On Combining Bags to Better Learn from Label Proportions

Rishi Saket
Google Research
Bangalore, India
rishisaket@google.com

Aravindan Raghuveer
Google Research
Bangalore, India
araghuveer@google.com

Balaraman Ravindran
Google Research
Bangalore, India
balaramanr@google.com

Abstract

In the framework of learning from label proportions (LLP) the goal is to learn a good instance-level label predictor from the observed label proportions of bags of instances. Most of the LLP algorithms either explicitly or implicitly assume the nature of bag distributions with respect to the actual labels and instances, or cleverly adapt supervised learning techniques to suit LLP. In practical applications however, the scale and nature of data could render such assumptions invalid and the many of the algorithms impractical. In this paper we address the hard problem of solving LLP with provable error bounds while being *bag distribution agnostic* and *model agnostic*. We first propose the concept of *generalized bags*, an extension of bags and then devise an algorithm to combine bag distributions, if possible, into *good* generalized bag distributions. We show that (w.h.p) any classifier optimizing the squared Euclidean label-proportion loss on such a generalized bag distribution is guaranteed to minimize the instance-level loss as well. The predictive quality of our method is experimentally evaluated and it equals or betters the previous methods on pseudo-synthetic and real-world datasets.

1 INTRODUCTION

In the *learning from label proportions* (LLP) problem setup, we are given subsets or *bags* of training instances along with only their observed class-label proportions. The goal is to learn a classifier predicting the class-labels of individual instances. Such a setting arises in multiple practical scenarios: a) privacy sensitive data in medical (Wojtusiak et al. (2011)) and anti-fraud (Rueping (2010)) domains b) Instrumentation Limitations in high energy physics (Dery et al. (2017)) c) High Human Annotation Cost in mass-spectrum labeling (Chen et al. (2004)).

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

LLP algorithms are broadly based on either assuming a generative model (e.g. Quadrianto et al. (2009); Patrini et al. (2014)) or applying a classifier as the instance-level predictor and optimizing for the model parameters using suitable bag-level loss functions (e.g. Yu et al. (2013); Musicant et al. (2007); Chen et al. (2009); Stolpe and Morik (2011)). However, prior research does not focus on two key aspects that are critical in practical applications. First, in practical settings, bags are seldom randomly sampled from single simple distributions but rather the drawn from multiple distributions each representing different phenomena of the environment where the bags are generated. Second, prior work has mostly focused on redesigning specific algorithms like SVM, GANs for the LLP use case. While such approaches have significantly advanced the state of the art for LLP, their usability is restricted to situations where that particular modeling algorithm is applicable.

The distribution of the training bags (i.e., their sizes and their constituent instances) is an important factor determining the instance level accuracy yielded by many of the prior LLP algorithms. Obtaining theoretically tight error bounds without making any assumptions on bag distribution and the modeling approach is a hard problem and has not been analyzed in-depth in prior work. In this paper, we focus on both of the above challenges - **a) Distribution Agnostic:** We propose algorithms that are agnostic to the underlying bag distributions that generated the bags. **b) Model Agnostic:** Proposed techniques can work across any models that minimize the simple ℓ_2^2-loss. Our approach provides provable bounds on the instance level prediction error, based on the solutions to certain convex programs.

To show that real, large-scale web applications do exhibit significant variations in bag distributions we pick the problem of product aggregators (PA). In product aggregator services (for e.g., travelocity.com for flights & hotels), there exist three primary actors : the *user*, the *merchant* from who the user eventually buys the product and the *aggregator* that provides a consistent search interface across multiple merchants. The merchant has the instance labels of which leads from the aggregator resulted in a sale. A bag consists of all users whose searched for the product from a merchant. Due to privacy considerations only the percentage of users in the bag that made a purchase (label proportion) is made available to the aggregator. This directly maps to a LLP setting to build a instance level purchase prediction classifier. We describe below three bag distribution characteristics for the above PA example.

1. Intra-bag Instance Correlation: The instances within each bag can be strongly correlated with each other via their features. This violates the class-conditioned independence assumptions made by Quadrianto et al. (2009). Also, certain subsets of instances have a characteristic that bags either contain all/most of them or none of them, which may lead to the predictor not learning the underlying variance in the subset's labels. In the PA example on car aggregator website, all users who are interested in the product *"Volkswagen Golf GTI 2021"* are highly likely to be interested in other compact family cars.

2. Instance Membership Long Tail: The probability of instance membership to a bag is different across instances. Therefore a subset of instances appear more often in some bags than others. This violates the disjointedness property (assumed for example in Yu et al. (2013)), i.e., instances appear in exactly one bag and thus have equal representation. In the PA example for a travel aggregator, frequent travellers often are present in multiple bags while the bulk of casual travellers are present in only one bag.

3. Bag Size Long Tail: A small subset of bags inherently have a large number of instances while there exists a long tail of small bags. In the example of flight aggregators, popular destinations will be considerably larger sized bags than flights to rarer destinations. Due to such a diversity of bag sizes, the effectiveness of what can be learnt from each bag varies significantly.

To address the challenges listed above we make the following contributions in this paper:

Generalized Bags and their efficient sampling: We propose the concept of *generalized bags*, a novel method to linearly combine bag distributions using real-valued weights. Generalized bags provide the means to control the distribution of the resulting aggregation of instances even when the underlying bags exhibit the less desirable characteristics mentioned in the previous paragraphs. To the best of our knowledge, the technique of linearly combining bag distributions has not been studied in LLP literature (Section 3).

We propose an algorithm to efficiently compute a weight distribution (if one exists) derived from the properties of the underlying bag distributions for sampling generalized bags, so that the generalized bag distribution satisfies certain *goodness* properties. (Section 5)

Error Bounds for Generalized Bags: We prove that when the goodness conditions of a generalized bag distribution are satisfied, the bag-level error of any model trained using the ℓ_2^2-loss function on the generalized bag-level label proportions is a nearly tight estimate of its per-instance error (Section 4).

Experimental Validation: We substantiate our theoretical analysis through experiments on pseudo-synthetic datasets that simulate realistic bag distribution scenarios with around 10^4 instances. We also perform evaluations on two real-world datasets of size around 10^7 instances admitting natural bag distributions. In these experiments, our proposed method equals or betters the performance of previous techniques (Section 6).

2 PREVIOUS WORK

Over the last two decades, many specialized approaches to LLP have been developed: de Freitas and Kück (2005) and Hernández-González et al. (2013) proposed MCMC trained probabilistic models, Musicant et al. (2007) provided adaptations of standard supervised learning approaches such as SVM, k-NN and neural nets, and Chen et al. (2009); Stolpe and Morik (2011) gave k-means based clustering methods. The work of Quadrianto et al. (2009) assumed an exponential generative model and provided an efficient algorithm using only bag-level mean estimates. This approach was adapted by Patrini et al. (2014) to more general loss functions for linearized classifiers. A separate class of methods applying SVM to LLP were proposed by Rüping (2010) – using mean of the bags as representative instances – and by Yu et al. (2013) whose $\propto$-SVM method estimated the actual labels and fit an SVM by optimizing the bag and the instance level losses. Subsequently, several deep neural net based techniques have been studied by Kotzias et al. (2015), Bortsova et al. (2018), Ardehaly and Culotta (2017), Liu et al. (2019), Dulac-Arnold et al. (2019), and Shi et al. (2020). Recently, Scott and Zhang (2020) proposed an algorithm which pairs up bags and then trains a model in the mutual contamination framework. However, from the practical perspective (discussed in Sec. 1) the main drawbacks of the previous methods are:

Scalability and complexity, for e.g. the above mentioned MCMC based methods, supervised learning adaptations and the clustering methods by do not scale well for large datasets. The $\propto$-SVM technique runs into issues in loss optimization with ill-fitting data.

Distributional assumptions, such as as class conditioned independence of bags (Quadrianto et al. (2009)), the milder weak distinguishability of bags (Patrini et al. (2014)), and specific types of bag and label proportion distributions (Scott and Zhang (2020); Yu et al. (2014)).

Model specificity, for e.g. of the deep neural net based techniques and assumption of powerful discriminators (e.g. GANs in Liu et al. (2019)).

Multiple Instance Learning (MIL) Dietterich et al. (1997) is another learning setting where an instance level predictor is learned in aggregated bag level labels. However the solutions are not broadly applicable across the two settings since the label aggregation for the MIL setting is logical instead of statistical as in the case of the LLP.

In contrast to previous works ours is the first to consider collections of bag distributions without any *a priori* assumptions on how their feature-vectors or labels are distributed, as would be prevalent in real-world scenarios. In particular, frameworks in which bags are generated conditioned on some class label proportions $\boldsymbol{\gamma}$ (e.g. in Scott and Zhang (2020)) can be modeled by appropriately grouping the bags into k collections – with uniform distribution on each – giving k bag distributions. Any induced dependencies between these bag distributions (due to conditioning on $\boldsymbol{\gamma}$) can also be handled as described in Section 5.4.

Further, our methods are not tied to any particular classifier, which makes our techniques more broadly applicable. Our method can be implemented as a mini-batch training loop and can be made applicable to real-world datasets (Sec. 5.2, 5.3) for which we provide empirical evidence as well (Sec. 6.2, 6.3).

3 GENERALIZED BAGS

Bags can naturally be visualized as corresponding indicator vectors in $\mathbb{R}^n$ where the instance set is indexed by $[n]$. The idea behind generalized bags is to *linearly* transform a collection of bag vectors into new ones which satisfy certain uniformity conditions we define later. We show that such transformations, if they exist, can be used to solve the same LLP problem with tighter bounds on instance level prediction errors.

