A Additional Analysis

A.1 Additional Ablations

A.1.1 1-Nearest Neighbor captures a lot of the gains

We now show that using 1 Nearest Neighbour Information for the `SUM-PRODUCT-BP` is empirically close enough to the best performance we get on the best k chosen for k-nearest neighbors used in `SUM-PRODUCT-BP` by hyperparameter search. From Figure 4, we see that across multiple datasets test metrics are very close for the two settings. This shows lower k which reduces time complexity of the BP step (see section 6.1) does not affect the performance greatly.

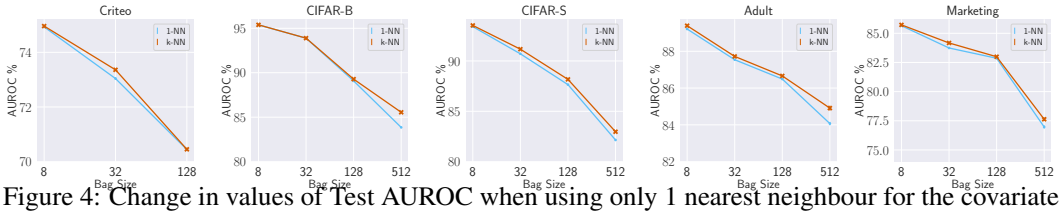

Figure 4: Change in values of Test AUROC when using only 1 nearest neighbour for the covariate factor creation, v/s when using an optimal higher k for the covariate factor creation.

A.1.2 Noisy Embeddings as Input

We add noise variables sampled from $\mathcal{N}(0, \sigma^2 I_d)$ to our features input to the first iteration and report the numbers obtained in 2nd-Iteration supervised learning step in figure 5. Medium Noise regime corresponds to $\sigma = 0.05$ and High Noise regime corresponds to $\sigma = 0.1$. As is clearly visible our method is able to recover performance even when using noisy inputs. We would like to note that there is some degradation at bag size level 512. We point out that this is case where there about ~ 100 bag level labels in total. Covariate information is rather crucial to make any progress. Hence noise addition has the most impact in this regime. This also suggests importance of using covariate information for large bag sizes due to very weak supervision available. The drop in performance due to noisy embeddings is significantly higher for the *Marketing* dataset than the *Adult* dataset. This shows that the coariates are very important for the Marketing datasets and our method exploits it very well. We posit thatt his could be the reason our method significantly outperforms others on the *Marketing* dataset (See Table 1).

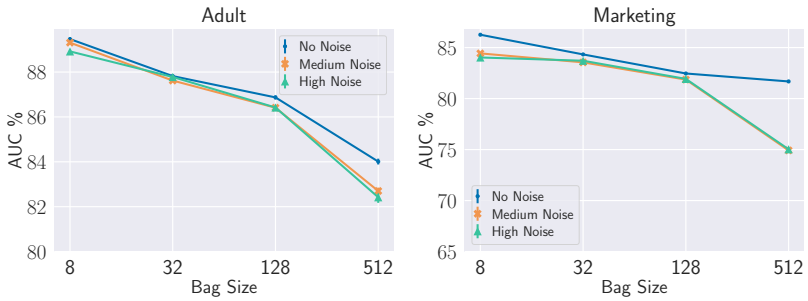

Figure 5: Recovered performance on adding noise to the initial embeddings.

A.1.3 Soft Weighted Hard Threshold v/s Vanilla Hard Threshold

In figure 6 we report the numbers obtained on using the soft-labels from the BP-Marginals as opposed to hard thresholding them. We use the soft labels in two ways, directly to train the MLP using a sigmoid cross entropy loss formulation as opposed to the usual binary cross entropy loss, and the other for weighing the hard-labels by $|p - \tau|$ where p is the soft label and τ the threshold for creating the hard labels. We do not notice any consistent improvements on using the soft labels in either form across datasets and bag sizes and thus stick to hard labels for our setup.

