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/EarlyStopping.

[3] Ehsan Mohammady Ardehaly and Aron Culotta. Co-training for demographic classification using deep learning from label proportions. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1017–1024. IEEE, 2017.

[4] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[5] Robert Istvan Busa-Fekete, Heejin Choi, Travis Dick, Claudio Gentile, et al. Easy learning from label proportions. *arXiv preprint arXiv:2302.03115*, 2023.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

[8] Cynthia Dwork, Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 11–20, 2014.

[9] Ahmed El Alaoui, Aaditya Ramdas, Florent Krzakala, Lenka Zdeborová, and Michael I Jordan. Decoding from pooled data: Phase transitions of message passing. *IEEE Transactions on Information Theory*, 65(1):572–585, 2018.

[10] Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8 (1):21–28, 1962.

[11] Olivier Chapelle Jean-Baptiste Tien, joycenv. Display advertising challenge, 2014. URL https://kaggle.com/competitions/criteo-display-ad-challenge.

[12] JinHyung Kim and Judea Pearl. A computational model for causal and diagnostic reasoning in inference systems. In *International Joint Conference on Artificial Intelligence*, pages 0–0, 1983.

[13] Ryoma Kobayashi, Yusuke Mukuta, and Tatsuya Harada. Risk consistent multi-class learning from label proportions. *arXiv preprint arXiv:2203.12836*, 2022.

[14] Ron Kohavi et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pages 202–207, 1996.

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[16] Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[17] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[18] Jiabin Liu, Bo Wang, Xin Shen, Zhiquan Qi, and Yingjie Tian. Two-stage training for learning from label proportions. *arXiv preprint arXiv:2105.10635*, 2021.

[19] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[20] Kushal Majmundar, Sachin Goyal, Praneeth Netrapalli, and Prateek Jain. Met: Masked encoding for tabular data. *arXiv preprint arXiv:2206.08564*, 2022.

[21] Sergio Moro, Raul Laureano, and Paulo Cortez. Using data mining for bank direct marketing: An application of the crisp-dm methodology. 2011.

[22] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*, 2016.

[23] Giorgio Patrini, Richard Nock, Paul Rivera, and Tiberio Caetano. (almost) no label no cry. *Advances in Neural Information Processing Systems*, 27, 2014.

[24] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.

[25] Judea Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 129–138. 2022.

[26] Rafael Poyiadzi, Raul Santos-Rodriguez, and Niall Twomey. Label propagation for learning with label proportions. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2018.

[27] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 2002.

[28] Novi Quadrianto, Alex J Smola, Tiberio S Caetano, and Quoc V Le. Estimating labels from label proportions. In *Proceedings of the 25th international conference on Machine learning*, pages 776–783, 2008.

[29] Thomas J Richardson and Rüdiger L Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Transactions on information theory*, 47(2):599–618, 2001.

[30] Rishi Saket, Aravindan Raghuveer, and Balaraman Ravindran. On combining bags to better learn from label proportions. In *International Conference on Artificial Intelligence and Statistics*, pages 5913–5927. PMLR, 2022.

[31] Jonathan Scarlett and Volkan Cevher. Phase transitions in the pooled data problem. *Advances in Neural Information Processing Systems*, 30, 2017.

[32] Clayton Scott and Jianxin Zhang. Learning from label proportions: A mutual contamination framework. *Advances in neural information processing systems*, 33:22256–22267, 2020.

[33] Xingyou Song, Sagi Perel, Chansoo Lee, Greg Kochanski, and Daniel Golovin. Open source vizier: Distributed infrastructure and api for reliable and flexible black-box optimization. In *Automated Machine Learning Conference, Systems Track (AutoML-Conf Systems)*, 2022.

[34] Kuen-Han Tsai and Hsuan-Tien Lin. Learning from label proportions with consistency regularization. In *Asian Conference on Machine Learning*, pages 513–528. PMLR, 2020.

[35] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33:11033–11043, 2020.

[36] Felix X. Yu, Dong Liu, Sanjiv Kumar, Jebara Tony, and Shih-Fu Chang. \proptosvm for learning with label proportions. In *International Conference on Machine Learning*, pages 504–512. PMLR, 2013.

[37] Felix X. Yu, Krzysztof Choromanski, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. On learning from label proportions. *arXiv preprint arXiv:1402.5902*, 2014.

[38] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[39] Jianxin Zhang, Yutong Wang, and Clayton Scott. Learning from label proportions by learning with label noise. *Advances in Neural Information Processing Systems*, 35:26933–26942, 2022.

[40] Guangyao Zhou, Nishanth Kumar, Antoine Dedieu, Miguel Lázaro-Gredilla, Shrinu Kushagra, and Dileep George. Pgmax: Factor graphs for discrete probabilistic graphical models and loopy belief propagation in jax. *arXiv preprint arXiv:2202.04110*, 2022.