We now define the formal notations used for the remainder of the paper. Instances are N-dimensional feature vectors with a label set $[L]$. Let (i) X be a dataset (indexed by $[n]$) of feature-vectors $\{\mathbf{x}^{(i)} \in \mathbb{R}^N\}_{i=1}^n$, (ii) $\mathsf{Y} = \{\mathbf{y}^{(i)} \in \{0,1\}^L\}_{i=1}^n$ be the one-hot observed-label encodings, and (iii) $\widehat{\mathsf{Y}} = \{\widehat{\mathbf{y}}^{(i)} \in \mathbb{R}_+^L\}_{i=1}^n$ denote the model-predicted labels. A *bag* $B \subseteq [n]$ corresponds to a subset of feature-vectors $\mathsf{X}_B := \cup_{i\in B}\mathbf{x}^{(i)}$, with its observed and predicted *label-histograms* $\mathbf{y}_B := \sum_{i\in B}\mathbf{y}^{(i)}$ and $\widehat{\mathbf{y}}_B := \sum_{i\in B}\widehat{\mathbf{y}}^{(i)}$. A *generalized bag* $\bar{B}$ is a weighted sum of bags B_j represented as $\sum_{j\geq 1} w_j B_j$, and analogously its observed and predicted label-histograms are $\mathbf{y}_{\bar{B}} = \sum_{j\geq 1} w_j \mathbf{y}_{B_j}$ and $\widehat{\mathbf{y}}_{\bar{B}} = \sum_{j\geq 1} w_j \widehat{\mathbf{y}}_{B_j}$. To aid our subsequent analysis it will be useful to uniquely identify bags and generalized bags with n-dimensional vectors. Given a bag $B \subseteq [n]$, let $\mathbb{1}_B \in \{0,1\}^n$ be its *characteristic* (indicator) vector. For a generalized bag $\bar{B} = \sum w_j B_j$ where B_j are bags, its characteristic vector is $\mathbf{Z}_{\bar{B}} := \sum w_j \mathbb{1}_{B_j}$. A *bag distribution* is a distribution over all possible pairs of bags and their label histograms $(B, \mathbf{y}_B)$. Similarly, a *generalized bag distribution* is over all possible generalized bags and their label histograms $(\bar{B}, \mathbf{y}_{\bar{B}})$ defined over the space of instances. Analogous to the above, bag distributions can be combined to form generalized bag distributions: given a collection of bag distributions $D_1, \ldots, D_k$, and weights $w_1, \ldots, w_k$, a generalized bag distribution can be obtained by independently sampling $(B_j, \mathbf{y}_{B_j}) \leftarrow D_j$ $(j \in [k])$ and outputting $(\bar{B}, \mathbf{y}_{\bar{B}}) = \left(\sum_{j\geq 1} w_j B_j, \sum_{j\geq 1} w_j \mathbf{y}_{B_j}\right)$.

We note that the results of this paper remain applicable even if bags are taken to be multisets (see Sec. 5.4). However, for ease of exposition we stick to bags being sets.

4 LOSS BOUNDS

For an LLP dataset with observed and predicted labels $\mathsf{Y}, \widehat{\mathsf{Y}}$, the instance-wise squared Euclidean loss is $\mathsf{L}_{\mathsf{rse}}(\mathsf{X}, \mathsf{Y}, \widehat{\mathsf{Y}}) := \sum_{i=1}^n \|\mathbf{y}^{(i)} - \widehat{\mathbf{y}}^{(i)}\|_2^2$. For a collection of bags or generalized-bags $\mathcal{B}$ the generalized bag-level loss is $\mathsf{L}_{\mathsf{bse}}(\mathsf{X}, \mathsf{Y}, \widehat{\mathsf{Y}}, \mathcal{B}) := (1/\sqrt{|\mathcal{B}|}) \sum_{\bar{B}\in\mathcal{B}} \|\mathbf{y}_{\bar{B}} - \widehat{\mathbf{y}}_{\bar{B}}\|_2^2$. (We scale by $1/\sqrt{|\mathcal{B}|}$ for ease of notation).

The losses defined above can be conveniently rewritten as follows. Let $\mathbf{Y}$ be a $n \times L$ matrix whose ith row is $\left(\mathbf{y}^{(i)}\right)^{\mathsf{T}}$, and let $\widehat{\mathbf{Y}}$ be similarly defined with $\left(\widehat{\mathbf{y}}^{(i)}\right)^{\mathsf{T}}$ as the ith row. Then, $\mathsf{L}_{\mathsf{rse}}(\mathsf{X}, \mathsf{Y}, \widehat{\mathsf{Y}}) = \|\mathbf{Y} - \widehat{\mathbf{Y}}\|_F^2$ where $\|.\|_F$ is the matrix Frobenius norm. For a generalized-bag collection $\mathcal{B}$, define the matrix $\mathbf{A} := \mathbf{A}_{\mathcal{B}}$ having $(\mathbf{Z}_{\bar{B}})^{\mathsf{T}}$ (where $\mathbf{Z}_{\bar{B}}$ is its characteristic vector defined in Sec. 3) as its rows for each $\bar{B} \in \mathcal{B}$ and (for convenience) normalized by $1/\sqrt{|\mathcal{B}|}$. Then,

$$\mathsf{L}_{\mathsf{bse}}(\mathsf{X}, \mathsf{Y}, \widehat{\mathsf{Y}}, \mathcal{B}) = \|\mathbf{A}\mathbf{Y} - \mathbf{A}\widehat{\mathbf{Y}}\|_F^2.$$

It is can be seen (ref. Lemma 4.1 below) that $\mathsf{L}_{\mathsf{bse}}$ and

$\mathsf{L}_{\mathsf{rse}}$ are within a factor $\sigma^2 \in [\sigma_{\min}(\mathbf{A})^2, \sigma_{\max}(\mathbf{A})^2]$ of each other. Here $\sigma_{\max}(\mathbf{A}) \geq \sigma_{\min}(\mathbf{A})$ are the maximum and minimum singular values of $\mathbf{A}$. Formally:

$$\sigma_{\min}(\mathbf{A}) := \inf_{\mathbf{x}, \|x\|_2=1} \|\mathbf{Ax}\|_2,\ \sigma_{\max}(\mathbf{A}) = \sup_{\mathbf{x}, \|x\|_2=1} \|\mathbf{Ax}\|_2$$

The condition number of $\mathbf{A}$, $\mathsf{cond}(\mathbf{A})$ is defined as the ratio (if finite) $\sigma_{\max}(\mathbf{A})/\sigma_{\min}(\mathbf{A})$. For $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n\times L}$, we have the following.

Lemma 4.1. $\sigma_{\min}(\mathbf{A}) \leq \frac{\|\mathbf{AU}-\mathbf{AV}\|_F}{\|\mathbf{U}-\mathbf{V}\|_F} \leq \sigma_{\max}(\mathbf{A})$. *In particular, if* $\|\mathbf{AU} - \mathbf{AV}\|_F \leq \varepsilon$ *then* $\|\mathbf{U} - \mathbf{V}\|_F \leq \varepsilon/\sigma_{\min}(\mathbf{A})$, *and if* $\|\mathbf{U}-\mathbf{V}\|_F \leq \varepsilon$ *then* $\|\mathbf{AU}-\mathbf{AV}\|_F \leq \varepsilon\sigma_{\max}$.

Proof. Letting $\mathbf{U}^{(l)}$ and $\mathbf{V}^{(l)}$ be the lth columns of $\mathbf{U}$ and $\mathbf{V}$ respectively for $l \in [L]$, we have

$$\sigma_{\min}(\mathbf{A})^2 \leq \min_l \frac{\|\mathbf{A}(\mathbf{U}^{(l)} - \mathbf{V}^{(l)})\|_2^2}{\|\mathbf{U}^{(l)} - \mathbf{V}^{(l)}\|_2^2} \leq \frac{\|\mathbf{AU} - \mathbf{AV}\|_F^2}{\|\mathbf{U} - \mathbf{V}\|_F^2}$$
$$\leq \max_l \frac{\|\mathbf{A}(\mathbf{U}^{(l)} - \mathbf{V}^{(l)})\|_2^2}{\|\mathbf{U}^{(l)} - \mathbf{V}^{(l)}\|_2^2} \leq \sigma_{\max}(\mathbf{A})^2$$

□

Sampling Well-Conditioned Matrices. A distribution over vectors $\mathbf{Z} \in \mathbb{R}^n$ is *isotropic* if $\mathbb{E}\left[\mathbf{ZZ}^{\mathsf{T}}\right] = \mathbf{I}$. The following result by Vershynin (2012b) shows that independently sampling rows of $\mathbf{A}$ (and suitably normalizing) from an *isotropic* distribution of vectors $\mathbf{Z}$ leads to $\sigma_{\max}(\mathbf{A}), \sigma_{\min}(\mathbf{A}) \approx 1$ w.h.p when $\mathbf{A}$ is tall enough depending on $\mathbf{Z}$'s maximum norm.

Theorem 4.2. (Theorem 5.41 in Vershynin (2012b)) *Let $\tilde{\mathbf{A}}$ be $m \times n$ matrix with rows independently sampled from the isotropic distribution above satisfying $\|\mathbf{Z}\|_2 \leq \sqrt{d}$ almost surely, for some number d. Then, there is an absolute constant c such that $\mathbf{A} = (1/\sqrt{m})\tilde{\mathbf{A}}$ satisfies,*

$$1 - t\sqrt{d/m} \leq \sigma_{\min}(\mathbf{A}) \leq \sigma_{\max}(\mathbf{A}) \leq 1 + t\sqrt{d/m},$$

with probability at least $1-2n\cdot\exp(-ct^2)$, for any $t \geq 0$. In particular, choosing $t = \sqrt{(4\ln n + \ln 2)/c}$ and $m = d(t/\delta)^2$ for some $\delta > 0$ we obtain that with probability at least $1 - n^{-3}$, $\sigma_{\min}(\mathbf{A}), \sigma_{\max}(\mathbf{A}) \in [1-\delta, 1+\delta]$.

