Domain-Agnostic Contrastive Representations for Learning from Label Proportions

Jay Nandy
Google Research
Bangalore, India
jnandy@google.com

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

Prateek Jain
Google Research
Bangalore, India
prajain@google.com

Jatin Chauhan
Google Research
Bangalore, India
chauhanjatin@google.com

Balaraman Ravindran
Robert Bosch Centre for Data Science and AI, Indian Institute of Technology, Madras
Chennai, India
ravi@cse.iitm.ac.in

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

ABSTRACT

We study the weak supervision learning problem of *Learning from Label Proportions (LLP)* where the goal is to learn an instance-level classifier using proportions of various class labels in a bag – a collection of input instances that often can be highly correlated. While representation learning for weakly-supervised tasks is found to be effective, they often require domain knowledge. To the best of our knowledge, representation learning for tabular data (unstructured data containing both continuous and categorical features) are not studied. In this paper, we propose to learn diverse representations of instances within the same bags to effectively utilize the weak bag-level supervision. We propose a domain agnostic LLP method, called *"Self Contrastive Representation Learning for LLP" (SelfCLR-LLP)* that incorporates a novel self–contrastive function as an auxiliary loss to learn representations on tabular data for LLP. We show that diverse representations for instances within the same bags aid efficient usage of the weak bag-level LLP supervision. We evaluate the proposed method through extensive experiments on real-world LLP datasets from e-commerce applications to demonstrate the effectiveness of our proposed SelfCLR-LLP.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**.

KEYWORDS

Learning from Label Proportions, Weak Supervision, Contrastive Learning, Click-through-rate prediction, Item Recommendation

ACM Reference Format:
Jay Nandy, Rishi Saket, Prateek Jain, Jatin Chauhan, Balaraman Ravindran, and Aravindan Raghuveer. 2022. Domain-Agnostic Contrastive Representations for Learning from Label Proportions. In *Proceedings of the 31st ACM Int'l Conference on Information and Knowledge Management (CIKM '22), Oct. 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557293

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10.
https://doi.org/10.1145/3511808.3557293

	Movie Genres	Movie ID	Movie Title	User ID	Watch Time (min)	User Rating
Sample 1	Comedy	7367	Ladykillers, The (2004)	58198	100	3.5
Sample 2	Sci-Fi	109487	Interstellar (2014)	149681	150	4.0
Sample 3	Romance	2629	Love Letter, The (1999)	129	90	2.5

Table 1: Example of a tabular dataset with both continuous and categorical features. The neighborhood to perturb these instances (rows) without changing labels ('User Rating') is not well-defined.

1 INTRODUCTION

Learning from label-proportions (LLP) is a weak learning task, where training data is provided in "bags" along with the proportion of each class. The goal is to learn a classification model to predict the class-labels of individual instances [10, 32]. The LLP formulation has *two key properties*, making it attractive for web-scale and e-commerce applications: (a) Privacy Preservation and (b) Weak Supervision.

a) Privacy Preservation: Usage of user data in applications like *online advertising* is being heavily regulated by governments and major advertising platforms are moving to restricted tracking of user events [34]. In LLP formulation, the labels of individual records are not exposed and hence is a direct fit for learning from privacy-sensitive data that need k-anonymity. Recent works [34, 39, 41] show how LLP can be used for training models in privacy-sensitive settings. Another aspect to note here is that most practical applications in the web domain extensively use tabular data, containing both numerical and categorical features (Table 1.)

b) Weak Supervision: Clean supervised data are not always easy to obtain for data-hungry neural models. For instance, in power-constrained environments like mobile phone clients, collecting very precise user interaction data for tasks like app install prediction will add significant energy consumption overhead [3]. Hence, in such situations, the instrumentation is usually at the level of hourly/daily aggregates. Also, research directions like Green AI advocate for doing more with fewer data [40]. For these two reasons, weakly supervised methods are gaining increasing focus in the research community as a more sustainable and practical route to learning large-scale models that are common in e-commerce applications.

An effective approach of learning from weak supervisions is to incorporate unsupervised representation learning techniques for uncovering patterns in the data to aid the task at hand [5, 6, 28]. However, prior representation learning methods typically require domain-specific knowledge for strong augmentations or masking strategies [15, 46, 52, 53]. For instance, in computer vision, different perturbations (e.g., cropping, rotation) are applied to produce samples with the guarantee that they will not change the semantics for the task at hand - such as classification. However, such methods do not readily extend to tabular data, containing both continuous and categorical features. In these cases, it is not trivial to define a neighborhood for perturbations without changing the class-labels (see Table 1). Hence, we cannot apply the same perturbations as vision datasets as discussed above. To the best of our knowledge, none of the existing methods focus on learning the representations for tabular datasets, containing continuous and categorical features. In this paper, we propose a new representation learning based method, called **SelfCLR-LLP** that works on tabular data without any assumption on their structure. We show that SelfCLR-LLP aids LLP by introducing the right level of diversity in the learned representation, paving the way for a more effective classifier.

We make the following contributions in this paper:

1. We posit that LLP, a weak supervision learning problem, would benefit from an appropriate representation learning method. To the best of our knowledge, this direction of research has not been pursued in the LLP community. We systematically analyze the shortcomings of the existing representation learning method when applied to tabular data and formulate the LLP-specific representation learning problem accordingly. (Section 2, 3)

2. We propose a novel domain agnostic representation learning based method for LLP, called SelfCLR-LLP. Our proposed method makes no assumptions about the structure of the data. Also, it does not require an explicit distance metric between rows of tabular data. To the best of our knowledge, we have not seen prior work to learn representations on tabular data that contain both continuous and categorical features. We propose a simple yet effective auxiliary contrastive loss to produce diverse representations to utilize the weak supervision effectively for LLP tasks. (Section 4)

3. We conduct extensive experiments on real-world e-commerce applications: (a) *click-through-rate (CTR) prediction* using Criteo-Kaggle [8] with $\approx$ 45 million instances, (b) *item recommendation* using MovieLens-1M with $\approx$ 0.8 million instances [18]. Existing works typically created the training bags by randomly sampling instances from training sets [2, 13, 30] without capturing realistic instance-level correlations within the bags. Here, we adopt a more involved method to produce more realistic bags. Our method achieves up to $\approx$ 2% and $\approx$ 4.3% higher AUROC scores compared to the state of art baseline DLLP [2, 13] on Criteo-Kaggle and MovieLens-1M respectively. It is important to note that even $\approx 0.2 - 0.5\%$ improvement can be very challenging on these datasets [17, 29, 42]. (Section 5)

2 BACKGROUND AND RELATED WORK

2.1 Learning from Label Proportions

Consider a K-class classification problem in the LLP setting where individual instance labels are unknown [10, 32]. Let $x \in \mathcal{X}$ denote the input features, and $y \in [K]$ are the corresponding class labels. Notably, the individual labels, y, remain unavailable for training. We consider a domain-agnostic framework where the feature space $\mathcal{X}$ can take both continuous and categorical features. The training data is a set of bags along with label proportions: $\{(B_i, \boldsymbol{p}_i)\}$ where $B_i = \{x_i^1, \cdots, x_i^{N_i}\}$ denotes the training bags. The label proportions, $\boldsymbol{p}_i = \{p_i^1, \cdots, p_i^K | \sum_k p_i^k = 1\}$ are K-dimensional vectors. p_i^k denotes the proportion of instances of bag B_i belonging to class k:

$$p_i^k = \frac{|\{x | x \in B_i, y(x) = k\}|}{|B_i|}. \tag{1}$$

where $y(x)$ denotes the *inaccessible* ground-truth class label corresponding to instance x, and $|S|$ denotes the cardinality of set S, i.e., $|\{x | x \in B_i, y = k\}|$ is the number of instances in LLP training bag B_i with class-label k. For our paper, we do not make any assumptions about the training bags. In particular, these bags may overlap or contain a different number of instances.