A.1.4 Distance Metric: Cosine v/s L2

As mentioned earlier, we experiment with using Cosine and L2 distance, $d(\cdot, \cdot)$ for the construction of our neighbour graph. While we don't find significant differences on using the two methods, using

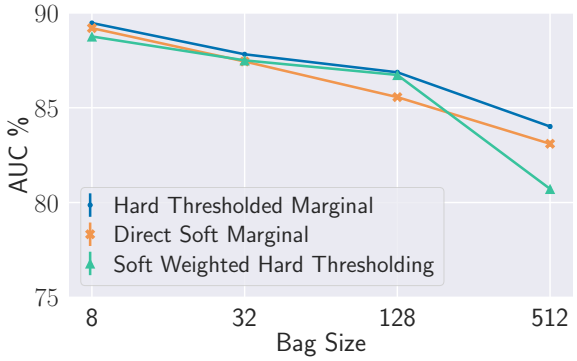

Figure 6: Change in values of MLP AUROC when using Soft Weighted Hard Thresholding for the conversion of BP-Marginals to pseudolabels v/s Directly using the soft marginals v/s using Hard Thresholding on Adult Dataset for 2nd Iteration.

Cosine led to better downstream performance across datasets and bag sizes. This can be interpreted to be due to the better neighbour graph construction as depicted in Figure 7 for Adult, by better Test Score (Accuracy) of the kNN constructed by the two distance metrics for varying number of neighbours (k) for the construction of the neighbour graph.

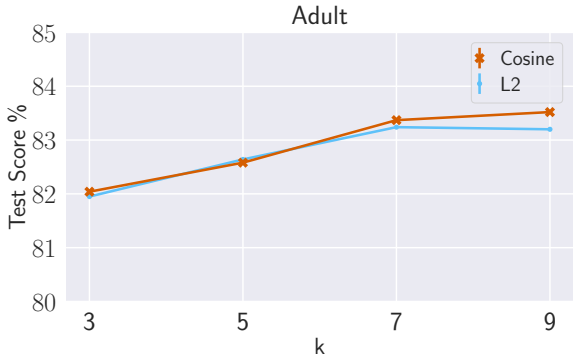

Figure 7: Variation in the kNN Score across bag sizes compared for the two popular distance metrics, Cosine and L2 on Adult Dataset as $d(\cdot, \cdot)$ for the neighbour graph creation.

A.1.5 Similarity Kernel: RBF v/s Matern

As discussed earlier, we tried both the RBF Kernel and Matern Kernel for our experiments and note as in Figure 8 for Criteo, that the Matern Kernel resulted in slightly better Test AUROC % Scores across various datasets and bag sizes. While there is no substantial increase in our performance the marginally better numbers can be attributed to the fact that the Matern Kernel is a generalization of the RBF Kernel and might capture the similarity information between the embeddings more aptly.

A.1.6 Optimal Values of λ_a

As observed in Figure 9, it is clear that the bag-loss head plays a more important role when the bag sizes are smaller. The optimal value of λ_a is dependent on the requirement of the reinforcement of the bag constraint via the bag-loss head. For small bags, the bag constraints hold much more information than that for large bags, and hence are more useful. In the case of small bags, during aggregation of embeddings, more information is retained and utilized downstream since fewer embeddings are pooled. We pool 8 embeddings per bag for bag size 8 and 512 embeddings are pooled for bag

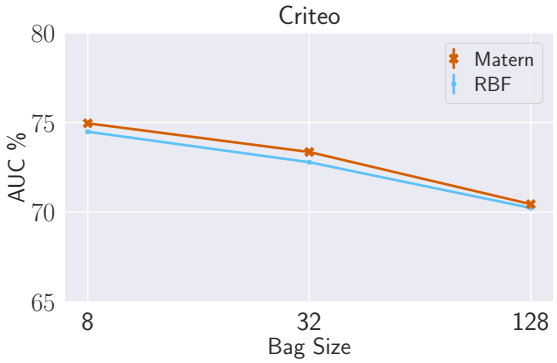

Figure 8: Variation in the Test AUROC % of the MLP after 1st Iteration on using RBF Kernel v/s on using Matern Kernel as $k(\cdot, \cdot)$ for similarity calculation during formation of pairwise factors on Criteo.

size 512, clearly there is a stark divide in the information summarized across bag sizes. This trend is consistently noticed across multiple datasets. We also notice that the optimal values of λ_a are comparatively lower for the 2nd Iteration of our method. We think this might be due to the refined embeddings and neighbour graph in the 2nd iteration of Belief Propagation.