If these bounds on singular values hold for $\mathbf{A} := \mathbf{A}_{\mathcal{B}}$, then Lemma 4.1 implies that optimizing the generalized bag-level loss optimizes the instance-level loss and vice-versa, *no matter what the actual instance labels are.*

5 SAMPLING GENERALIZED BAGS

Given multiple bag-distributions $D_1, \ldots, D_k$, we provide a method to efficiently sample a weight vector $\mathbf{w} = (w_1, \ldots, w_k)$ from a distribution (if it exists) ensuring that the generalized bag $\sum_i w_i B_i$ (where $B_1, \ldots, B_k$ are independent samples from $D_1, \ldots, D_k$ respectively) induces an isotropic distribution $\mathbf{Z}$ on its characteristic vector (ref. Sec. 4). The algorithm formulates the isotropic condition as linear equations over the second moments of the desired distribution with coefficients from the second-moment matrices of $D_1, \ldots, D_k$. Solving the resultant semi-definite program (SDP) and sampling using the obtained second-moment matrix yields $\mathbf{w}$. We minimize the trace of the solution matrix which helps (as shown in Sec. 5.1) in bounding the maximum norm of $\mathbf{Z}$, thus limiting the number of generalized-bags samples $\mathcal{B}$ for guaranteeing (w.h.p) $\sigma_{\min}(\mathbf{A}), \sigma_{\max}(\mathbf{A}) \in [1-\delta, 1+\delta]$ for $\mathbf{A} := \mathbf{A}_{\mathcal{B}}$.

In the rest of this section $(B_i \subseteq [n]) \leftarrow D_i$ $(1 \leq i \leq k)$ shall be independently sampled bags and define $\mathbf{Q}^{(i)} := \mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_i}^{\mathsf{T}}\right]$. For a fixed $\mathbf{w} = (w_1, \ldots, w_k)$, define the random generalized bag $\bar{B} := \sum_{i=1}^k w_i B_i$ with characteristic vector $\mathbf{Z} := \sum_{i=1}^k w_i \mathbb{1}_{B_i} \in \mathbb{R}^n$ and $\mathbf{M}^{(\mathbf{w})} := \mathbb{E}\left[\mathbf{ZZ}^{\mathsf{T}} \mid \mathbf{w}\right]$. The following shows how $\mathbf{M}^{(\mathbf{w})}$ is determined by quadratic forms over $\mathbf{w}$ with coefficients from entries of $\mathbf{Q}^{(i)}$.

Lemma 5.1. *For any two $u, v \in [n]$,* $\mathbf{M}^{(\mathbf{w})}_{u,v} = \sum_{i=1}^k w_i^2 \mathbf{Q}^{(i)}_{u,v} + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} w_i w_j \mathbf{Q}^{(i)}_{u,u}\mathbf{Q}^{(j)}_{v,v}$.

Proof. Note that for any $u, v \in [n]$, the (u, v)th coordinate of $\mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_i}^{\mathsf{T}}\right]$ is $\mathbf{Q}^{(i)}_{u,v}$. The same for $\mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_j}^{\mathsf{T}}\right]$ $(i \neq j)$ equals $\Pr[u \in B_i]\Pr[v \in B_j]$ (since B_i, B_j are independently sampled) which is $\mathbf{Q}^{(i)}_{u,u}\mathbf{Q}^{(j)}_{v,v}$ by definition of $\mathbf{Q}^{(i)}, \mathbf{Q}^{(j)}$. This along with the expansion of $\mathbf{M}^{(\mathbf{w})}$ as

$$\mathbb{E}\left[\left(\sum_{i=1}^k w_i \mathbb{1}_{B_i}\right)\left(\sum_{i=1}^k w_i \mathbb{1}_{B_i}\right)^{\mathsf{T}}\right]$$
$$= \mathbb{E}\left[\sum_{i=1}^k w_i^2 \mathbb{1}_{B_i}\mathbb{1}_{B_i}^{\mathsf{T}} + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} w_i w_j \mathbb{1}_{B_i}\mathbb{1}_{B_j}^{\mathsf{T}}\right]$$
$$= \sum_{i=1}^k w_i^2\, \mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_i}^{\mathsf{T}}\right] + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} w_i w_j\, \mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_j}^{\mathsf{T}}\right] \quad (1)$$

and matching the (u, v)th coordinates on the both sides completes the proof. □

If $\mathbf{w}$ is sampled from some G_w independently of B_i

$(i \in [k])$, then letting $\mathbf{M} := \mathbb{E}_{G_w}\left[\mathbf{M}^{(\mathbf{w})}\right]$

$$\begin{aligned}\mathbf{M}_{u,v} &= \sum_{i=1}^{k} \mathbb{E}[w_i^2]\mathbf{Q}^{(i)}_{u,v} + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} \mathbb{E}[w_iw_j]\mathbf{Q}^{(i)}_{u,u}\mathbf{Q}^{(j)}_{v,v} \\ &= \sum_{i=1}^{k} \mathbf{W}_{i,i}\mathbf{Q}^{(i)}_{u,v} + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} \mathbf{W}_{i,j}\mathbf{Q}^{(i)}_{u,u}\mathbf{Q}^{(j)}_{v,v} \qquad (2)\end{aligned}$$

where $\mathbf{W}$ is the psd (positive semi-definite) matrix $\mathbb{E}_{G_w}[\mathbf{w}\mathbf{w}^{\mathsf{T}}]$. Thus, if there exists G_w such that $\mathbf{Z}$ is isotropic i.e., $\mathbf{M} = \mathbf{I}$ then there is a psd $\mathbf{W} \in \mathbb{R}^{k\times k}$ s.t. the constraint (4) in Fig. 2 is satisfied. Since the constraints and the trace objective in Fig. 2 are linear in the entries of psd $\mathbf{W}$ it is indeed an SDP which can be solved in polynomial time using interior point or ellipsoid methods. If SDP-I in Fig. 2 feasible, then using the psd decomposition $\mathbf{W} = \sum_{r=1}^{k} \mathbf{a}^{(r)}\left(\mathbf{a}^{(r)}\right)^{\mathsf{T}}$ and sampling $\sqrt{k}\mathbf{a}^{(r)}$ $(r \in [k])$ uniformly as $\mathbf{w}$ gives us the desired distribution. To minimize the norm of $\mathbf{Z}$ we add the objective in SDP-I minimizing the trace of $\mathbf{W}$ which – as we show below – yields approximate bound on the norm of $\mathbf{Z}$. Note that instead of sampling using the psd decomposition one can also take $\mathbf{w}$ to be mean-zero Gaussian vector sampled using the covariance matrix $\mathbf{W}$, a more convenient procedure in practice. Algorithm 1 returns the desired distribution (decomposition or Gaussian based), while Algorithm 2 uses the output of the former along with the bag distributions to sample generalized bags. Both these algorithms appear in Fig. 1.

5.1 Bounding the sample size $|\mathcal{B}|$

We bound the norm of $\mathbf{Z}$ corresponding to a generalized bag $\sum_{i=1}^{k} w_iB_i$ using weights given by the decomposition method in Alg. 1. First, observe that if $\mathbf{W} = \sum_{r=1}^{k} \mathbf{a}^{(r)}\left(\mathbf{a}^{(r)}\right)^{\mathsf{T}}$ then $\mathsf{tr}(\mathbf{W}) = \sum_{r=1}^{k} \|\mathbf{a}^{(r)}\|_2^2$. Letting $b_{ij} = \mathbb{1}_{\{j\in B_i\}}$ $(j \in [n])$, the jth coordinate of $\mathbf{Z}$ is $\sum_{i=1}^{k} w_ib_{ij}$ which is at most $\left(\left(\sum_{i=1}^{k} b_{ij}\right)\sum_{i=1}^{k} w_i^2\right)^{\frac{1}{2}}$ using Cauchy-Schwartz and the fact that $b_{ij} \in \{0,1\}$. Thus, $\|\mathbf{Z}\|_2 \leq \|\mathbf{w}\|_2\sqrt{\sum_{i,j} b_{ij}} \leq \|\mathbf{w}\|_2\sqrt{nk}$. Since $\|\mathbf{w}\|_2 \leq \max_r \sqrt{k}\|\mathbf{a}^{(r)}\|_2 \leq \sqrt{k\cdot\mathsf{tr}(\mathbf{W})}$, we obtain $\|\mathbf{Z}\|_2 \leq k\sqrt{n\cdot\mathsf{tr}(\mathbf{W})}$.

Applying Theorem 4.2, we can take the number of generalized bags m to be sampled as $O\left((\mathsf{opt}\cdot k^2 n\log n)/\delta^2\right)$ to obtain that with probability at least $1 - n^{-3}$, $\sigma_{\min}(\mathbf{A}), \sigma_{\max}(\mathbf{A}) \in [1-\delta, 1+\delta]$, where opt is the value of the SDP solved in Alg. 1.