Given the LLP training bags along with the label proportions $\{(B_i, \boldsymbol{p}_i)\}$, our objective is to construct an *instance-level* classification model that correctly maps a test instance, x to their corresponding class-label $y(x)$ i.e., : $f_{\boldsymbol{\theta}} : x \rightarrow y(x)$; where, $f_{\boldsymbol{\theta}}$ denotes the classification model with learnable parameters $\boldsymbol{\theta}$.

LLP is a well-known problem in the machine learning community with various influential real-world applications. For example, census or medical databases are typically provided as label proportion due to privacy concerns [19, 36]. Several other applications include fraud detection [38], object recognition [10], video event detection [25], ice-water classification [27] etc.

Shallow Learning Models for LLP. Several shallow learning models were proposed in the pre-deep learning era. Musicant et al. [32] applied standard supervised classification models, including support vector machines (SVM), k-NN, and neural networks for LLP, while de Freitas and Kück [32] and later Hernández-González et al. [20] proposed MCMC-based probabilistic models. Several other shallow models were proposed using SVMs [38], *proportional* SVMs [50], using bag means with some generative assumptions [36, 37], or using the mutual contamination framework [41]. These shallow learning models were mainly proposed for binary classification settings. Notably, these methods do not scale for higher dimensional and larger datasets.

Deep LLP models. An influential line of research in deep LLP models is to learn a deep classifier by matching the given label proportions on the training data [2, 13, 32]. More precisely, these methods aim to minimize some distance/divergence function between the given label proportions, $\boldsymbol{p}_i$ and predicted proportions, $\tilde{\boldsymbol{p}}_i$. For a bag B_i and a deep model f with parameters $\boldsymbol{\theta}$, we obtain predicted proportions as: $\tilde{\boldsymbol{p}}_{\boldsymbol{i}} = \sum_i p(y|x_i^j, \theta)$ where $p(y|x_i^j, \theta)$ is the distribution over labels, predicted by f for a given *instance* x_i^j in B_i.

To this end, one can select standard ℓ_1 or ℓ_2^2 distance functions. Recently, Ardehaly and Culotta [2] and Dulac-Arnold et al. [13] proposed to minimize a modified version of cross-entropy loss between $\boldsymbol{p}_{\boldsymbol{i}}$ and $\tilde{\boldsymbol{p}}_{\boldsymbol{i}}$ for each bag B_i as follows:

$$\mathcal{L}_{prop}(\boldsymbol{p}_i, \tilde{\boldsymbol{p}}_i) := -\boldsymbol{p}_{\boldsymbol{i}}^T \log \tilde{\boldsymbol{p}}_i \tag{2}$$

Dulac-Arnold et al. [13] further used the concept of pseudo-labeling [26], where an estimate of the individual labels within each bag is jointly optimized with the model during training in a

multi-class LLP problem. Liu et al. [31] revisited the pseudo labeling problem for LLP by proposing a two-stage scheme that alternately updates the network parameters and the pseudo-labels. However, as the bag sizes increase to provide weaker instance-level supervisions, the performance of these models degrades rapidly with increasing bag sizes [30, 44].

Auxiliary loss for Deep LLP models. Recently a line of work attempts to address these challenges of weak supervision for large bags by learning "natural" representations of data or augmenting the data. For example, Liu et al. [30] proposed an LLP-GAN framework by incorporating *generative adversarial networks* (GAN) [16]. It consists of a generator and a discriminator model where the generator produces instances to fool the discriminator network, learning better representations. The *instance-level* discriminator is trained in a multi-task fashion to distinguish the generated samples and minimize the bag proportion loss. Tsai & Lin [44] proposed LLP-VAT that incorporated an additional consistency regularizer to produce consistent representations when the input instances are perturbed adversarially [4, 45].

Limitations. Existing auxiliary objectives for LLP tasks work well for structured input domains, e.g., image, video, or speech datasets. In these domains, the generators/ discriminators are well-known deep networks to apply methods such as LLP-GAN. Similarly, we can easily define the distance metrics to apply techniques such as LLP-VAT. Unfortunately, we cannot readily incorporate these objectives for input domains with sensitive categorical or numerical attributes (as in Table 1).

2.2 Representation Learning

Several recent works proposed unsupervised representation learning techniques without labeled samples that can be used for complex downstream tasks, including classification. One may trivially extend such techniques to the LLP setting to train the classifier in a multi-task fashion incorporating representation learning loss along with the proportion loss (Eq. 2). However, representation learning techniques are typically applicable for structured datasets or require significant domain knowledge.

Existing representation learning techniques can be broadly categorized into two groups: *hand-crafted pretext tasks* and *contrastive learning-based frameworks*. The *hand-crafted pretext tasks* methods use pseudo-supervised learning by manipulating the original input training samples to learn representations. Example of such frameworks include relative patch prediction [11], solving jigsaw puzzles [33], colorization [51] and rotation prediction [7, 23]. Such techniques typically rely on a carefully crafted pretext, which is challenging to generalize across domains.

Contrastive learning based models augment the data and learn representations that are close to each other for positive pairs – a sample and its augmentation – and away for any other samples (i.e., negative pairs) [6]. These techniques achieve the current state-of-the-art performance for various domains with structured inputs e.g., computer vision [5, 6, 28], graphs [52, 53], text [15], tabular data [46] etc. Further, Khosla et al. [21] demonstrated that pre-training with contrastive learning improves classification performance even in supervised settings. A few recent works also introduced a contrastive learning framework to some web applications [35, 47, 48]. However, these models typically require a domain-specific augmentation. Hence, these techniques are not immediately applicable for domains where natural augmentation techniques are unavailable.

3 CHALLENGES

In this section, we discuss the key challenges in developing a domain agnostic representation for tabular data commonly used in LLP applications. First, we discuss why popular approaches like SimCLR will not work for tabular datasets. Next, we discuss the unique challenges of representation learning for LLP tasks.

3.1 Limitations of Traditional Contrastive Learning Methods

We analyze the *normalized temperature-scaled cross-entropy (NT-Xent) loss* for the well-known SimCLR method as an example [6]. Let us consider a mini-batch of N training instances. For each instance of the mini-batch, we produce positive pairs using stochastic data augmentation techniques to generate $2N$ training instances. The contrastive loss is then defined to attract the representations of positive pairs and repel the other instances as "negative pairs". Let (i, i') denote a positive pair where both $i, i' \in I \equiv \{1, \cdots, 2N\}$ Then, NT-Xent loss is defined as follows [6]:

$$\mathcal{L}_{NT-Xent} = \frac{1}{N} \sum_{i \in I} -\log \frac{\exp\left(\frac{z_i^T z_{i'}/\tau}{||z_i|| \cdot ||z_{i'}||}\right)}{\sum_{j \in I \setminus \{i\}} \exp\left(\frac{z_i^T z_j/\tau}{||z_i|| \cdot ||z_j||}\right)} \tag{3}$$

where z denotes the representations produced by the DNN for input, x and z^T is the transpose of z. τ is the temperature parameter. $\frac{z_i^T z_j}{||z_i|| \cdot ||z_j||}$ computes the dot-product similarity between representations z_i and z_j.

NT-Xent loss for the SimCLR method requires both positive and negative instance pairs corresponding to each training instance [6]. The performance of these models depends heavily on the quality of the stochastic data augmentation to produce positive pairs. The augmentation requires maintaining the semantic similarities between the generated positive pairs. However, it is not trivial for feature spaces with sensitive/immutable attributes, such as CTR or recommendation tasks [8, 18].

The SimCLR method uses randomly-sampled instances within a mini-batch as negative samples for a given instance. When the number of class labels is limited (binary in the extreme case), random sampling can lead to spurious negative pairs, leading to poor representations. Unlike NT-Xent loss, our proposed selfCLR-LLP method is a domain-agnostic technique that does not require any positive or negative pairs.

3.2 Representations for LLP

Our goal is to learn a deep-neural network (DNN) based instance-level classification model. Here, the main challenge is to learn the hidden instance labels from the weak supervision of training bags that may often contain highly correlated instances. Consequently, classification models often fail to distinguish correlated instances from the same training bags. Hence, as we receive larger training bags with higher instance-level uncertainties, incorporating an auxiliary loss can boost the classification performance significantly.