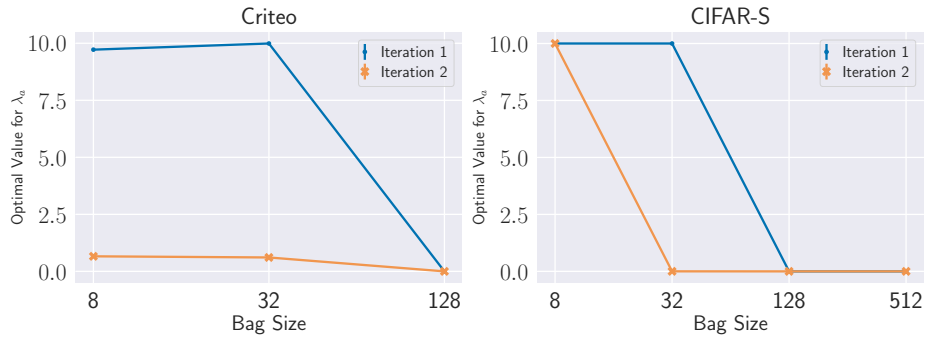

Figure 9: Variation in the Test AUROC % of the MLP after 1st Iteration on using RBF Kernel v/s on using Matern Kernel as $k(\cdot, \cdot)$ for similarity calculation during formation of pairwise factors on Criteo.

A.2 Extended Experimentation

A.2.1 Wall Clock Times on Adult Dataset

We have wall clock time reported in Table 4 (main paper) and Table 5 for Criteo and Adult respectively. Criteo has 1 million samples in total in the training. For bag size 32, the wall clock time of the entire BP stage is only 1279s for millions of factors in the factor graph inclusive of the creation and iteration. For smaller datasets like Adult with 50k samples, even on bag size as large as 2048, the BP stage takes only 1054s on a NVIDIA P100 GPU, which is a minimal computational overhead above standard supervised learning. It is also conventionally known that BP on sparse graphs is very fast and our sparsity is controlled by using K-NN instead of all pairs in imposing nearness constraints. We reiterate that the number of factors in the factor graph is thus only linear in the number of samples allowing for faster iterations. The time taken in the entire BP section, right from creation of the factor graph to message passing for 100 iterations happens in the order of $O(\text{bag\_size})$ seconds for the Adult dataset which is exceptionally fast as simple MLP training itself takes $O(10^2)$ seconds.

Table 5: Time for various parts of our algorithm compared to time taken by DLLP on Adult Dataset. All time values are in seconds.

	Adult ∼50k Samples				
Bag Size	DLLP Training	Ours - Data Setup	Ours - BP	Ours - MLP	Ours - Total
8	52.23 (51.38)	630.88 (827.63)	12.34 (8.57)	85.00 (145.57)	728.21 (863.69)
32	45.4 (69.04)	523.59 (449.32)	21.11 (14.65)	54.38 (77.15)	599.08 (458.12)
128	73.79 (125.96)	623.20 (458.23)	71.16 (8.89)	54.34 (63.55)	748.71 (459.86)
512	25.11 (40.15)	767.58 (431.50)	269.88 (32.52)	65.07 (158.34)	1102.53 (459.12)
1024	26.39 (38.06)	645.04 (674.94)	550.32 (113.28)	32.42 (25.76)	1227.78 (756.34)
2048	24.34 (28.32)	612.87 (539.07)	1054.47 (73.71)	49.93 (78.34)	1717.27 (534.88)

A.2.2 Convergence of the two step method

In Table 6 empirically we demonstrate that our method converges in two iterations by looking at relative improvements between iterations 2 and 3. We demonstrate that two iterations of our algorithm suffice empirically.

Table 6: Empirical Convergence: The % AUROC scores for varying bag sizes for 2 Iterations and 3 Iterations of our algorithm.