Algorithm 1 FIND-WT-DIST

```
Input: $\left\{\mathbf{Q}^{(i)} \mid i \in [k]\right\}$, option $\in$ {decomp, gauss}
if SDP-I in Fig. 2 is infeasible then Return nil end if
Let $\mathbf{W}$ be an optimizer for SDP-I.
if option = decomp, Decompose $\mathbf{W} = \sum_{r=1}^{k} \mathbf{a}^{(r)}\left(\mathbf{a}^{(r)}\right)^{\mathsf{T}}$.
    Return $G_w$ uniform on $\left\{\sqrt{k}\mathbf{a}^{(r)} \mid 1 \leq r \leq n\right\}$. end if
if option = gauss then Return $G_w$ as $\mathbf{N}(0,\mathbf{W})$ end if
```

Algorithm 2 SAMPLE-GEN-BAG

```
Input: Ind. distns. $G_w$ over $\mathbb{R}^k$, bag distns. $D_1,\ldots,D_k$.

Independently sample (bag, histogram) pairs $(B_i,\sigma_i) \leftarrow D_i$  $(1 \leq i \leq k)$.
Independently sample $\mathbf{w} = (w_1,\ldots,w_k) \leftarrow G_w$.
Return (gen. bag, histogram) pair $(B,\sigma) = \left(\sum_{i=1}^{k} w_iB_i, \sum_{i=1}^{k} w_i\sigma_i\right)$.
```

Figure 1: Procedures For Sampling Generalized Bags.

$$\min \mathsf{tr}(\mathbf{W}) \quad \text{s.t. } \mathbf{W} \succeq \mathbf{0}, \text{ and } \forall u,v \in [n], \quad (3)$$

$$\sum_{i=1}^{k} \mathbf{W}_{i,i}\mathbf{Q}^{(i)}_{u,v} + \sum_{\substack{1\leq i,j\leq k\\ i\neq j}} \mathbf{W}_{i,j}\mathbf{Q}^{(i)}_{u,u}\mathbf{Q}^{(j)}_{v,v} = \mathbb{1}_{\{u=v\}}. \qquad (4)$$

Figure 2: Semi-definite Program SDP-I

5.2 Training Setup

Let $\mathcal{M}$ be the instance-level model sought to be trained using the (training) bag distributions $D_1,\ldots,D_k$. The training setup consists of the number of (mini-)batch training steps N_{train}, mini-batch size Q_{train}, and optimizer $\mathcal{F}_{\text{train}}$. We apply Algorithms 1, 2 to sample generalized bags. At each of the N_{train} training steps the training algorithm:

1. constructs a (mini-)batch $\bar{\mathcal{B}}$ of R_{train} independently sampled (generalized-bag, histogram) pairs $\{(B_r, \mathbf{y}_{B_r})\}_{r=1}^{R}$,
2. using labels predicted by $\mathcal{M}$ computes the predicted label histograms $\widehat{\mathbf{y}}_{B_r}$ $(1 \leq r \leq R)$,
3. computes the mini-batch loss: $\mathsf{L}_{\mathsf{bse}}(\bar{\mathcal{B}}) = \left(1/\sqrt{R}\right)\sum_{r=1}^{R} \|\mathbf{y}_{B_r} - \widehat{\mathbf{y}}_{B_r}\|^2$,
4. updates the model $\mathcal{M} \leftarrow \mathcal{F}\left(\mathcal{M}, \mathsf{L}_{\mathsf{bse}}(\bar{\mathcal{B}})\right)$.

The average size of a generalized bag (in terms its underlying instances) is at most $\sum_{j=1}^{k} \mu_j = k\mu$ where μ_j

is the average bag size from D_j, and $\mu = (\sum_j \mu_j)/k$. In particular, the average size of a mini-batch of generalized bags is within a factor of k of the same for bags. In practice (see Sec. 6.2), most of the bags are much smaller than the total number of instances, and those which are larger than a threshold can be omitted from the training without much impact on performance, thereby bounding the computational cost of the training loop.

5.3 Real-world bag distributions

We have provided in this section an exposition, along with provable performance bounds, of our generalized bags based method for LLP. In real world scenarios the bag distributions depend on the actual application - for example each seller could be a source of bags in the Product Aggregator (PA) scenario.

The only idealized assumption of our method is that SDP-I (Figure 2) can be feasibly solved. However, it may be infeasible/difficult to solve over the set of instances in real-world data. Nevertheless, the complexity of the problem can be reduced by heuristically choosing a representative sample of instances (e.g. by identifying clusters of instances together) and satisfying the constraints to a sufficient approximation. The second-moment matrices $\mathbf{Q}^{(i)}$ can be computed by sampling around $O(n \log n)$ bags (where n is the number of reduced instances), which provides good estimates for well-behaved distributions (Vershynin (2012a)).

In case SDP-I remains infeasible, we can approximately satisfy the constraints by adding a variable $y_{uv} \geq 0$ (one for each tuple $(u, v) \in [n] \times [n]$) satisfying,

$$\left| \sum_{i=1}^{k} \mathbf{W}_{i,i} \mathbf{Q}^{(i)}_{u,v} + \sum_{\substack{1 \leq i,j \leq k \\ i \neq j}} \mathbf{W}_{i,j} \mathbf{Q}^{(i)}_{u,u} \mathbf{Q}^{(j)}_{v,v} - \mathbb{1}_{\{u=v\}} \right| \leq y_{uv} \quad (5)$$

and minimizing $\mathsf{tr}(\mathbf{W}) + \lambda \sum_u y_{uu} + \lambda' \sum_{u,v \,|\, u \neq v} y_{uv}$ for appropriate $\lambda, \lambda' \geq 0$.

We empirically show in Sections 6.2 and 6.3 that even basic implementations of the above approximations – in particular, using the values of a single categorical feature as a proxy for instances to solve SDP-I – applied to real-world datasets provides better performance than previous baselines. An in-depth investigation of the above mentioned clustering and SDP approximation techniques is beyond the scope of this work.

5.4 Multiset bags and correlated bag distributions

Our techniques presented in this section can be extended to the case of bags being multisets. In this case, the characteristic vector $\mathbb{1}_B$ of a bag B has non-negative integer entries (rather than only $\{0, 1\}$). Thus, the diagonal of $\mathbf{Q}^{(i)}$ is no longer $\boldsymbol{\mu}^{(i)} := \mathbb{E}[\mathbb{1}_{B_i}]$ where $\mathbf{Q}^{(i)}$ is $\mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_i}^{\mathsf{T}}\right]$ as before, for the ith bag distribution ($i \in [k]$). Therefore, the (u, v)th coordinate of $\mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_j}^{\mathsf{T}}\right]$ ($i \neq j$) is $\boldsymbol{\mu}^{(i)}_u \boldsymbol{\mu}^{(j)}_v$, and one can appropriately modify (2) and (4).

The techniques can also be applied when the bags B_i ($i = 1, \ldots, k$) are sampled from a joint distribution over k-tuples of bags. In this case, the explicit computations of $\mathbb{E}\left[\mathbb{1}_{B_i}\mathbb{1}_{B_j}^{\mathsf{T}}\right]$ can be used in (2) and (4).

6 EXPERIMENTS

6.1 Pseudo-synthetic datasets

In Section 1, we motivated through examples three dimensions of variations in bag distributions. i) Intra-Bag Instance correlation ii) Instance Membership Long Tail iii) Bag Size Long tail. In order to examine the performance of our and other algorithms in these bag distribution variations, we synthetically generate bags from the following publicly available datasets from the UCI repository (Dua and Graff (2017)) (also used by Patrini et al. (2014)): Ionosphere Data Set ion and Statlog Australian Credit Data Set (sta).

Dataset Creation. Table 1 lists the characteristics of the different simulated bag distribution scenarios I - VII. It is notable that our scenarios are much more detailed than studied in the previous works. We now describe the generative process for these scenarios. The set of training instances are first partitioned into C clusters (varying with the setup) using K-MEANS applied to the feature vectors. For each of the bag distribution characterstic in Section 1, we control generation as follows:
Intra-Bag Instance Correlation: By selecting multiple instances from the same cluster to form bags, with different bag distributions per cluster. (Seen in scenarios II-VII)
Instance Membership Long Tail: By sampling different proportions of bags from different clusters. (Seen in scenarios III - V)
Bag Size: Controlled by having different fixed Bernoulli sampled bags, and by varying the parameter of the Bernoulli distribution sampled from one or more power-law distributions. Scenarios II, III and V-VII use different fixed Bernoullis, and scenario IV uses power law to sample the Bernoulli parameter.

Scn.	Description	#Clust. C	Distribution per Cluster	# dist.	# Bags/Clust.
I	Small bags size.	1	$Ber(0.1)$, 2 iid copies	2	125 per clust. Total 250
II	Large & small bags intra-bag corr.	3	$Ber(0.1)$, $Ber(0.9)$	6	80 per clust. Total 240
III	Large & small bags, intra-bag corr., unbal. inst. membership	3	$Ber(0.1)$, $Ber(0.9)$	6	$30, 90, 150$ Total 270
IV	Powerlaw bags, intra-bag corr., unbal. inst. membership	3	$Ber(p)$, $p \sim$ plaw(1.66) 2 iid copies	6	$30, 90, 150$ Total 270
V	Bags Straddle Clusts. Feasible SDP, intra-bag corr.	3	$(Ber(0.2), Ber(0.2), Ber(0.2))$ from (C_1, C_2, C_3): Straddle Bags Distn., $\{Ber(0.2)$ from $C_i\}_{i=1}^3$	4	60 per distn. Total 240
VI	Bags Straddle Clusts. Infeasible SDP, intra-bag corr.	3	$(Ber(0.4), Ber(0.8), Ber(0.8))$ from (C_1, C_2, C_3): Straddle Bags Distn., $\{Ber(0.2)$ from $C_i\}_{i=1}^3$	4	60 per distn. Total 240
VII	Two Bag distns straddle Clusts. Feasible SDP, intra-bag corr.	3	$(Ber(0.2), Ber(0.2))$ from (C_1, C_2), $(Ber(0.6), Ber(0.6))$ from (C_2, C_3) : Straddle Bags Distns., $\{Ber(0.2)$ from $C_i\}_{i=1}^3$	4	60 per distn. Total 240

Table 1: Bag Distributions Scenarios.