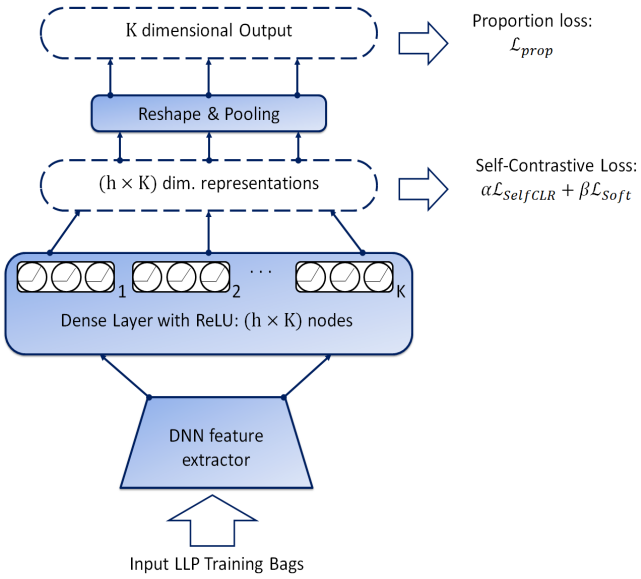


Figure 1: Schematic diagram of our proposed SelfCLR-LLP model.

In this paper, we argue that producing diverse representations for instances within the same bags to learn strong domain agnostic representations for LLP tasks. It allows the classifier to operate on each instance separately. We use a natural hypothesis of latent sub-classes where each class-label is divided among multiple soft latent sub-classes to propose our domain agnostic SelfCLR-LLP method. Intuitively, SelfCLR-LLP combines the above-mentioned hypothesis of latent sub-classes with our proposed variation of self-contrastive loss along with a specific regularization term to promote controlled diversity in representations within the same LLP bags.

4 PROPOSED METHODOLOGY

We now present our proposed SelfCLR-LLP method that incorporates a novel self-contrastive loss to introduce controllable diverse representations for instances within the same training bags, efficiently utilizing the weaker LLP supervisions.

Network architecture. Figure 1 presents the schematic diagram for our proposed SelfCLR-LLP method. We aim to learn at most h sub-classes per class. To this end, we append a dense layer with $(h \times K)$ nodes for a K-class classification problem. Here, each node denotes a latent feature corresponding to one output class. We take the $(h \times K)$-dimensional representations obtained from the dense layer to apply our proposed auxiliary self-contrastive loss. Finally, we obtain the class-specific K-class logit values by extracting the maximum values corresponding to each class from the $(h \times K)$-dimensional output. The bag label proportion loss, $\mathcal{L}_{prop}$ (Eq. 2) is applied on these logits during our multi-task training. We describe our proposed self-contrastive loss in the following:

Self-Contrastive Loss. Consider a mini-batch, $\mathcal{B}$ of training bags, consisting of correlated training instances. Let Z be the set of representations of all the feature-vectors contained in the collection of bags $\mathcal{B}$. The SelfCLR loss is defined as follows:

$$\mathcal{L}_{selfCLR}(Z) := \frac{1}{|Z|} \sum_{z \in Z} -\log \frac{\exp\Big(sim(z, z)\Big)}{\sum_{z' \in Z} \exp\Big(sim(z, z')\Big)} \tag{4}$$

where, $sim(z, z')$ is the similarity between the representations.

Analysis. We can rewrite $\mathcal{L}_{selfCLR}(Z)$ as:

$$\frac{1}{|Z|} \sum_{z \in Z} \log\Big(1 + \sum_{z' \in Z; z' \neq z} \exp\big[sim(z, z') - sim(z, z)\big]\Big) \tag{5}$$

Hence, min $\mathcal{L}_{selfCLR}(Z)$ is equivalent to min $\big[sim(z, z') - sim(z, z)\big]$. Intuitively, it minimizes the similarity between any pair of instances within the mini-batch.

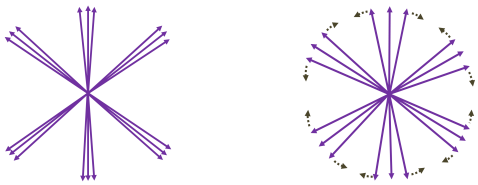
By choosing the similarity function as $sim(z, z') = z^T \cdot z' = ||z|| \cdot ||z'|| \cdot cos\ \phi$, we get $sim(z, z') - sim(z, z) = ||z||(||z'||cos\ \phi - ||z||)$; where ϕ denotes the angle between z and z'. Hence, the optimizer can minimize $\mathcal{L}_{selfCLR}(Z)$ either by (a) maximizing the angle between z and z', or (b) increasing the norms of z and z' (i.e., $||z||$ and $||z'||$) when $cos\ \phi < 1$.

However, we also optimize the network using the weakly supervised bag-level proportion loss (Eq. 2) in a multi-task fashion. Hence, arbitrarily increasing the norm, $||z||$ for different representations can blow up the training loss even when the classifier makes minor mistakes on the training dataset [12, 43]. Therefore, the optimizer minimizes the loss $\mathcal{L}_{selfCLR}(Z)$ in Eq. 4 only by maximizing the angle among the representations for training instances within the bags. Hence, it enables the classifier to operate on each training instance separately within the same LLP bags, leading to better discriminating representations for classification.

Algorithm 1 Training-step for SelfCLR-LLP method

Input: Set of LLP training bags with label proportions, $\{(B, \boldsymbol{p})\}$. Bags per mini-batch: m.
Output: Classification model, $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$.

▷ Sampled LLP training bags
$\mathcal{B} = \{(B_i, \boldsymbol{p}_i)\}_{i=1}^{m} \in \{(B, \boldsymbol{p})\}$
Obtain representations for instances in $\mathcal{B}$.
$Z = []$
for $x \in \{B_i\}_{i=1}^{m}$ **do**
  $z = f_{\boldsymbol{\theta}}(x)$
  $Z.append(z)$
end for
▷ *Compute auxiliary losses (Eq. 4 & Eq. 6)*
$\mathcal{L} \Leftarrow \alpha\mathcal{L}_{SelfCLR}(Z) + \beta\mathcal{L}_{soft}(Z)$
▷ *Compute proportion loss (Eq. 2)*
for $i = 1, \ldots, m$ **do**
  $\tilde{\boldsymbol{p}} = 0$
  for $z \in \{\ z := f_{\theta}(x), x \in B_i\}$ **do**
    $\tilde{\boldsymbol{p}} = \tilde{\boldsymbol{p}} + \text{softmax}(\ \max(\ \text{reshape}(\ z, [k, h]), axis = 1))$
  end for
  $\tilde{\boldsymbol{p}}_i \Leftarrow \tilde{\boldsymbol{p}}/|B_i|$
end for
$\mathcal{L} \Leftarrow \mathcal{L} + \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}_{prop}(\boldsymbol{p}_i, \tilde{\boldsymbol{p}}_i)$
Optimize model $f_{\boldsymbol{\theta}}$ by minimizing overall loss $\mathcal{L}$
Return $f_{\boldsymbol{\theta}}$



(a) without $\mathcal{L}_{soft}$ (b) with $\mathcal{L}_{soft}$

Figure 2: (a) $\mathcal{L}_{selfCLR}$ may produce multiple diverse cluster of representations without without uniformly spreading them for the same LLP training bags. (b) We incorporate an additional regularizer, $\mathcal{L}_{selfCLR}$ to produce softer representations to uniformly spread the representations.