	Adult				Marketing			
Bag Size	8	32	128	512	8	32	128	512
Itr-2	**89.47**	**87.82**	86.87	84.01	**86.26**	**84.33**	**82.46**	**81.68**
Delta	-0.29	-1.26	0.49	0.71	-0.26	-0.94	-0.13	-1.05
Itr-3	89.18	86.56	**87.36**	**84.72**	86.00	83.39	82.33	80.63

As visible, performance gains from 2nd to 3rd iterations are not consistently better. Thus there is no clear reason to run higher iterations of the algorithm, as a maximum of 2 iterations suffice to achieve significantly consistent performance.

A.2.3 Goodness of Pseudo Label of BP

We report the AUROC of BP after the first iteration with respect to the true labels in Table 7. It is considerable indicating that it has good ordering information (ranking of samples belonging to class 1 above class 0). The effect of high quality pseudo labels is reflected in the downstream performance.

Table 7: The % AUROC scores of the pseudo labels obtained from the Belief Propagation algorithm when compared to the ground truth labels for iteration 1 of the algorithm

Bag Size	8	32	128	512	1024	2048
Adult	86.34	79.63	75.25	75.02	67.88	63.54
Marketing	88.53	78.26	75.5	75.66	74.9	74.64

While Table 7 only reports after Step 1 of iteration 1 to showcase value of the BP step, the second aggregate embedding loss based MLP training Step 2 boosts performance of Step 1 further and we do see it as expected in Table 1. Refer to Section 4, subsection 4.1 for Step 1, and subsection 4.2 for description of these steps in our algorithm.

Step 2 is necessary because information from pseudo labels may not satisfy bag constraints exactly. So we have a composite loss that again imposes the bag constraint through an aggregate embedding loss (see Equation 8 and Equation 9 in the paper in Section 4.2)

A.2.4 Noisy Labels and Privacy

Note: We will update the citations in the main paper to include the updated citations mentioned as footnote in the supplementary section.

In this section we explore utility-privacy tradeoff of our algorithm when we add noise to label proportions for every bag by Gaussian Mechanism to achieve a target *label differential privacy* of (ϵ, δ) by using the following result:

Theorem A.1 (Theorem 2 in [8]). *Let $f : \mathcal{A} \to \mathbb{R}$ be a real-valued function. Let $\tau = \Delta f \sqrt{2\ln(1.25/\delta)}/\epsilon$. The Gaussian Mechanism, which adds independently drawn random noise distributed as $\mathcal{N}(0, \tau^2)$ to output of $f(A)$, ensures (ϵ, δ)-differential privacy.*

We take $\mathcal{A}$ to the set of true labels of instances in bag $S \in \mathcal{B}$, $f(\cdot) = \frac{y(S)}{B}$, the label proportion. We observe that sensitivity of the label proportion to change in a single label is $\Delta f = \frac{1}{B}$, where B denotes the bag size. Standard deviation of the noise added is proportional to $1/B$ for a fixed ϵ, δ.

We demonstrate the following interesting privacy-utility tradeoff: utility degradation, as measured by Test AUROC, due to Gaussian Mechanism is much more in smaller bags as compared to larger bags for a target privacy level. Through this, we empirically verify the intuition that points to the fact that larger bags offer better privacy. We note that our algorithm performs much better in large bags regime compared to baseline and we conjecture that this is crucial to utilize the better privacy utility degradation tradeoffs at larger bag sizes.

We experiment with 2 sets of differential privacy parameters, Medium Noise: $(\delta, \epsilon) = (10^{-5}, 10)$ and High Noise: $(\delta, \epsilon) = (10^{-5}, 1)$, both popular choices in literature [22] Note that, bag size B takes values in $\{8, 32, 128, 512\}$. Our results are reported in Table 8.

Table 8: Test AUROC scores after Iteration-1 of our method across different bag sizes for varying levels of label-noise.