Table 2: AUC for *Ionosphere* (%). Instance level oracle (Logistic Reg.) has AUC of 91.87 ± 3.05.

#	MM	LMM (v(G,s))	AMM (MM)	AMM (LMM) (v(G,s)))	LMMCM	LIN (KL-div, U)	LIN (ℓ_2^2, S) Our Method
I	84.41 ± 6.2	84.41 ± 5.4	89.91 ± 3.7	89.95 ± 4.1	$\mathbf{94.3} \pm 3.1$	88.08 ± 3.8	87.48 ± 5.0
II	77.84 ± 6.8	80.74 ± 6.6	80.66 ± 11.4	80.75 ± 11.2	84.83 ± 6.7	79.83 ± 5.4	$\mathbf{85.42} \pm 5.4$
III	75.07 ± 7.8	80.04 ± 7.9	77.14 ± 9.2	78.28 ± 12.6	74.13 ± 14.3	80.42 ± 4.9	$\mathbf{87.62} \pm 4.5$
IV	75.93 ± 7.0	79.7 ± 7.6	74.82 ± 11.6	75.34 ± 11.8	70.7 ± 18.5	79.2 ± 5.8	$\mathbf{81.46} \pm 9.1$
V	77.51 ± 7.3	84.95 ± 6.8	75.96 ± 14.8	76.07 ± 13.3	84.77 ± 7.2	83.23 ± 4.6	$\mathbf{86.31} \pm 5.8$
VI	78.3 ± 7.6	84.81 ± 6.9	84.72 ± 6.6	84.66 ± 7.5	80.15 ± 6.7	84.03 ± 4.4	$\mathbf{87.79} \pm 4.8$
VII	79.37 ± 7.6	82.25 ± 6.9	79.17 ± 9.8	79.44 ± 11.8	81.75 ± 6.7	86.05 ± 3.7	$\mathbf{87.12} \pm 3.8$

Scenario I is a vanilla case of small bags. In I-IV, for each cluster we have two independent bag distributions given by Bernoulli sampling instances independently into bags. In II and III these correspond to contrasting bag sizes, while in IV each bag distribution is a mixture of such Bernoulli samplings with (for each bag) the probability first drawn from the same powerlaw distribution. In V-VI we have one bag distribution which samples from across the three clusters, while in VII we have two bag distributions straddling two clusters each. Scenario VI has straddle bags with different cluster biases, does not admit a feasible SDP-I (Fig. 2), and we use an approximate solution to sample the weights.

Experiment Setup. We compare against the following algorithms from previous works: (i) the following mean-map (MM) based methods: the original MM of Quadrianto et al. (2009); LMM ($v^{G,s}$), AMM(MM) and AMM (LMM (v^{GS})) (min versions) of Patrini et al. (2014); (iii) the LMMCM method from Scott and Zhang (2020), and the (iv) KL-divergence loss baseline from Ardehaly and Culotta (2017). The implementations of (i) and (ii) are adapted from the code made available by Patrini et al. (2014), and of (iii) from the code made available by Scott and Zhang (2020). Our generalized bag based model training given in Sec. 5.2, implemented in TensorFlow 2 and instantiated with a linear classifier (with sigmoid activation), is denoted by LIN(ℓ_2^2, S) and is trained as per Sec. 5.2 while the loss formulation of Ardehaly and Culotta (2017) for uniformly sampled bags, LIN(KL-div, U), was similarly implemented for training the same linear model. Since our focus is to validate our generalized bag based training, among the recent state of the art methods, we choose to compare against training and loss frameworks such as Scott and Zhang (2020); Ardehaly and Culotta (2017) rather than model (DNN, GAN etc) specific methods.

We also compare the following variations in our method within the minibatch training : (i) using original bags sampled from uniformly at random (u.a.r.) chosen bag distribution instead generalized bags LIN(ℓ_2^2, U), (ii) using iid $N(0, 1)$ weights $w_1, \dots, w_k$ to construct the generalized bags, LIN(ℓ_2^2, R).

For each experiment we recorded the AUCs for the above scenarios. The experiments were done using 5-times 5-fold cross validation of each dataset. For each of the 5-folds, and the 5-test/train splits per fold, and each scenario I-VII, the bag distributions are created only from the instance-level training set. The train set consists of separate sets of bags sampled from the different bag distributions (as shown in Table 1) provided along with each bag's label proportions. The test set however is still at the instance level. A random bag from a bag distribution is generated by sampling uniformly from the corresponding training subset of

Table 3: AUC for *Australian* (%). Instance level oracle (Logistic Reg.) has AUC of 93.47 ± 2.4.

#	MM	LMM (v(G,s))	AMM (MM)	AMM (LMM) (v(G,s)))	LMMCM	LIN (KL-div, U)	LIN (ℓ_2^2, S) Our Method
I	89.78 ± 3.0	89.92 ± 3.0	92.26 ± 2.3	92.29 ± 2.3	92.1 ± 2.8	90.81 ± 2.3	$\mathbf{92.31} \pm 2.4$
II	83.38 ± 2.9	86.83 ± 3.0	86.3 ± 4.7	86.39 ± 4.7	$\mathbf{90.82} \pm 2.8$	87.62 ± 2.5	90.15 ± 2.8
III	85.72 ± 2.4	86.85 ± 2.8	85.74 ± 3.5	85.81 ± 3.7	90.0 ± 3.0	87.13 ± 2.6	$\mathbf{90.88} \pm 2.6$
IV	85.78 ± 2.3	86.7 ± 3.0	85.14 ± 3.6	85.26 ± 4.1	$\mathbf{89.63} \pm 3.6$	86.49 ± 2.4	85.12 ± 3.2
V	83.75 ± 2.2	88.35 ± 2.6	87.79 ± 3.6	87.85 ± 3.6	$\mathbf{90.44} \pm 2.6$	88.08 ± 2.3	90.34 ± 2.7
VI	84.41 ± 2.8	87.63 ± 2.9	88.28 ± 3.3	88.47 ± 3.8	90.21 ± 2.6	87.12 ± 4.5	$\mathbf{91.43} \pm 2.5$
VII	84.72 ± 2.5	87.38 ± 3.0	87.95 ± 3.6	87.91 ± 3.5	90.21 ± 2.9	89.07 ± 2.9	$\mathbf{91.53} \pm 2.7$

bags. The algorithms in previous works do not consider multiple bag distributions and are provided the entirety of the training set of bags as a whole.

Algorithms MM of Quadrianto et al. (2009), LMM ($v^{G,s}$), AMM(MM) and AMM (LMM (v^{GS})) (min versions) of Patrini et al. (2014), and LMMCM of Scott and Zhang (2020) are executed for all of their parameter ranges as described in the respective works, except for the $\lambda = 10^{-5}$ setting in LMMCM which was observed to cause instability and timeouts. The performance of the best parameter setting for each scenario was taken for each of the algorithms. The classifier in our method was optimized using SGD with training rate 10^{-4} over $N_{\text{train}} = 1000$ steps of $Q_{\text{train}} = 32$ sized mini-batch training with each sample in a mini-batch being a generalized bag (or bag for the single bag baselines). We point out that we only perform hyper-parameter sweep for previous works; not for our training loop.

Appendix A provides additional details of these experiments.

Table 4: SDP [S] vs iid $N(0, 1)$ [R] wts vs Single Bags [U] for LIN(ℓ_2^2, _)

dataset	#	S	R	U
Ionosphere	I	87.48 ± 5.0	88.67 ± 4.2	$\mathbf{89.11} \pm 4.1$
	II	$\mathbf{85.42} \pm 5.4$	77.17 ± 5.6	77.86 ± 5.9
	III	$\mathbf{87.62} \pm 4.5$	78.42 ± 4.5	78.64 ± 5.4
	IV	$\mathbf{81.46} \pm 9.1$	76.06 ± 6.6	77.31 ± 6.6
	V	$\mathbf{86.31} \pm 5.8$	78.64 ± 6.3	82.13 ± 4.9
	VI	$\mathbf{87.79} \pm 4.8$	81.9 ± 5.7	83.27 ± 5.0
	VII	$\mathbf{87.12} \pm 3.8$	83.39 ± 4.9	85.79 ± 3.9
Australian	I	$\mathbf{92.31} \pm 2.4$	92.18 ± 2.4	92.24 ± 2.4
	II	$\mathbf{90.15} \pm 2.8$	86.67 ± 2.7	87.03 ± 2.5
	III	$\mathbf{90.88} \pm 2.6$	86.41 ± 2.7	86.46 ± 2.7
	IV	85.12 ± 3.2	84.67 ± 2.5	$\mathbf{85.68} \pm 2.5$
	V	$\mathbf{90.34} \pm 2.7$	87.43 ± 2.6	87.76 ± 2.4
	VI	$\mathbf{91.43} \pm 2.5$	84.92 ± 7.7	86.48 ± 5.3
	VII	$\mathbf{91.53} \pm 2.7$	88.47 ± 3.2	89.0 ± 3.0

Classifier Performance. Tables 2 and 3 show the performance of our method compared to previous techniques and a instance level Logistic regression model on two datasets. On the Ionosphere dataset (Table 2) our method LIN(ℓ_2^2, S) performs better than the previous methods in all except the scenario I where LMMCM performs the best. In the more complicated bag distribution scenarios LIN(ℓ_2^2, S) outperforms LMMCM by up to five percentage points e.g. in scenarios III, IV, VI and VII. Among the mean-map based methods LMM has the best performance, though lower than LIN(ℓ_2^2, S) in all scenarios.