We train a SelfCLR-LLP model in a multi-task fashion using proposed self-contrastive loss along with the weakly supervised proportion loss. Note that we introduce diversity to ensure that the DNN models can efficiently utilize the weaker bag-level supervisions on individual training instances. *Therefore, the resultant representations are not necessarily very diverse as random representations. Instead, we aim to learn appropriate discriminative representations for improving the overall classification performance.*

Advantages. SelfCLR-LLP eliminates the requirement of producing positive instance pairs using sophisticated data augmentation techniques, making it readily applicable for general LLP tasks. Furthermore, $\mathcal{L}_{selfCLR}(Z)$ maximizes diversity among all instances within an LLP training bag, irrespective of their class labels. Hence, it automatically eliminates the requirement of choosing negative samples. Notably, we do not normalize the representations or apply temperature scaling (as in Eq. 3). We experimentally observe that it degrades the performance for SelfCLR-LLP (see Table 6).

Additional Regularization. While incorporating $\mathcal{L}_{SelfCLR}$ already produces diverse representations, it may not enforce to uniformly increase the angle of different representations over the feature space. In particular, it may lead to multiple small clusters of representation with high diversity for correlated instances within the LLP training bags, as shown in Figure 2(a).

To this end, we propose an additional regularization term, $\mathcal{L}_{soft}$ for SelfCLR-LLP to produce softer representations, as follows:

$$\mathcal{L}_{soft}(Z) = \frac{1}{|Z|} \sum_{z \in Z} \max sigmoid(z) \tag{6}$$

where sigmoid activation is applied to bound the values of z.

$\mathcal{L}_{soft}$ minimizes the sigmoid of logit value corresponding to the predicted class. It increases the smaller logit values of different feature coordinates, producing *"softer" representations*. Hence, the similarities among the smaller cluster of representations increase and spread the representations more uniformly, as shown in Figure 2(b). Note that, as we keep on increasing the weight for $\mathcal{L}_{soft}$, it eventually leads to uniform values for feature coordinates, ignoring $\mathcal{L}_{selfCLR}$ and $\mathcal{L}_{prop}$. Hence, larger weights for $\mathcal{L}_{soft}$ degrade both diversity and overall classification performance. However, we experimentally show that by appropriately choosing the weight for such *smoothness* constraints typically significantly improves the performance of SelfCLR-LLP models.

Overall loss function. We train the DNN classifier in a multi-task fashion using our proposed auxiliary self-contrastive losses, along with the weakly-supervised proportion loss (Eq. 2) for an LLP task. For a given mini-batch $\{(B_i, p_i)\}_{i=1}^{m} =: \mathcal{B}$ of LLP training bags with output representations Z, we define the overall loss function as:

$$\mathcal{L}_{SelfCLR-LLP}(\mathcal{B}) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_{prop}(\boldsymbol{p}_i, \tilde{\boldsymbol{p}}_i) + \alpha \mathcal{L}_{SelfCLR}(Z) + \beta \mathcal{L}_{soft}(Z) \tag{7}$$

where α and β are user-chosen weight parameters and $\boldsymbol{p}_i$ and $\tilde{\boldsymbol{p}}_i$ represents the ground-truth and predicted label proportions for bag B_i in $\mathcal{B}$. We provide the complete training algorithm for our SelfCLR-LLP method in Algorithm 1.

5 EXPERIMENTS

We investigate the effectiveness of our proposed SelfCLR-LLP models on different LLP scenarios. We conduct experiments on datasets from two different real-world application: (a.) *click-through-rate (CTR) prediction* using Criteo-Kaggle dataset and (b.) *item recommendation* using MovieLens-1M dataset with sensitive categorical features. We train 5 different models for each method and report the average performance. We also apply *t-test* to verify the statistical significance of performance gains achieved by our models. Throughout our experiments, we provide thorough ablation studies for our method to report the performance with subtle changes in the data, and its dependence on different hyper-parameters.

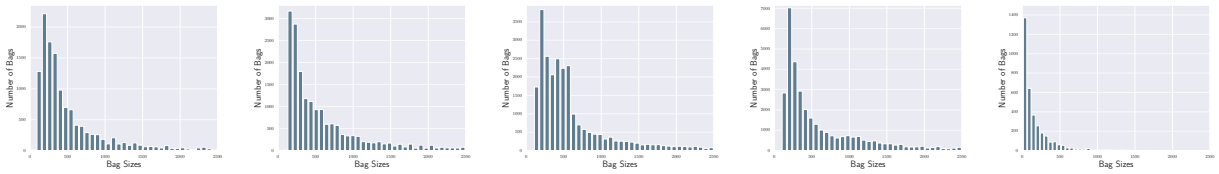
While LLP is a widely applicable problem with several applications, this research domain typically lacks datasets to effectively capture the real-world nuances. Most of the existing works typically create datasets using classification benchmarks by randomly combining the instances into bags [30, 31, 44]. However, these randomly created bags cannot capture the instance level correlations within the same bags for real-world applications. We instead present LLP datasets for CTR prediction and item recommendation using the combinations of categorical feature attributes for creating bags, closely simulating the real-world bag distributions.

We recall that the existing advanced methods for LLP [30, 44], require structured datasets or domain-specific modeling techniques (section 2.1). Hence, these models are not applicable for datasets such as Criteo-Kaggle and MovieLens-1M with sensitive categorical/ numerical features. Therefore, we compare with the existing DLLP method [2, 13] . It only uses the bag-level proportion loss function without focusing on learning specific representations. We choose the standard task-specific backbone DNN architecture and use the same setups for both DLLP and SelfCLR-LLP models.

5.1 Dataset preparation & Training details

5.1.1 ***Data sets.*** We first evaluate our SelfCLR-LLP models on datasets with categorical input features. Here, we use two benchmark datasets i.e., Criteo Kaggle [8] and Movielens-1M [18] for CTR prediction task and item recommendation respectively.

Crito-Kaggle dataset [8] contains $\approx$ 45 million records, each being a displayed ad, along with a binary label indicating whether it resulted in a click. Each sample contains 26 categorical and 13 numerical features. The feature names are anonymized. Also, the categorical features are anonymized by hashing. We rename the



(a) Criteo-12K (b) Criteo-17K (c) Criteo-24K (d) Criteo-33K (e) ML-3K

Figure 3: Visualizing the distributions of training bag-sizes (bag size versus no. of bags) for different LLP datasets.

categorical features to $C1$ to $C26$ and numerical features to $I1$ to $I13$. For each categorical feature, we remove the infrequent values that appear in less than the threshold of 10 rows and treat them as a single value as $< unknown >$. We also normalize the numerical values v using a transformation function as: $\log(v+1)$ (as in [17, 42]). We randomly split the dataset to select 75% as training samples, 20% as test dataset and remaining 5% as the validation set. We use the validation dataset only for supervised training and not for LLP.

MovieLens-1M [18] contains 739, 012 instances of users' ratings (integers 1 to 5) on different movies. We convert the dataset into a binary classification task (as in [17, 42]) by treating ratings > 3 as positive samples. We remove the instances with a rating of 3, and consider the remaining as negative samples. We randomly shuffle the dataset and use 80% as training and remaining 20% as test set.

5.1.2 *Creating natural LLP training bags.*

Criteo Kaggle [8]: Criteo Kaggle dataset contains anonymous features. We can create the training bags from the training split by grouping one or multiple categorical features with unique combinations. We observe a *long tail* behavior with a large number of bags containing only a few training instances. Hence, we combine bags of size ≤ 150. It leads to only larger training bags of size > 150, increasing the difficulty of the weakly-supervised LLP task. For each size i ($1 \leq i \leq 150$) we re-aggregate all instances in bags of size i into 25 large bags (with the possible exception of one small residual bag). We select instances from all bags of size$= i$ where $i \leq 150$, and divide them among 25 bags. It ensures that there are (almost) no bags of size ≤ 150. We also exclude all bags of size > 2500 to ensure mini-batches of bags fit in memory for training the DNN models. Figure 4 presents a scatter plot for bag counts versus the average number of instances in these bags (i.e. average bag-size) as we select different categorical feature combinations. Here, we only select the combinations of two categorical features that produced at least 10, 000 bags. We observe that as the number of bags increases for a given combination, the average bag-size decreases. For our experiments, we choose four such combinations with a diverse range of bag counts (see Table 2 and Figure 3).