Dataset:	Criteo			CIFAR-B			CIFAR-S		
Noise:	Noiseless	Medium	High	Noiseless	Medium	High	Noiseless	Medium	High
8	74.96 (0.01)	74.83 (0.07)	71.06 (0.15)	95.39 (0.01)	94.80 (0.04)	90.99 (0.1)	93.53 (0.03)	93.28 (0.28)	87.07 (0.46)
32	73.36 (0.03)	72.43 (0.02)	70.40 (0.03)	93.89 (0.02)	93.36 (0.05)	90.25 (0.06)	91.17 (0.03)	91.07 (0.21)	86.08 (0.07)
128	70.45 (0.05)	69.45 (0.20)	69.53 (0.21)	89.28 (0.05)	88.92 (0.13)	87.79 (0.09)	88.17 (0.19)	87.39 (0.10)	85.17 (0.23)
512	-	-	-	85.55 (0.75)	83.29 (0.32)	84.65 (0.21)	82.97 (0.33)	81.32 (0.37)	79.39 (0.58)

We also want to highlight that under the effect of both medium and high noise our method recovers performance up to a reasonable degree, especially for larger bag sizes which as stated earlier are more important from a privacy perspective.

A.3 Analysis of 1-NN Graphs

Section A.1.1 shows that running our algorithm with 1-NN for the BP step captures most of the performance in terms of the final Test AUROC score. We show that many parts of the factor graph in the case of 1-NN are cycle free.

Consider the bi-partite factor graph $K(V, \mathcal{B} \cup \mathcal{F}, E)$ where variable nodes $V = [1 : N]$ representing $\{x_i\}$ form one partition and bag factor nodes $\mathcal{B}$ and 1-NN factor nodes $\mathcal{F} = \{f : (f, i), (f, j) \in E,\ x_i \in N_1(x_j) \vee x_j \in N_1(x_i)\}$ are on the other partition. Edges between a bag factor node S and variable node i exists if $i \in S$ (x_i belong to bag S).

In our setup (experiments), all bag factor nodes have disjoint neighbors since bags are formed randomly without replacement. Therefore, the bi-partite factor graph between $\mathcal{B}$ and V is a forest. Now, consider the bi-partite factor graph between $\mathcal{F}$ and V. We now show that this is also a forest. Since every factor node connects only a pair of distinct nodes it is enough to show that the undirected 1-NN graph does not have any cycles.

We now show that the 1-NN graph does not have any undirected cycles. Recall that $N_1(x)$ is the nearest neighbor of point x.

Lemma A.2. *For a set of points $\{x_i\}_{i=1}^N$, consider the following undirected graph $G(V, E)$ where $V = [1 : N]$ and $E = \{(i, j) : x_i \in N_1(x_j) \vee x_j \in N_1(x_i)\}$. For every node i, if the edge set is formed by choosing one amongst many equivalent nearest neighbors of i appropriately (i.e. $N_1(x_i)$ is chosen to be a singleton), then G does not have any cycles.*

Proof. Let us define a directed graph G_d, where the outgoing edge $(i \to j)$ exists if x_j is the closest neighbour of x_i where ties are broken arbitrarily. Thus every node i has out-degree of at-most 1. However, note that the in-degree of any node can be > 1. Note that, G can be obtained by replacing oriented edges in G_d by undirected edges. Further, if $(i \to j), (j \to i)$ both exists, we replace it by one undirected edges $(i, j) \in G$.

There are 3 types of cycles in G_d:

1. Directed cycle with at least 3 distinct elements (except end point which is repeated). An example of this kind (of length 3) is illustrated in Figure 11.
2. Cycle with a collider whose undirected skeleton forms a cycle in G of length at least 3 with distinct elements. This is illustrated in Fig. 12.
3. Directed cycle $i \to j \to i$. This is illustrated in Figure 10.

Now, we proceed to show that the first two cycles are not possible. Since a bi-directed edge in G_d (Figure 10) will get replaced by a single un-directed edge in G, this proves the Lemma.

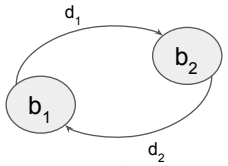

Figure 10: Cycle of Type 3

Case 1: We deal with the directed cycle by first proving a claim about a directed path in G_d.