On the Australian dataset, all algorithms perform relatively better. Our method LIN(ℓ_2^2, S) and LMMCM perform nearly the same (and the best overall) in all the scenarios except IV in which LMMCM is better.

We notice that all algorithms have higher standard deviations on the Ionosphere dataset than the Australian one (also observed in Patrini et al. (2014)). A notable pattern is that our method's performance across all scenarios in the Ionosphere dataset is relatively stable to distribution shifts. In contrast, performance of other methods varies significantly based on the scenario and bag distributions used.

Table 4 provides the comparison of (i) our method, (ii) taking random normal weights, and (iii) using only the original bags. We see that using SDP computed weights do provide significant benefit vs the random weights. Similarly using generalized bags over original bags also provides significant gains.

6.2 Real World Dataset: Criteo CTR

We provide an evaluation of our method on the Criteo Kaggle Display Advertising Challenge Dataset (Criteo (2014)) for click-through-rate (CTR) prediction. This consists of around 45 million rows, each an ad served with click/no-click label. It has 13 numeric features (N1 to N13), and 26 categorical features (C1 to C26) whose values are hashed. Typically, bag distributions would emerge from aggregation over some categorical features (e.g. product vertical, location, advertiser). Accordingly, we construct 5 bag distributions by aggregating over C14 and C7. Further, as a *simple* approximation method, we use the values of C15 as a proxy for the instances in the corresponding SDP-I. Adhering to practical privacy constraints, bags smaller than size 50 are discarded. Additionally, bags larger than 2500 are discarded for faster convergence. Details of these preprocessing and approximation steps are deferred to Appendix B.

We use a 5-fold train-test split with the bag distributions generated on each training set. The implementations of the previous methods (except for (KL-div, U) evaluated on UCI datasets (Tables 2, 3) do not scale to the Criteo dataset size and are omitted. We train a similar (as in Sec. 6.1) linear classifier (with sigmoid activation) using (i) single bags ℓ_2^2-loss baseline (ℓ_2^2, U), (ii) single bag KL-divergence loss baseline (KL-div, U), (iii) our generalized bags with SDP computed weights (ℓ_2^2, S), and (iv) our method with ℓ_1-loss (ℓ_1, S). A mini-batch has 8 samples of 5-bag tuples, each with one bag from the 5 bag distributions. The AUC scores on the test set in the last 10 training epochs (out of a total of 20) are considered. The results are included in Table 5. Our method performs the best overall with a statistically significant AUC gap over the baselines.

Table 5: AUC for *Criteo* (%). Instance level Linear classifier (log. reg) AUC of 76.25 ± 0.02.

LIN (ℓ_2^2, U)	LIN (KL-div, U)	LIN (ℓ_2^2, S) Our Method	LIN (ℓ_1, S)
72.22 ± 0.20	71.63 ± 0.20	$\mathbf{72.65} \pm 0.17$	72.52 ± 0.14

6.3 Real World Dataset: MovieLens 20M

Along similar lines as the Criteo CTR experiments, we also provide an evaluation on the MovieLens-20m (Harper and Konstan (2016); Movielens-20M) dataset for movie good/bad rating prediction. Firstly, the movie ratings are reduced to binary $\{0, 1\}$ using a natural partition of the rating scale. The dataset consists of around 19.8 million timestamped ratings of 10370 movies, with movies rated multiple times by different users. Each of these movies further has an associated vector of of 1128 genome tag scores with each score measuring the relevance of that tag to the movie's reviews. It is reasonable to assume that this genome tag scores vector captures the quality of the movie, and correlates with the ratings it receives, albeit from different users. Of course, movies may receive a mixture of good and bad ratings making this a noisy classification dataset, with an instance consisting of a timestamped rating with the feature vector of the genome tag scores for the movie.

We created natural bags as follows: For a given date, the instances timestamped with that date are grouped into 5 bags based on `timestamp mod 5`. These bags are then partitioned into 12 bag distributions, one for each month of the year. In a manner similar to the previous subsection, we use the movie genres strings to create proxy instances to formulate SDP-I for computing the weight distribution. Bags smaller than size 50 are discarded and larger than 2500 are discarded. Additional details are deferred to Appendix C.

Using a 5-fold train-test split with the bag distributions generated on each training set, we train a 2-layer perceptron (2LP) classifier – with a dense 64-node hidden layer with relu activations and sigmoid activated output – using the same methods as in Sec. 6.2. A mini-batch has 4 samples of 12-bag tuples, each with one bag from the 12 bag distributions. The AUC scores on the test set in the last 10 training epochs (out of a total of 20) are considered. The results included in Table 6 show that our method performs the best overall with a statistically significant AUC gap over the baselines. The scores are low for all the methods due to this being a noisy classification dataset, the instance level AUC itself being 71.79%.

Table 6: AUC for *MovieLens20m* (%). Instance level 2-layer perceptron (log. reg) AUC of 71.79 ± 0.03.

2LP (ℓ_2^2, U)	2LP (KL-div, U)	2LP (ℓ_2^2, S) Our Method	2LP (ℓ_1, S)
62.62 ± 0.35	63.53 ± 0.25	$\mathbf{64.35} \pm 0.47$	64.03 ± 1.03

Experimental Code. The code used for the empirical studies included in this paper is available at `https://github.com/google-research/google-research/tree/master/On_Combining_Bags_to_Better_Learn_from_Label_Proportions`.

7 CONCLUSION

We approach the LLP problem from a less studied direction - distribution of bags. Since bags are predetermined, they may not be amenable to an optimal LLP solution. We propose a method to linearly combine bags from different distributions to form *Generalized Bags* that satisfy certain isotropicity conditions. We prove bounds on the instance level accuracy of an LLP model trained on such generalized bags, *independent* of how the labels are distributed. Our experiments also demonstrate the applicability and robustness of our methods to various bag distribution scenarios on pseudo-synthetic and real-world datasets. We showed empirically that even a simple approximation methodology of the SDP computation provides a statistically significant gain over baselines on large datasets. A future direction of research is to design more sophisticated and robust approximation methods that can help further improve the performance of the classifier.

References

UCI Ionosphere. `https://archive.ics.uci.edu/ml/datasets/ionosphere`.

Statlog (Australian Credit Approval) data set. `https://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)`.

Ehsan Mohammady Ardehaly and Aron Culotta. Co-training for demographic classification using deep learning from label proportions. In *ICDM*, pages 1017–1024, 2017.

Gerda Bortsova, Florian Dubost, Silas N. Ørting, Ioannis Katramados, Laurens Hogeweg, Laura H. Thomsen, Mathilde M. W. Wille, and Marleen de Bruijne. Deep learning from label proportions for emphysema quantification. In *Medical Image Computing and Computer Assisted Intervention - MICCAI*, volume 11071 of *Lecture Notes in Computer Science*, pages 768–776. Springer, 2018.

Lei Chen, Zheng Huang, and Raghu Ramakrishnan. Cost-based labeling of groups of mass spectra. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 167–178, 2004.

Shuo Chen, Bin Liu, Mingjie Qian, and Changshui Zhang. Kernel k-means based framework for aggregate outputs classification. In Yücel Saygin, Jeffrey Xu Yu, Hillol Kargupta, Wei Wang, Sanjay Ranka, Philip S. Yu, and Xindong Wu, editors, *ICDM*, pages 356–361, 2009.

Criteo. Kaggle display advertising challenge dataset, 2014. URL `http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset/`.

Nando de Freitas and Hendrik Kück. Learning about individuals from group statistics. In *UAI*, pages 332–339, 2005.

Lucio Mwinmaarong Dery, Benjamin Nachman, Francesco Rubbo, and Ariel Schwartzman. Weakly supervised classification in high energy physics. *Journal of High Energy Physics*, 2017(5):1–11, 2017.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1–2): 31–71, January 1997. ISSN 0004-3702. doi: 10.1016/S0004-3702(96)00034-3. URL `https://doi.org/10.1016/S0004-3702(96)00034-3`.

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

Gabriel Dulac-Arnold, Neil Zeghidour, Marco Cuturi, Lucas Beyer, and Jean-Philippe Vert. Deep multi-class learning from label proportions. *CoRR*, abs/1905.12909, 2019. URL `http://arxiv.org/abs/1905.12909`.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, 2016. doi: 10.1145/2827872. URL `https://doi.org/10.1145/2827872`.

Jerónimo Hernández-González, Iñaki Inza, and José Antonio Lozano. Learning bayesian network classifiers from label proportions. *Pattern Recognit.*, 46(12): 3425–3440, 2013.

Dimitrios Kotzias, Misha Denil, Nando de Freitas, and Padhraic Smyth. From group to individual labels using deep features. In *SIGKDD*, pages 597–606, 2015.

Jiabin Liu, Bo Wang, Zhiquan Qi, Yingjie Tian, and Yong Shi. Learning from label proportions with generative adversarial networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *NeurIPS*, pages 7167–7177, 2019.

Movielens-20M. Movielens 20m dataset. URL `https://grouplens.org/datasets/movielens/20m/`.

David R. Musicant, Janara M. Christensen, and Jamie F. Olson. Supervised learning by training on aggregate outputs. In *ICDM*, pages 252–261. IEEE Computer Society, 2007.

Giorgio Patrini, Richard Nock, Tibério S. Caetano, and Paul Rivera. (almost) no label no cry. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, pages 190–198, 2014.

Novi Quadrianto, Alexander J. Smola, Tibério S. Caetano, and Quoc V. Le. Estimating labels from label proportions. *J. Mach. Learn. Res.*, 10:2349–2374, 2009.

Stefan Rueping. Svm classifier estimation from group probabilities. In *ICML*, 2010.

Stefan Rüping. SVM classifier estimation from group probabilities. In Johannes Fürnkranz and Thorsten Joachims, editors, *ICML*, pages 911–918, 2010.