MovieLens-1M [18]: We use the *zip-code* field to obtain the training bags. In other words, the bags are created using the geographical locations of users. Hence, these bags are likely to be correlated in terms of place. We obtain 3, 439 bags, one for each unique entry of *zip-code*. The statistics of training bags and distributions of training bag-sizes are provided in Table 2 and Figure 3 respectively.

5.1.3 *Training details.*

DNN architectures. We use the benchmark AutoInt architecture for our experiments on both Criteo-Kaggle and MovieLens-1M datasets [42]. DLLP models only require modifying the loss function without changing the architecture (see Equation 2). For our SelfCLR-LLP models, we append a dense layer with $(h \times K)$ nodes such that we can learn h latent features corresponding to each class $K = 2$ using the self-contrastive loss (Algorithm 1). We use h to 500 for Criteo-Kaggle and 250 for MovieLens-1M dataset.

Training hyper-parameters. We use the same hyper-parameters and learning rates for both DLLP and our SelfCLR-LLP models. We train the models using Adam optimizer [22]. For each iteration, we create the batches using 10 and 7 randomly sampled LLP training bags for Criteo-Kaggle and MovieLens-1M datasets respectively. We set α to 0.5 for both datasets and β to 0.01 and 0.001 for Criteo-Kaggle and MovieLens-1M datasets respectively across all experiments. For Criteo Kaggle, we train the models with a learning rate of 10^{-5} for the first 50, 000 iterations, followed by 10^{-6} for the last 20, 000 iterations. For MovieLens-1M, we set the learning rate to 5×10^{-5} for first 5, 000 iterations, followed by 10^{-5} for next 20, 000 iterations and finally 10^{-6} for last 10, 000 iterations. However, we observe that the DLLP models on ML-3K often overfit (Section 5.2.4). Hence, we stop training after 15, 000 iterations for DLLP on this LLP dataset. We provide detailed ablation studies and sensitivity analysis for different hyper-parameters in Section 5.3.

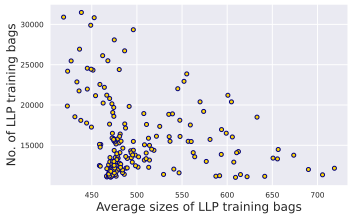


Figure 4: Scattered plots for bag counts versus the average number of instances in these bags (i.e. average bag-size) for different combination of categorical features using Criteo-Kaggle dataset.

	No. of Bags	Bag Sizes	Avg. positive proportions	Aggr. columns
Criteo-12K	12, 431	$503.31_{\pm 443.28}$	$0.25_{\pm 0.12}$	(C19, C26)
Criteo-17K	17, 461	$574.27_{\pm 498.38}$	$0.23_{\pm 0.12}$	(C16, C17)
Criteo-24K	24, 440	$586.27_{\pm 475.78}$	$0.23_{\pm 0.12}$	(C7, C14)
Criteo-33K	32, 879	$589.24_{\pm 519.10}$	$0.24_{\pm 0.13}$	(C2, C7)
ML-3K	3, 439	$171.91_{\pm 221.82}$	$0.81_{\pm 0.15}$	zip-code

Table 2: Statistics of training bags for different LLP datasets.

Datasets	DLLP	SelfCLR-LLP without $\mathcal{L}_{soft}$	SelfCLR-LLP with $\mathcal{L}_{soft}$	Supervised (instance-level)
Criteo-12K	$71.55_{\pm 0.09}$	$\mathbf{71.89_{\pm 0.16}}$	$\mathbf{72.00_{\pm 0.25}}$	*80.24*
Criteo-17K	$73.14_{\pm 0.14}$	$73.50_{\pm 0.09}$	$\mathbf{73.83_{\pm 0.14}}$	
Criteo-24K	$71.27_{\pm 1.14}$	$73.17_{\pm 0.16}$	$\mathbf{73.33_{\pm 0.19}}$	
Criteo-33K	$73.34_{\pm 0.17}$	$73.91_{\pm 0.09}$	$\mathbf{74.21_{\pm 0.09}}$	
ML-3K	$71.19_{\pm 1.98}$	$74.98_{\pm 0.94}$	$\mathbf{75.51_{\pm 0.10}}$	*85.75*

Table 3: AUROC (%) for different LLP datasets. Bold numbers represent statistically significant AUROC scores with p-value < 0.005.

We also train the AutoInt models in the fully supervised setting using the training datasets with instance-level labels. We use the same training set up as proposed in [42].

Performance Metric. We use *area under the receiver operating characteristics curves (AUROC)* as the metric to evaluate the performance on the test set using instance level labels [9]. Note that the model selection is challenging in LLP, as in practice, we cannot access the individual instance labels even for the validation set. Hence, we measure the average AUROC on the test set at an interval of 1,000 for the final 5,000 training iterations for comparison.

5.2 Performance Analysis

5.2.1 ***Quantitative results.*** Table 3 compares the performance of our proposed SelfCLR-LLP models with the existing DLLP [2, 13], the only applicable method among the existing LLP models. For our SelfCLR-LLP models, we separately report the results with and without incorporating the regularization term, $\mathcal{L}_{soft}$. Our SelfCLR-LLP models consistently outperform the existing DLLP models by incorporating $\mathcal{L}_{SelfCLR}$ for effectively utilizing the weak supervisions to learn better representations. Moreover, we further improve the performance by incorporating $\mathcal{L}_{soft}$ regularizer to increase the smoothness among the representations. Notably, even a minor improvement can be significant for the CTR prediction on the Criteo-Kaggle dataset. Several previous peer-reviewed methods often achieve only $\approx 0.2\%$ improvement in AUROC [17, 29, 42].

Note that the performance of both DLLP and our SelfCLR-LLP models remains significantly lower compared to the supervised setting. However, this is not surprising for two following reasons: **(1)** We receive *weak supervisions* from the LLP training bags. Further, the large LLP training bags often contain almost similar label proportions as the whole training set, providing extremely weak supervision for training. **(2)** An LLP dataset contains a significantly lower number of training bags, due to the aggregation of samples, compared to the number of labeled training instances for supervised settings. Therefore, the DNN models need to satisfy a significantly *lower number of constraints in an LLP setting*. For example, we receive $\approx$ 33.75 million training instance-label pairs to train the supervised models on the Criteo-Kaggle dataset. In contrast, we only receive a few thousand bags for LLP datasets (see Table 2). We observe that the performance LLP models (particularly for our SelfCLR-LLP) on the Criteo-Kaggle dataset typically improve as we receive more training bags. In particular, our self-contrastive loss decreases the pairwise similarities for representations within the same training bags to efficiently exploit the weaker supervisions.

Datasets	DLLP	SelfCLR-LLP$_{rand}$	SelfCLR-LLP
Criteo-24K	$71.27_{\pm 1.14}$	$73.21_{\pm 0.12}$	$73.33_{\pm 0.19}$
Criteo-33K	$73.34_{\pm 0.17}$	$73.89_{\pm 0.06}$	$74.21_{\pm 0.09}$

Table 4: Performance of SelfCLR-LLP degrades when the self-contrastive losses are computed using random training instances even after accessing the entire training set.

		DLLP	SelfCLR-LLP without $\mathcal{L}_{soft}$	SelfCLR-LLP with $\mathcal{L}_{soft}$
Criteo-24K	Negative Class	$1.07_{\pm 0.01}$	$1.00_{\pm 0.003}$	$1.00_{\pm 0.002}$
	Positive Class	$0.95_{\pm 0.005}$	$1.02_{\pm 0.002}$	$1.02_{\pm 0.02}$
Criteo-33K	Negative Class	$1.10_{\pm 0.005}$	$1.01_{\pm 0.001}$	$1.01_{\pm 0.002}$
	Positive Class	$0.91_{\pm 0.004}$	$1.02_{\pm 0.002}$	$1.02_{\pm 0.002}$
ML-3K	Negative Class	$0.98_{\pm 0.01}$	$1.03_{\pm 0.003}$	$1.03_{\pm 0.002}$
	Positive Class	$1.13_{\pm 0.02}$	$1.01_{\pm 0.005}$	$1.00_{\pm 0.002}$

Table 5: Relative inter-class similarity scores indicating unbiased representations for SelfCLR-LLP models.