Claim A.3. *Consider a directed path in G_d of the form $a_1 \xrightarrow{d_1} a_2 \xrightarrow{d_2} a_3 \xrightarrow{d_3} a_4 \dots a_n$ with distinct elements. Here, d_i notes the distance $d(x_{a_i}, x_{a_{i+1}})$. Then, $d_1 \geq d_2 \geq d_3 \dots d_{n-1}$.*

Proof. Let us prove this by contradiction. Say this was not true, then without loss of generality suppose $d_i < d_{i+1}$. We know that the outgoing edge represents the nearest neighbour of a node. If $d_i < d_{i+1}$ was indeed true, then the nearest neighbour of a_{i+1} would have been a_i and not a_{i+2} which is clearly not the case since the edge $a_{i+1} \xrightarrow{d_{i+1}} a_{i+2}$ exists in G_d. We get a contradiction and therefore the claim is proven. □

Now, we consider the directed cycle $a_1 \xrightarrow{d_1} a_2 \xrightarrow{d_1} a_3 \dots a_n \xrightarrow{d_n} a_1$ where all elements a_i are distinct.

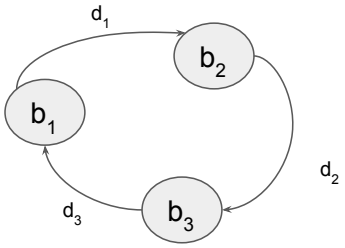

Figure 11: Cycle of Type 1

From, Claim A.3 we have $d_1 \geq d_2 \geq d_3 \geq d_n \geq d_1$ applying it on two directed paths a_n, a_1, a_2 and $a_1, a_2 \dots a_n$. This is only possible if $d_1 = d_2 \dots = d_n$. In this case, one can form an alternate G_d by breaking the cycle where one can have $a_2 \to a_1$ instead of $a_2 \to a_3$.

Therefore, this type of a cycle cannot exist. If it exists, it can be broken by re-assigning an equivalent nearest neighbor.

Case 2: Consider a configuration that is a undirected cycle in G but not a directed cycle in G_d. Then, it is a cycle consists of paths $a_1, a_2 \dots \to a_n$, $a_1, b_2, b_{n-1} \to a_n$ where they *collide* at a_n. Here all nodes a_i, b_i are distinct. An Example is the collider c_3 in Fig. 12. Other orientations are left unspecified in this cycle. For this to there must exist $a_i \neq a_n$ or b_i that has 2 outward edges in this

cycle. However, $\text{out} - \text{degree} \geq 2$ is clearly cannot be possible as we are only dealing with 1-NNs and each node can have at-most one outward edge. Therefore such a cycle is not possible. In the Figure 12, the edge $c_1 \to c_4$ or $c_1 \to c_2$ cannot exist as G_d is a directed 1-NN graph and thus such a cycle cannot exist.

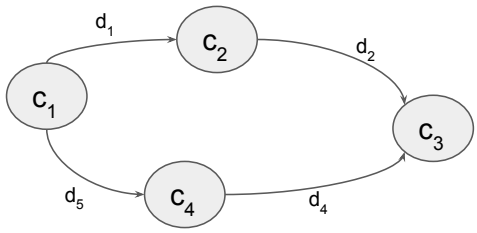

Figure 12: Cycle of Type 2

Thus, we have shown that no cycle can exist in the neighbour graph G. □

A.4 Baselines

We compare our methods with the following baselines.

1. DLLP: We use the DLLP method from Ardehaly and Culotta [3] as a baseline for both Tabular and Image Datasets. This method fits the prediction score averaged over a bag of a deep classifer to bag level proportions.

2. EasyLLP: This was proposed in Busa-Fekete et al. [5] and we use this as a competitive baseline on all our datasets. They define a surrogate loss function based on the global label proportion.

3. GenBags: Introduced in Saket et al. [30] is another popular algorithm for tabular datasets. The algorithm combines bag distributions, if possible, into good generalized bag distributions, which are then trained on by using standard proportion loss.

4. LLP-FC: This methods was introduced in Zhang et al. [39] where LLP problem was reduced to learning from label noise problem. We use this for image datasets on which it was applied in Zhang et al. [39].