Clayton Scott and Jianxin Zhang. Learning from label proportions: A mutual contamination framework. In *NeurIPS*, 2020.

Yong Shi, Jiabin Liu, Bo Wang, Zhiquan Qi, and Yingjie Tian. Deep learning from label proportions with labeled samples. *Neural Networks*, 128:73–81, 2020.

Marco Stolpe and Katharina Morik. Learning from label proportions by optimizing cluster model selection. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *ECML PKDD Proceedings, Part III*, volume 6913, pages 349–364. Springer, 2011.

R. Vershynin. How close is the sample covariance matrix to the actual covariance matrix? *J. Theor. Probab.*, 25:655—-686, 2012a.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing*, pages 210–268. Cambridge University Press, 2012b. URL `https://arxiv.org/pdf/1011.3027.pdf`.

Janusz Wojtusiak, Katherine Irvin, Aybike Birerdinc, and Ancha V Baranova. Using published medical results and non-homogenous data in rule learning. In *2011 10th International Conference on Machine Learning and Applications and Workshops*, volume 2, pages 84–89. IEEE, 2011.

Felix X. Yu, Dong Liu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. $\propto$SVM for learning with label proportions. In *ICML*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 504–512, 2013.

Felix X. Yu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. On learning with label proportions. *CoRR*, abs/1402.5902, 2014. URL `http://arxiv.org/abs/1402.5902`.

Supplementary Material: On Combining Bags to Better Learn from Label Proportions

A ADDITIONAL DETAILS FOR PSEUDO-SYNTHETIC DATASET EXPERIMENTS

A.1 Hyperparameter Tuning for Comparative Methods

As mentioned in Section 6.1, we compare against the following mean-map based methods: MM of Quadrianto et al. (2009), LMM ($v^{G,s}$), AMM(MM) and AMM (LMM (v^{GS})) (`min` versions) of Patrini et al. (2014). The implementation of all these (and for the Logistic Regression oracle baseline) is adapted from the the code[1] made publicly available by Patrini et al. (2014). We execute all these baselines for all the hyperparameter settings used in Patrini et al. (2014). In particular, (i) for logistic regression: $\lambda \in \{0, 1, 10, 100\}$, (ii) for MM: $\lambda \in \{0, 1, 10, 100\}$, (iii) for LMM ($v^{G,s}$):$\lambda \in \{0, 1, 10, 100\}, \gamma \in \{0.01, 0.1, 1\}, \sigma \in \{0.25, 0.5, 1\}$, (iv) for AMM(MM): $\lambda \in \{0, 1, 10, 100\}$, and (v) for AMM (LMM ($v^{GS}$)): $\lambda \in \{0, 1, 10, 100\}, \gamma \in \{0.01, 0.1, 1\}$.

The implementation of the LMMCM method was taken from the code[2] released by Scott and Zhang (2020). The method was executed on $\lambda \in \{1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. The parameter setting 10^{-5} resulted in instability and time-outs on the bag distribution scenarios in our experiments. We note that the LMMCM method requires that the number of bags be even which the experiments of Scott and Zhang (2020) ensured. In our experiments that may not be true, in which case one randomly chosen bag is dropped.

As mentioned in Section 6.1, for each method above, for each dataset and scenario the score taken was that corresponding to parameter setting with the highest average AUC across the 5 times 5 fold-split runs.

A.2 SDP Solutions Used

Here we describe solutions to the SDP in Find-wt-dist (Alg. 1) for the bag distribution scenarios I-VII. For each scenario ν = I,..., VII the solution $\mathbf{W}$ is a $d_\nu \times d_\nu$ matrix where d_ν is the total number of bag distributions in scenario ν. Except for scenario VI, the SDP is feasible – with scenarios II and III admitting the same solution. The resultant solutions to the SDP $\mathbf{W}_\nu$ are given in Table A1.

For the bag distributions in VI, the SDP however is infeasible. We use a simple slack variables method to relax the constraints of the SDP as given in Fig. 2. The number of slack variables in full generality is quadratic in the number of instances in the dataset. However, leveraging the independence in the sampling of instances in the bags of our distributions we solve the relaxed SDP on clusters of two instances each. Minimizing the slack variable sum (weighted by $\lambda = 10^2$) we obtain a rank-4 matrix $\mathbf{W}$ as a solution to the relaxed SDP, also listed in Table A1. The weights in the solution are ordered corresponding to bag distributions per cluster i, where for each cluster the heavier (by size) bag distribution is ordered earlier. For scenarios V-VII, the straddling bag distribution occur first, then the others for clusters $i = 1, 2, 3$.

A.3 Comparison with ℓ_1-loss

We also compare with the results given by optimizing the generalized bag level ℓ_1-loss instead of ℓ_2^2-loss. Table A2 shows the comparison of ℓ_2^2-loss using SDP computed weights vs ℓ_1-loss with SDP [S] or random Gaussian [R] weights as well as using original bags [U] instead of generalized bags. We observe that on both the (ℓ_2^2,S) and (ℓ_1,S) methods perform the best overall, with the latter performing better better than the former on the Ionosphere dataset, while both perform similarly on the Australian dataset. This provides evidence for the efficacy of ℓ_1-loss optimization, as well as the performance benefit accrued by using the SDP calculated weights even for the ℓ_1-loss case.

Extending our work to more loss functions and finding a general framework to understand the interaction between loss functions and generalized bags is a very interesting problem that we leave for future work.

B EXPERIMENTS ON CRITEO CTR DATASET

We elaborate on the description of the experiments presented in Sec. 6.2.

[1]`https://github.com/giorgiop/almostnolabel`

[2]`https://github.com/Z-Jianxin/Learning-from-Label-Proportions-A-Mutual-Contamination-Framework`

Table A1: SDP solution matrices $\mathbf{W}$

Scenario	$\mathbf{W}$
I	$\begin{bmatrix} 3.125 & -3.125 \\ -3.125 & 3.125 \end{bmatrix}$
II, III	$\begin{bmatrix} 0.136 & 0. & 0. & -1.22 & 0. & 0. \\ 0. & 0.136, & 0. & 0. & -1.22 & 0. \\ 0. & 0. & 0.136 & 0. & 0. & -1.22 \\ -1.22 & 0. & 0. & 10.98 & 0. & 0. \\ 0. & -1.22 & 0. & 0. & 10.98 & 0. \\ 0. & 0. & -1.22 & 0. & 0. & 10.98 \end{bmatrix}$
IV	$\begin{bmatrix} 2.131 & 0. & 0. & -2.131 & 0. & 0. \\ 0. & 2.131 & 0. & 0. & -2.131 & 0. \\ 0. & 0. & 2.131 & 0. & 0. & -2.131 \\ -2.131 & 0. & 0. & 2.131 & 0. & 0. \\ 0. & -2.131 & 0. & 0. & 2.131 & 0. \\ 0. & 0. & -2.131 & 0. & 0. & 2.131 \end{bmatrix}$
V	$\begin{bmatrix} 5.36 & -1.79 & -1.79 & -1.79 \\ -1.79 & 0.596 & 0.596 & 0.596 \\ -1.79 & 0.596 & 0.596 & 0.596 \\ -1.79 & 0.596 & 0.596 & 0.596 \end{bmatrix}$
VI	$\begin{bmatrix} 0.368 & -1.408 & -1.471 & -1.471 \\ -1.408 & 5.39 & 5.631 & 5.631 \\ -1.471 & 5.631 & 5.882 & 5.882 \\ -1.471 & 5.631 & 5.882 & 17.921 \end{bmatrix}$
VII	$\begin{bmatrix} 3.13 & -1.04 & -3.125 & 1.3e-04 & 3.13 \\ -1.04 & 0.6 & 1.04 & -7.4 & -1.8 \\ -3.125 & 1.04 & 3.125 & -1.7e-04 & -3.13 \\ 1.3e-04 & -0.74 & -1.7e-04 & 2.23 & 2.23 \\ 3.13 & -1.8 & -3.13 & 2.23 & 5.36 \end{bmatrix}$

Table A2: (ℓ_2^2,S) vs (ℓ_1,[S,R,U])

		Ionosphere		
#	(ℓ_2^2,S)	(ℓ_1,S)	(ℓ_1,R)	(ℓ_1,U)
I	87.48 ± 5.0	88.55 ± 4.3	88.94 ± 4.1	89.21 ± 4.0
II	85.42 ± 5.4	86.9 ± 4.6	79.63 ± 5.3	79.33 ± 5.6
III	87.62 ± 4.5	88.06 ± 3.7	80.43 ± 4.4	80.01 ± 5.1
IV	81.46 ± 9.1	84.2 ± 7.0	78.61 ± 5.8	78.78 ± 6.1
V	86.31 ± 5.8	87.44 ± 4.9	80.92 ± 5.5	83.19 ± 4.6
VI	87.79 ± 4.8	88.4 ± 4.1	83.48 ± 4.8	84.07 ± 4.6
VII	87.12 ± 3.8	87.87 ± 3.6	84.83 ± 4.3	86.39 ± 3.7
		Australian		
#	(ℓ_2^2,S)	(ℓ_1,S)	(ℓ_1,R)	(ℓ_1,U)
I	92.31 ± 2.4	92.22 ± 2.4	92.18 ± 2.4	92.23 ± 2.4
II	90.15 ± 2.8	90.1 ± 2.8	87.68 ± 2.6	87.56 ± 2.5
III	90.88 ± 2.6	90.66 ± 2.5	87.41 ± 2.5	87.05 ± 2.6
IV	85.12 ± 3.2	86.4 ± 2.6	85.76 ± 2.4	86.28 ± 2.4
V	90.34 ± 2.7	90.32 ± 2.5	88.3 ± 2.5	88.23 ± 2.4
VI	91.43 ± 2.5	91.19 ± 2.3	86.27 ± 6.4	87.12 ± 4.8
VII	91.53 ± 2.7	91.42 ± 2.6	89.17 ± 3.0	89.34 ± 3.0

Feature Preprocessing. The features of the dataset are first preprocessed as follows. The numerical feature values are categorized into around 40 buckets – of exponentially increasing values – labeled by integers starting from 0. The raw categorical feature values are given as hashes, and for each such feature the distinct values are converted into integers starting from 0. The missing values in each feature are assigned to be integer means of the values of that feature.