5.2.2 ***Self-contrastive loss on randomly selected instances.*** Next, we investigate the importance of diversifying the representations within the LLP training bags. Here, we train another set of SelfCLR-LLP models for Criteo-24K and Criteo-33K datasets, denoted as SelfCLR-LLP$_{rand}$ using the same set of hyper-parameters as before. However, we now compute the unsupervised self-contrastive loss functions using randomly sampled training instances instead of the LLP training bags. We separately provide the training bags only to compute the proportion loss, $\mathcal{L}_{prop}$. Hence, we can access the entire training dataset in an unsupervised fashion. Recall that the training bags with bag size>2500 were inaccessible for SelfCLR-LLP and DLLP models for Criteo-Kaggle.

In Table 4, SelfCLR-LLP$_{rand}$ outperforms DLLP models. However, their performance remains lower than the SelfCLR-LLP models, particularly on Criteo-33K. Hence, explicitly diversifying the representations for input instances within the same training bags is more effective, even if we only access the training set partially.

5.2.3 ***Unbiased Representations.*** We analyze the *average relative similarity scores* to further validate the effectiveness of our self-contrastive loss. Here, we randomly select 10,000 test samples and obtain the representations. We compute the average absolute similarities between intra-class and inter-class test instances by measuring their average dot-products. We get the *relative intra-class similarity scores* by dividing the average absolute intra-class similarity and average inter-class similarity. For DLLP models, we obtain these representations from their penultimate layer. We repeat this experiment 25 times and report the mean and standard deviation of relative similarity scores for different classifiers in Table 5.

In Table 5, we observe that the DLLP models consistently produce relative intra-class similarity of > 1 for negative class instances. In contrast, these models achieve < 1 for positive class instances for LLP datasets derived from Criteo-Kaggle. It indicates the DLLP models are biased towards negative class following average label-proportions of training bags (Table 2). Similarly, we observe that the DLLP model for ML-3K is biased for the positive class as it receives training bags with higher average positive-label proportions. We also observe that as we train the DLLP models for more iterations,

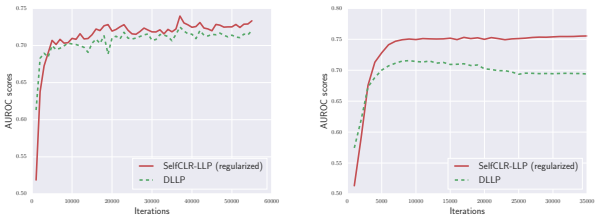


Figure 5: Convergence on (Left) Criteo-24K and (Right) ML-3K

	temp. scale & ℓ_2 norm		without temp. scale &
	$\tau = 0.5$	$\tau = 0.1$	ℓ_2 norm (Ours)
Criteo-24K	$68.79_{\pm 0.63}$	$69.36_{\pm 0.41}$	$\mathbf{73.33_{\pm 0.19}}$
Criteo-33K	$73.01_{\pm 0.04}$	$73.39_{\pm 0.09}$	$\mathbf{74.21_{\pm 0.09}}$
ML-3K	$74.32_{\pm 0.05}$	$70.52_{\pm 0.08}$	$\mathbf{75.51_{\pm 0.10}}$

Table 6: Effect of temp. scaling and normalization on SelfCLR-LLP.

their biases often increase. In contrast, by diversifying the representations within the same LLP training bags, SelfCLR-LLP models naturally tackle this issue. The relative intra-class similarity score of ≈ 1 indicates that SelfCLR-LLP models provide equal importance to both classes, producing unbiased representations.

Ideally, the intra-class relative similarity scores should be > 1 for fully supervised models. However, for SelfCLR-LLP models, we incorporate an additional $h \times K$ dimensional dense layer with h specific nodes for K different classes to produce h equidistant sub-clusters for each class without bringing them close to each other. Hence, even though the sub-clusters of different classes remain distinguishable from each other, we may not receive intra-class relative similarity scores of > 1 for SelfCLR-LLP models.

5.2.4 ***Convergence and Over-fitting.*** We design the experiment as: We choose a small instance-level set and report the AUROC scores at different training iterations on this validation set.

In Figure 5, we plot the AUROC scores of the classifiers at an interval of 1000 for Criteo-24K and ML-3K dataset respectively. We use a small validation set for Criteo-24K and the entire test set for the ML-3K dataset. We observe that the DLLP models tend to *overfit* for ML-3K datasets and often significantly degrade from their best AUROC scores as we keep training them. In contrast, our SelfCLR-LLP models remain stable after achieving their best AUROC scores. We observe that the DLLP model achieves the AUROC score of 71.55% after 15, 000 iterations. However, their performance significantly dropped to 69.38% after 30, 000 iterations. In contrast to the ML-3K dataset, we do not observe such phenomena for any variations of LLP datasets obtained from Criteo-Kaggle.

5.3 Ablation studies

5.3.1 ***Design choices.***

Temperature scaling & ℓ_2 normalization. We recall that the original SimCLR applies normalized temperature-scaled cross entropy or NT-Xent loss for representation learning on image datasets (see Equation 3) [6]. In Table 6, we compare the performance of SelfCLR-LLP as we apply temperature scaling and ℓ_2 feature normalization on representations, without incorporating the regularizer $\mathcal{L}_{soft}$ (as in [6]). We use temperature as $\tau = 0.1$ and $\tau = 0.5$ which were found to be effective in image domains. We keep the same learning rate and other hyper-parameters for training. However, we found that the ℓ_2 normalization and temperature scaling technique often significantly reduce the performance of our SelfCLR-LLP models for weakly supervised LLP datasets.

$\mathcal{L}_{prop}$		ℓ_1	ℓ_2^2	CE
Criteo-24K	DLLP	$71.67_{\pm 0.19}$	$72.39_{\pm 0.06}$	$71.27_{\pm 1.27}$
	SelfCLR-LLP	$\mathbf{73.52_{\pm 0.07}}$	$\mathbf{72.83_{\pm 0.21}}$	$\mathbf{73.33_{\pm 0.19}}$
Criteo-33K	DLLP	$72.67_{\pm 0.11}$	$73.06_{\pm 0.05}$	$70.46_{\pm 0.06}$
	SelfCLR-LLP	$\mathbf{74.03_{\pm 0.05}}$	$\mathbf{74.24_{\pm 0.06}}$	$\mathbf{74.0_{\pm 0.09}}$
ML-3K	DLLP	$68.45_{\pm 0.06}$	$72.66_{\pm 0.11}$	$70.70_{\pm 1.93}$
	SelfCLR-LLP	$\mathbf{74.86_{\pm 0.05}}$	$\mathbf{75.49_{\pm 0.02}}$	$\mathbf{75.51_{\pm 0.10}}$

Table 7: AUROC scores for different $\mathcal{L}_{prop}$ loss.

Choice of proportion loss. Several existing works explored different choices of $\mathcal{L}_{prop}$ including ℓ_1, ℓ_2^2 [24, 32] and cross-entropy (CE) functions [2, 13]. In Table 7, we compare the performance of our SelfCLR-LLP instantiated using these different $\mathcal{L}_{prop}$ functions. We see that SelfCLR-LLP models consistently achieve better performance compared to DLLP models for any choices of $\mathcal{L}_{prop}$.

5.3.2 ***Hyper-parameters selection.***

Number of bags per training batch. As we increase the mini-batch size, we can apply the contrastive loss on a larger set of instances. It allows the network to learn better discriminative representations [5, 6, 28]. However, increasing the number of bags per training iteration also increases the computational overhead of extracting and pre-processing the bags, increasing the overall training time. Further, the mini-batch sizes are often constrained due to hardware limitations such as RAM size in a GPU.