5. LLP-VAT: Method from Tsai and Lin [34] that we use on Image Datasets. This method is inspired by consistency regularization to produce a decision and approach LLP from a semi-supervised angle. boundary that better describes the data manifold

A.5 Implementation Details

In most of our experiments, we fix $d(x, x')$ to be Cosine Distance: $1 - \frac{x \cdot x'}{\|x\|_2 \|x'\|_2}$ or Euclidean Distance: (Minkowski Distance with p=2) $(\sum_{i=1}^{n} |x_i - x'_i|^p)^{\frac{1}{p}}$ and we choose $k(x, x')$ to be one of RBF Kernel: $\exp(-\gamma \cdot d(x, x')^2)$ or Matern Kernel with default parameters from their tensorflow implementation: (a generalization of the RBF kernel) $\frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\frac{\sqrt{2\nu}}{l} d(x, x') \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}}{l} d(x, x') \right)$ where $d(\cdot, \cdot)$ is the Euclidean distance, $K_v(\cdot)$ is a modified Bessel function and $\Gamma(\cdot)$ is the gamma function. We justify our choice via the slightly superior performance observed on using Cosine Distance and Matern Kernel as discussed in ablations in A.1.4 and A.1.5 respectively.

The bags are batched into batches of size $max(BatchSize_{train}, TotalBags)$. We always run the second step, the MLP training for max 100 epochs, using Adam Optimizer with MLP_{LR}, MLP_{WD} learning rate and weight decay respectively. We use the Early Stopping criterion to decide when to stop training. According to this, we stop training if the validation AUROC does not increase for 20 consecutive epochs, and then restore the model with the best validation AUROC. Such a validation-based early stopping technique is quite popular in literature and well described in Prechelt [27]. Attached is the tensorflow callback code that we implement for the same, based on documentation provided in ker [2]:

```
tf.keras.callbacks.EarlyStopping(
          monitor='val_auc',
          patience=20,
          restore_best_weights=True,
          min_delta=0,
          verbose=1,
          mode='max',
      )
```

We use the official GitHub Implementations of Saket et al. [30], Zhang et al. [39] and Tsai and Lin [34] and perform a grid search over relevant mentioned parameters in their readme. We use WideResNet-16-4 [38] as the backbone for LLP-VAT and LLP-FC methods as it provides the best performance. For methods described in Busa-Fekete et al. [5], Ardehaly and Culotta [3] we implement the algorithm described in the paper with the same MLP as described in 5.1 and sweep the appropriate hyperparams as described in the respective papers.

For Pooling with MultiHeadAttention we use the standard MultiHeadAttention framework from Set Transformers [17] using $d = 128$ dimensional embeddings as input (the 2nd last hidden layer of our MLP), 2 heads, 1 seed vector, and 2 row-wise feedforward layers each of size d.

A.6 Illustrative Best Hyper-parameter values

Here we provide the set of hyperparameters to reproduce the numbers obtained for the first iteration of our algorithm across all datasets and bag sizes in Table 9, Table 10, Table 11, Table 12 and Table 13.

Table 9: The set of hyperparameters for various bag sizes for Adult Dataset for the first iteration.

Bag Size	λ_s	λ_b	λ_a	MLP_{LR}	MLP_{WD}	k	τ	δ_d	T
2048	0.0001	0.0184	0.0001	0.0007	1.00E-12	1	0.03558	1	100
1028	0.0001	0.0576	0.0001	0.0012	1.00E-12	1	0.01	1	100
512	0.003	0.0796	0.0001	0.00033	0.00025	1	0.03	1	100
125	0.0001	0.3422	10	0.00028	0.1	1	0.0216	0.01	100
32	186	0.1659	7.5	0.00014	2.12E-11	17	0.3327	0.6	100
8	0.0001	0.4427	10	0.001	1.00E-12	1	0.3515	1	100

Table 10: The set of hyperparameters for various bag sizes for Marketing Dataset for the first iteration.

Bag Size	λ_s	λ_b	λ_a	MLP_{LR}	MLP_{WD}	k	τ	δ_d	T
2048	1.928	7.6986	0.0136	0.0002	3.70E-07	15	0.5311	0.1489	100
1028	0.0001	0.02227	0.000167	0.00005	0.06129	28	0.03077	0.4963	100
512	0.00287	0.2676	0	0.00053	0.0825	7	0.0815	1	100
125	200	26.058	10	0.0027	0.0007735	28	0.03357	0.01	100
32	200	200	9.661	0.00058	4.54E-12	29	0.0111	1	100
8	4.146	2	0.0001	0.00085	9.20E-11	2	0.2023	1	100

B Extended Related Work

Belief Propagation: Belief Propagation (BP) has been used to compute marginals and find MAP estimates in standard sparse graphical models, like Bayesian networks and Markov random fields [25] by message passing across edges on an appropriate graph. Sum-product BP algorithm is used for

Table 11: The set of hyperparameters for various bag sizes for Criteo Dataset for the first iteration.