Bag Distributions. We choose two categorical features C14 and C7 to create the bag distributions as follows. For each distinct pairs of values of C14 and C7 there is a bag consisting of all instances with those two values in C14 and C7 respectively. Thus, for each distinct value of C14 we have a collection (distribution) of bags given by the different co-occurring values of C7. However, the number of bags for each of the 26 distinct values of C14 vary significantly. To make sure we have roughly equal number of bags per distribution we partition the values of C14 into 5 groups and for each group we take the union of the corresponding collections as a bag distribution. Thus, we obtain 5 different bag distributions. We only consider bags of sizes at least $t_{\text{low}} = 50$ and at most $t_{\text{high}} = 2500$. The bag distributions are computed for each train-split (of the 5-fold split) separately. There are approximately 36,100 training bags for each of the train-splits, partitioned roughly evenly among the 5 bag distributions.

Approximate SDP Solution. (Table A3) The number of constraints of SDP-I in Fig.2 makes it prohibitive to formulate and solve exactly at the scale of the this dataset. For our experiment we do a crude approximation as follows. We select C15, which has around 15,000 distinct values, and partition these values into 50 buckets have similar similar frequency counts. We use the new C15_bucketized feature as a proxy for instances and formulate an approximate version of SDP-I. Note that the bags may now be multisets in terms of the values of C15_bucketized. Thus, we compute for the 5 bag distributions, the 50×50 second moment matrix $\mathbf{Q}^{(i)}$ as well as the 50-dimensional mean-vectors $\boldsymbol{\mu}^{(i)}$ $(i = 1, \ldots, 5)$ to correctly formulate an extension of SDP-I for multiset bags. The SDP is over a 5×5 psd matrix, and has $\binom{50}{2} + 50 = 1275$ constraints.

The SDP is, due to these real-data constraints, not

Table A3: Approx. SDP solution normalized matrix **W** for Criteo Kaggle dataset experiments

$$\begin{bmatrix} 0.26 & -0.271 & -0.073 & -0.106 & 0.164 \\ -0.271 & 0.305 & 0.142 & 0.122 & -0.209 \\ -0.073 & 0.142 & 0.217 & 0.062 & -0.16 \\ -0.1061 & 0.122 & 0.062 & 0.0487 & -0.086 \\ 0.164 & -0.209 & -0.160 & -0.086 & 0.169 \end{bmatrix}$$

exactly feasible and is solved approximately using slack variables for the constraints as given in (5), with $\lambda = 101$ and $\lambda' = 1$. We formulate and solve this SDP for the entire unsplit dataset using higher bag size threshold values $t_{\text{low}} = 63$ and at most $t_{\text{high}} = 3125$ and use the normalized solution (presented in Table A3) in the training loops (as described in Sec. 5.2) for each of the 5 train/test splits.

Model Training. As mentioned above, the feature vectors values preprocessed to be integers. We use a linear classifier with sigmoid activation which takes as input one-hot encodings of the feature vectors. We omit those features which have more than 10,000 distinct values. The total number of the input features (expanded as one-hot encodings) to the classifier is 20179.

The model is trained on the training data aggregated into bags as above for our generalized bags methods and the single bag baselines as presented in Sec. 6.2. In the bag training loop, each mini-batch step has 8 samples of 5-tuples of bags, one bag each sampled uniformly from the 5 bag distributions. The single bag methods consider the mini-batch as consisting of 8×5 = 40 training bags. Our generalized bags methods generate 25 generalized bags for each of the 5-tuples, by independently sampling multiple weight vectors uniformly using the Gaussian covariance matrix **W** given by the SDP.

The model is also trained on the row-level training data for the linear (log. reg.) classifier using binary cross-entropy loss. Here, the training splits have around 36.5 million labeled instances.

The test splits used for evaluating the performance for all the methods have approximately 9 million labeled instances. We use the Adam optimizer with 10^{-3} learning rate, and for each split take AUC scores on the test set for the last 10 (out of a total of 20) training epochs.

C EXPERIMENTS ON MOVIELENS 20M

As mentioned in Sec. 6.3 the genome tag score feature vectors are available per movie, and therefore no additional feature preprocessing is required. The movie ratings are reduced to binary by letting a rating of 4 and above be 1 and 0 otherwise.

The 12 bag distributions (one for each month) as described in 6.3 are computed for each train-split (of the 5-fold split) separately. There are approximately 33,000 training bags for each of the train-splits, partitioned roughly evenly among the 12 bag distributions.

Approximate SDP Solution. (Table A4) Each movie has a genres string and we use these to construct proxy instances for solving the SDP approximately as follows. To even out the distribution of the genres strings so that we obtain 50 representative proxies, we first randomly *subdivide* the genres strings - for a movie with a genres string g, we assign a rating instance of the movie the subdivided genres string (g, r) where r is an independent uniform random integer in $\{1, 2, 3, 4, 5\}$. These subdivided genres strings are then partitioned into 50 buckets of roughly equal sizes to obtain the 50 proxy instances. As before, for each of the 12 bag distributions, we compute the the 50×50 second moment matrix as well as the 50-dimensional mean-vector. The SDP is over a 12×12 psd matrix, and has $\binom{50}{2} + 50 = 1275$ constraints. The use of slack variables (see (5)) with $\lambda = 101$ and $\lambda' = 1$ to ensure feasibility of the SDP is as before. To ensure that the SDP solution is not concentrated on just a few bag distributions, we also add a constraint to the psd matrix ensuring that no diagonal entry is more than twice the minimum diagonal entry.

Model Training. The 2-layer perceptron (2LP) described in Sec. 6.3 is trained on the training data aggregated into bags for our generalized bags methods and the single bag baselines. In the bag training loop, each mini-batch step has 4 samples of 12-tuples of bags, one bag each sampled uniformly from the 12 bag distributions. The single bag methods consider the mini-batch as consisting of 4×12 = 48 training bags. Our generalized bags methods generate 60 generalized bags for each of the 12-tuples, by independently sampling multiple weight vectors uniformly using the Gaussian covariance matrix **W** given by the SDP.

The model is also trained on the row-level training data for the 2LP classifier using binary cross-entropy loss. Here, the training splits have around 15.84 million labeled instances.

The test splits used for evaluating the performance for all the methods have approximately 3.96 million labeled instances. We use the Adam optimizer with 10^{-3} learning rate, and for each split take AUC scores on the test set for the last 10 (out of a total of 20) training epochs.

Table A4: Approx. SDP solution normalized matrix $\mathbf{W}$ for MovieLens 20M dataset experiments

$$\begin{bmatrix}
0.1170 & 0.1169 & 0.1169 & -0.1151 & 0.0862 & -0.0826 & -0.0821 & -0.0824 & 0.1169 & -0.0825 & 0.0812 & -0.0240 \\
0.1169 & 0.1170 & 0.1169 & -0.1155 & 0.0860 & -0.0826 & -0.0822 & -0.0825 & 0.1169 & -0.0826 & 0.0809 & -0.0226 \\
0.1169 & 0.1169 & 0.1170 & -0.1149 & 0.0863 & -0.0825 & -0.0819 & -0.0823 & 0.1168 & -0.0824 & 0.0814 & -0.0250 \\
-0.1151 & -0.1155 & -0.1149 & 0.1170 & -0.0830 & 0.0821 & 0.0826 & 0.0824 & -0.1157 & 0.0822 & -0.0772 & 0.0098 \\
0.0862 & 0.0860 & 0.0863 & -0.0830 & 0.0646 & -0.0604 & -0.0595 & -0.0600 & 0.0858 & -0.0602 & 0.0613 & -0.0250 \\
-0.0826 & -0.0826 & -0.0825 & 0.0821 & -0.0604 & 0.0586 & 0.0583 & 0.0584 & -0.0827 & 0.0585 & -0.0567 & 0.0138 \\
-0.0821 & -0.0822 & -0.0819 & 0.0826 & -0.0595 & 0.0583 & 0.0586 & 0.0584 & -0.0823 & 0.0584 & -0.0556 & 0.0100 \\
-0.0824 & -0.0825 & -0.0823 & 0.0824 & -0.0600 & 0.0584 & 0.0584 & 0.0586 & -0.0825 & 0.0585 & -0.0562 & 0.0120 \\
0.1169 & 0.1169 & 0.1168 & -0.1157 & 0.0858 & -0.0827 & -0.0823 & -0.0825 & 0.1170 & -0.0826 & 0.0807 & -0.0217 \\
-0.0825 & -0.0826 & -0.0824 & 0.0822 & -0.0602 & 0.0585 & 0.0584 & 0.0585 & -0.0826 & 0.0586 & -0.0565 & 0.0131 \\
0.0812 & 0.0809 & 0.0814 & -0.0772 & 0.0613 & -0.0567 & -0.0556 & -0.0562 & 0.0807 & -0.0565 & 0.0586 & -0.0274 \\
-0.0240 & -0.0226 & -0.0250 & 0.0098 & -0.0250 & 0.0138 & 0.0100 & 0.0120 & -0.0217 & 0.0131 & -0.0274 & 0.0586
\end{bmatrix}$$