In Table 8 (a), we present the comparative performance as we set the mini-batch size to $\{4, 7, 10\}$ Criteo-24K and Criteo-33K and $\{3, 5, 7\}$ for ML-3K datasets while training the models for the same number of iterations. We observe that the performance of our SelfCLR-LLP models consistently improves as we increase the batch size and tend to converge for the mini-batch size of 10 bags Criteo-24K and Criteo-33K and 7 bags for ML-3K.

Number of nodes per class. Recall that we append a dense layer with $(h \times K)$ nodes for a K-class classification task for our SelfCLR-LLP models. The number of nodes per class, h controls the number of latent features learned for each class. In Table 8 (b), we present the performance as we vary h. As we choose smaller values of h, we cannot correctly learn the class-specific discriminative latent features. Hence, it leads to degrading performance. We find that SelfCLR-LLP tends to converge for $h = 500$ for Criteo-Kaggle datasets and $h = 250$ for MovieLens-1M dataset.

ReLU activation on SelfCLR-LLP representations. We apply the ReLU activation function for output $(h \times K)$ dimensional representations obtained from the SelfCLR-LLP models to improve the sparsity. We note that the benefits of producing sparse representation for classification is discussed in several previous studies [1, 14, 49]. In Table 8 (c), we compare the performance of SelfCLR-LLP models for different LLP datasets with and without applying ReLU activation. We find that the performance of SelfCLR-LLP slightly improves by using ReLU on the output representations.

Bags per batches	4	7	10
Criteo-24K	$72.79_{\pm 0.15}$	$\mathbf{73.32_{\pm 0.19}}$	$\mathbf{73.33_{\pm 0.19}}$
Criteo-33K	$73.25_{\pm 0.08}$	$73.83_{\pm 0.08}$	$\mathbf{74.21_{\pm 0.09}}$
Bags per batches	3	5	7
ML-3K	$74.05_{\pm 0.16}$	$\mathbf{75.49_{\pm 0.06}}$	$\mathbf{75.51_{\pm 0.10}}$

(a) Varying no. of bags per training batch.

h	250	500	1000
Criteo-24K	$72.81_{\pm 0.04}$	$73.33_{\pm 0.19}$	$73.52_{\pm 0.08}$
Criteo-33K	$73.78_{\pm 0.11}$	$74.21_{\pm 0.09}$	$74.21_{\pm 0.06}$
h	100	250	500
ML-3K	$75.12_{\pm 0.03}$	$75.51_{\pm 0.10}$	$75.52_{0.02}$

(b) Varying number of nodes per class

	without ReLU norm	With ReLU (ours)
Criteo-24K	$73.17_{\pm 0.07}$	$73.33_{\pm 0.19}$
Criteo-33K	$74.17_{\pm 0.04}$	$74.21_{\pm 0.09}$
ML-3K	$74.14_{\pm 0.05}$	$75.51_{\pm 0.10}$

(c) Effect of ReLU activation

Table 8: Choice of hyper-parameters for SelfCLR-LLP models.

	DLLP ($\alpha = 0$)	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 1.0$
Criteo-24K	$71.27_{\pm 1.14}$	$73.13_{\pm 0.03}$	$73.17_{\pm 0.16}$	$73.05_{\pm 0.10}$
Criteo-33K	$73.34_{\pm 0.17}$	$73.39_{\pm 0.05}$	$73.91_{\pm 0.91}$	$73.05_{\pm 0.06}$

(a) Varying α without using $\mathcal{L}_{soft}$ ($\beta = 0$).

	$\beta = 0.0$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 0.2$
Criteo-24K	$73.17_{\pm 0.16}$	$73.33_{\pm 0.19}$	$72.85_{\pm 0.17}$	$71.85_{\pm 0.25}$
Criteo-33K	$73.91_{\pm 0.91}$	$74.21_{\pm 0.09}$	$74.40_{\pm 0.06}$	$73.65_{\pm 0.19}$

(b) Varying β after selecting $\alpha = 0.5$.

Table 9: Sensitivity of α and β for AUROC scores.

	DLLP	SelfCLR-LLP	
Datasets		without $\mathcal{L}_{soft}$	with $\mathcal{L}_{soft}$
Criteo-12K	$6.92_{\pm 0.79}$	$\mathbf{19.56_{\pm 1.16}}$	$19.48_{\pm 1.17}$
Criteo-17K	$6.38_{\pm 1.44}$	$18.56_{\pm 1.49}$	$\mathbf{19.61_{\pm 1.62}}$
Criteo-24K	$7.57_{\pm 1.31}$	$20.10_{\pm 1.13}$	$\mathbf{20.34_{\pm 1.11}}$
Criteo-33K	$7.48_{\pm 1.38}$	$19.63_{\pm 1.58}$	$\mathbf{19.89_{\pm 1.58}}$
ML-3K	$\mathbf{19.91}_{\pm 7.81}$	$18.12_{\pm 3.00}$	$17.13_{\pm 2.90}$

Table 10: Average diversity scores ($\times 100$) over LLP training bags.

Sensitivity of α and β. The self-contrastive loss for our SelfCLR-LLP consists of two weight parameters, α and β corresponding to $\mathcal{L}_{selfCLR}$ and $\mathcal{L}_{soft}$ (Equation 7). In the following, we first vary α with $\beta = 0$ to analyze the sensitivity of $\mathcal{L}_{selfCLR}$ without $\mathcal{L}_{soft}$.

In Table 9 (a) presents the results for $\alpha = \{0.25, 0.5, 1.0\}$. Note that the choice of $\alpha = 0$ is equivalent to the DLLP method [2]. As we reduce the weights for $\mathcal{L}_{SelfCLR}$, the models cannot produce sufficient diversity within the LLP training bags. In contrast, the choice of large α tends to ignore the weak-supervision from $\mathcal{L}_{prop}$ loss. In Table 9 (a) we observe that we obtain the best performance by choosing $\alpha = 0.5$ for Criteo-Kaggle datasets.

Table 9 (b) presents the sensitivity of $\mathcal{L}_{soft}$ for SelfCLR-LLP models. Here, we set $\alpha = 0.5$ and vary β. We can see that the performance of SelfCLR-LLP models improves as we choose small β. However, the appropriate choice and sensitivity of β are dataset-dependent. For example, the performance of SelfCLR-LLP improves on Criteo-33K as we increase β from 0.01 to 0.1. In contrast, the same choice of $\beta = 0.1$ degrades their performance on Criteo-24K. However, as we keep increasing the value of β, the model produces uniform representations for all feature coordinates. Therefore, it eventually degrades the overall classification performance. We use $\beta = 0.01$ for all LLP datasets obtained from Criteo-Kaggle.

5.3.3 ***LLP Bag-level diversity.*** One may define the diversity score, $\mathcal{D}(Z)$ for a set of representations, Z, with real-valued entries as: $\mathcal{D}(Z) = 1 - \frac{||Z||_2}{||Z||_F} = 1 - \frac{\sigma_{max}(Z)}{\left[\sum_i \sigma_i(Z)^2\right]^{\frac{1}{2}}}$ – where, $||Z||_2$ denotes the spectral-norm of Z, given by the maximum singular value $\sigma_{max}(Z)$. $||Z||_F$ denotes the frobenius norm of Z, given by the ℓ_2 norm of singular values of Z. σ_i are the singular-values of Z. Since $||Z||_2 \leq ||Z||_F$, the minimum value of $\mathcal{D}(Z)$ is bounded by 0.

For a set of representations in a similar direction, we get a large singular value corresponding to the principal component, and the magnitude of other singular values becomes smaller. Hence, we get smaller $\mathcal{D}(Z)$ when Z contains similar representations. Consequently, a diverse set of representations produces multiple principal component directions, leading to larger $\mathcal{D}(Z)$.

Table 10 presents the average diversity scores, $\mathcal{D}$ over LLP training bags for different models. We randomly select 2,500 bags and report $\mathcal{D}$. For DLLP models, we use the representations from the penultimate dense layer after applying batch-normalization, followed by the ReLU activation function. For our SelfCLR-LLP models, we obtain the representations from the final $(h \times K)$ dimensional dense layer, followed by a ReLU activation (Figure 1).