Bag Size	λ_s	λ_b	λ_a	MLP_{LR}	MLP_{WD}	k	τ	δ_d	T
128	0.4359	0.2265	0	0.000001	0.0000078	11	0.14294	0.00288	200
32	0.0003	0.2502	9.9885	0.00002368	0.000547	23	0.5335	0.1326	200
8	0.00015	0.1884	9.716	0.00006	0.0009567	29	0.4104	0.9996	200

Table 12: The set of hyperparameters for various bag sizes for CIFAR-S Dataset for the first iteration.

Bag Size	λ_s	λ_b	λ_a	MLP_{LR}	MLP_{WD}	k	τ	δ_d	T
2048	0.057	0.02646	0.000156	0.000589	0.0000018	3	0.02687	0.6339	200
1024	0.00015	0.0175	0.00169	0.0006	0.000025	15	0.1214	0.9697	200
512	34.32	0.0338	0	0.00038	0.000014	1	0.267	0.2614	200
128	0.3674	0.105	0	0.0016	0.00476	9	0.31	0.01823	200
32	12.43	1.2541	10	0.000068	0.000336	4	0.4934	0.3727	200
8	0.0001	0.8555	10	0.0002	0.00001	28	0.1961	0.4421	200

Table 13: The set of hyperparameters for various bag sizes for CIFAR-B Dataset for the first iteration.

Bag Size	λ_s	λ_b	λ_a	MLP_{LR}	MLP_{WD}	k	τ	δ_d	T
2048	0.00043	0.8279	0.0002	0.000001	0.000001	2	0.5936	0.2771	200
1024	0.0001	0.003556	0.00695	0.00032	0.000001	4	0.435	0.6544	200
512	0.1289	0.00754	0	0.0116	0.0011	14	0.4337	0.407	200
128	0.0001	0.016	0	0.00214	0.00002	25	0.4508	0.0001	200
32	0.000192	0.0968	8.8918	0.000096	0.000723	1	0.4294	0.00385	200
8	0.0008	0.099	10	0.0000013	0.000009	1	0.4856	0.2427	200

computing marginals and it is known to converge on trees. It was also extended to polytrees [12]. It is also an effective approximate algorithm on general graphical models [24]. More relevant to our work is the fact that sum product Belief Propagation has found widespread in communication system, where it is used to soft-decode a binary string message from their parity checks as in LDPC (low-density parity-check) codes [29, 10], iterative decoding of turbo codes [19, 16]. In communication codes, parity checks are designed so as to have nice properties on the graphical models they induce. In our problem, the bag levels constraints can be thought of as parity checks but are given and we add additional constraints from covariate information that is also given. We use a public scalabale and efficient implementation `PGMax` [40] of the sum-product message passing algorithm.

Decoding from Pooled Data: Another very relevant area of work learning from pooled data paradigm Scarlett and Cevher [31], El Alaoui et al. [9] where the aim to identify the categorical labels of a large collection of items from histogram information at the bag level. El Alaoui et al. [9] present an approximate message passing algorithm for decoding a discrete signal of categorical variables from several histograms of pooled subsets of data. This line of work is also largely aimed at the regime of very large bags [31] ($\sim O(\frac{n}{\log n})$) and there is no covariate information available. In our problem, bags are constant in size and they are disjoint.

C Limitations and Future Work

There are several unexplored interesting directions that we wish to pick up as future work. Notably, one of the primary ones is to explore alternate energy potentials for the Gibbs distribution other than quadratic terms we use now. Further, it might be of independent interest to further investigate why such a simple proposition like BP works on such a scale efficiently converging to marginals proving highly useful in supervised learning even with 1-NN based covariate information. A complete theoretical understanding behind the success of BP for the target task would be an interesting direction building on the theoretical pointers in the supplement.