Note that $\mathcal{D}(Z)$ can only distinguish between a set of vectors in the same direction versus the vectors with multiple different directions. However, given a set of vectors in different directions, it cannot detect whether they are uniformly distributed over the feature space. For example, it would not distinguish between the vectors in Figure 2(a) and Figure 2(b). Hence, we obtain similar diversity scores with and without applying $\mathcal{L}_{soft}$.

Notably, our SelfCLR-LLP models consistently produce higher $\mathcal{D}$, compared to DLLP models for Criteo-Kaggle. In contrast, we achieve similar $\mathcal{D}$ from all models for the ML-3K with slightly higher scores for DLLP models. However, the *high standard deviation of* $\mathcal{D}$ for the DLLP model indicates that it produces higher diversity for some training bags. Hence, it may lead to *random representations*. Subsequently, it leads to significantly lower AUROC scores for DLLP (Table 3). On the other hand, we incorporate the diversity in representations in a controllable fashion to obtain discriminative features for improving overall classification performance.

6 CONCLUSION

We study the problem of weakly supervised learning from label proportions for tabular datasets with both continuous and categorical features. While incorporating auxiliary representation learning loss can boost performance significantly, we cannot apply these existing methods to 'unstructured' tabular datasets. We argue that producing diverse representations allow the classifiers to operate on each instance separately of a training bag, improving the overall classification performance. We propose an auxiliary self-contrastive loss to explicitly maximize the divergence of representations within the same LLP training bags. Extensive experiments on real-world e-commerce applications with benchmark datasets demonstrate that our proposed SelfCLR-LLP models consistently outperform the existing baseline.

REFERENCES

[1] Mahdi Abavisani and Vishal M Patel. 2019. Deep sparse representation-based classification. *IEEE Signal Processing Letters* (2019).

[2] Ehsan Mohammady Ardehaly and Aron Culotta. 2017. Co-training for demographic classification using deep learning from label proportions. In *ICDM Workshops (ICDMW)*.

[3] Ricardo Baeza-Yates, Di Jiang, Fabrizio Silvestri, and Beverly Harrison. 2015. Predicting the next app that you are going to use. In *Proceedings of the eighth ACM international conference on web search and data mining*. 285–294.

[4] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS* (2019).

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *ICML*.

[7] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. 2019. Self-supervised gans via auxiliary rotation loss. In *CVPR*.

[8] Criteo. 2014. Kaggle Display Advertising Challenge Dataset. http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset/

[9] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*. 233–240.

[10] Nando de Freitas and Hendrik Kuck. 2012. Learning about individuals from group statistics. *UAI* (2012).

[11] Carl Doersch, Abhinav Gupta, and Alexei A Efros. 2015. Unsupervised visual representation learning by context prediction. In *CVPR*.

[12] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. 2019. Gradient descent provably optimizes over-parameterized neural networks. *ICLR* (2019).

[13] Gabriel Dulac-Arnold, Neil Zeghidour, Marco Cuturi, Lucas Beyer, and Jean-Philippe Vert. 2019. Deep multi-class learning from label proportions. *arXiv* (2019).

[14] Yuan Gao, Jiayi Ma, and Alan L Yuille. 2017. Semi-supervised sparse representation based classification for face recognition with insufficient labeled samples. *IEEE Transactions on Image Processing* (2017).

[15] John M Giorgi, Osvald Nitski, Gary D Bader, and Bo Wang. 2021. Declutr: Deep contrastive learning for unsupervised textual representations. *ACL* (2021).

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *NeurIPS*.

[17] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *IJCAI*.

[18] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* (2015).

[19] Jerónimo Hernández-González, Inaki Inza, Lorena Crisol-Ortíz, María A Guembe, María J Iñarra, and Jose A Lozano. 2018. Fitting the data from embryo implantation prediction: Learning from label proportions. *Statistical methods in medical research*.

[20] Jerónimo Hernández-González, Inaki Inza, and Jose A Lozano. 2013. Learning Bayesian network classifiers from label proportions. *Pattern Recognition* (2013).

[21] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. *NeurIPS* (2020).

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv* (2014).

[23] Nikos Komodakis and Spyros Gidaris. 2018. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*.

[24] Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *ACM SIGKDD*.

[25] Kuan-Ting Lai, Felix X Yu, Ming-Syan Chen, and Shih-Fu Chang. 2014. Video event detection by inferring temporal instance labels. In *CVPR*.

[26] Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML workshop*.

[27] Fan Li and Graham Taylor. 2015. Alter-cnn: An approach to learning from label proportions with application to ice-water classification. In *NeurIPS Workshops*.

[28] Junnan Li, Pan Zhou, Caiming Xiong, and Steven C.H. Hoi. 2021. Prototypical Contrastive Learning of Unsupervised Representations. In *ICLR*.

[29] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *SIGKDD*.

[30] Jiabin Liu, Bo Wang, Zhiquan Qi, YingJie Tian, and Yong Shi. 2019. Learning from Label Proportions with Generative Adversarial Networks. *NeurIPS*.

[31] Jiabin Liu, Bo Wang, Xin Shen, Zhiquan Qi, and Yingjie Tian. 2021. Two-stage Training for Learning from Label Proportions. *IJCAI*.

[32] David R Musicant, Janara M Christensen, and Jamie F Olson. 2007. Supervised learning by training on aggregate outputs. In *ICDM*.

[33] Mehdi Noroozi and Paolo Favaro. 2016. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*.

[34] Conor O'Brien, Arvind Thiagarajan, Sourav Das, Rafael Barreto, Chetan Verma, Tim Hsu, James Neufield, and Jonathan J Hunt. 2022. Challenges and approaches to privacy preserving post-click conversion prediction. *arXiv preprint arXiv:2201.12666* (2022).

[35] Yujie Pan, Jiangchao Yao, Bo Han, Kunyang Jia, Ya Zhang, and Hongxia Yang. 2021. Click-through Rate Prediction with Auto-Quantized Contrastive Learning. *arXiv* (2021).

[36] Giorgio Patrini, Richard Nock, Paul Rivera, and Tiberio Caetano. 2014. (Almost) no label no cry. *NeurIPS*.

[37] Novi Quadrianto, Alex J Smola, Tiberio S Caetano, and Quoc V Le. 2009. Estimating labels from label proportions. *JMLR* (2009).

[38] Stefan Rueping. 2010. SVM classifier estimation from group probabilities. In *ICML*.

[39] Rishi Saket, Aravindan Raghuveer, and Balaraman Ravindran. 2022. On Combining Bags to Better Learn from Label Proportions. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 5913–5927.

[40] Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green ai. *Commun. ACM* 63, 12 (2020), 54–63.

[41] Clayton Scott and Jianxin Zhang. 2020. Learning from Label Proportions: A Mutual Contamination Framework. *NeurIPS* (2020).

[42] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *CIKM*.

[43] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. 2018. The implicit bias of gradient descent on separable data. *JMLR* (2018).

[44] Kuen-Han Tsai and Hsuan-Tien Lin. 2020. Learning from label proportions with consistency regularization. *ACML*.

[45] Vikas Verma, Kenji Kawaguchi, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. 2019. Interpolation consistency training for semi-supervised learning. *IJCAI* (2019).

[46] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc Le. 2021. Towards domain-agnostic contrastive learning. In *ICML*.

[47] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive Learning for Cold-Start Recommendation. *arXiv* (2021).

[48] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2020. Self-supervised Learning for Large-scale Item Recommendations. *arXiv* (2020).

[49] Jun Yin, Zhonghua Liu, Zhong Jin, and Wankou Yang. 2012. Kernel sparse representation based classification. *Neurocomputing* (2012).

[50] Felix X. Yu, Dong Liu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. 2013. ∝SVM for Learning with Label Proportions. In *ICML*.

[51] Richard Zhang, Phillip Isola, and Alexei A Efros. 2016. Colorful image colorization. In *ECCV*.

[52] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. *arXiv* (2020).

[53] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *The Web Conference*.