LEARNING FROM AGGREGATE LABELS WITH FINITE-SUPPORT LIKELIHOODS

Anonymous authors
Paper under double-blind review

ABSTRACT

Aggregate-supervised learning observes a label for a bag of instances while the instance labels remain hidden. When that label is a sum of finite-support contributions, an instance predictor can be trained by maximizing the probability of the observed total. We present FS-Conv, a likelihood layer that separates the instance predictor, the map from predictions to contribution probabilities, and the convolution backend. Under conditional factorization, discrete convolution computes the exact model-implied aggregate probability mass function. With Bernoulli contributions, the objective reduces to established count likelihoods; categorical contributions support integer-valued sums, and known signs encode cancellation. We compare this objective with mean matching and a Gaussian approximate likelihood. In three-seed digit-sum experiments on MNIST and on SVHN with an ImageNet-pretrained ResNet-18, FS-Conv has lower mean aggregate error and discrete aggregate negative log-likelihood, and higher hidden-digit accuracy, than both alternatives. Random-sign MNIST also favors FS-Conv on the reported means. On feature-defined Criteo click-count bags, FS-Conv is competitive with the evaluated label-proportion baselines. These results favor exact discrete likelihood training on the digit-sum and known-sign tasks; instance recovery still depends on the representation and the adequacy of the factorized model.

1 INTRODUCTION

Many learning problems provide supervision only after hidden instance labels have been combined. A bag may be labeled by a positive-instance count, a signed total, or the sum of integer-valued labels. Unlike classical binary multiple-instance learning, where a positive label indicates that at least one instance is positive, these observations specify an additive quantity. The goal is to learn an instance predictor using bag-level labels for the target task. Pretrained representations may supply additional supervision; we distinguish their use in the experiments.

Mean matching fits observed aggregates to sums of predicted instance expectations (Ardehaly & Culotta, 2017; Dulac-Arnold et al., 2019), without specifying probabilities over possible totals. Likelihood training instead maximizes the probability of the observed total under the aggregate probability mass function (PMF) induced by instance predictions. Existing objectives use this principle for small-bag label proportions (Baručić & Kybic, 2022) and Bernoulli counts (Shukla et al., 2023; Ma et al., 2026); probabilistic models also learn digit classifiers from sum-labeled image pairs (Manhaeve et al., 2018). We organize and evaluate this likelihood construction across additive finite-support labels.

Our contributions are:

1. A common finite-support likelihood interface that separates the instance predictor, contribution-PMF map, and computational backend (Section 3).
2. Task-specific contribution functions and reductions that identify the relationship to Bernoulli count likelihoods. Categorical integer sums use non-Bernoulli contributions, whereas the evaluated known-sign binary task reduces to a shifted count problem (Section 3.4).
3. A comparison of exact discrete likelihood training with mean matching and a Gaussian approximation on fixed-bag digit sums and known-sign counts, complemented by a feature-defined Criteo count task (Section 4).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

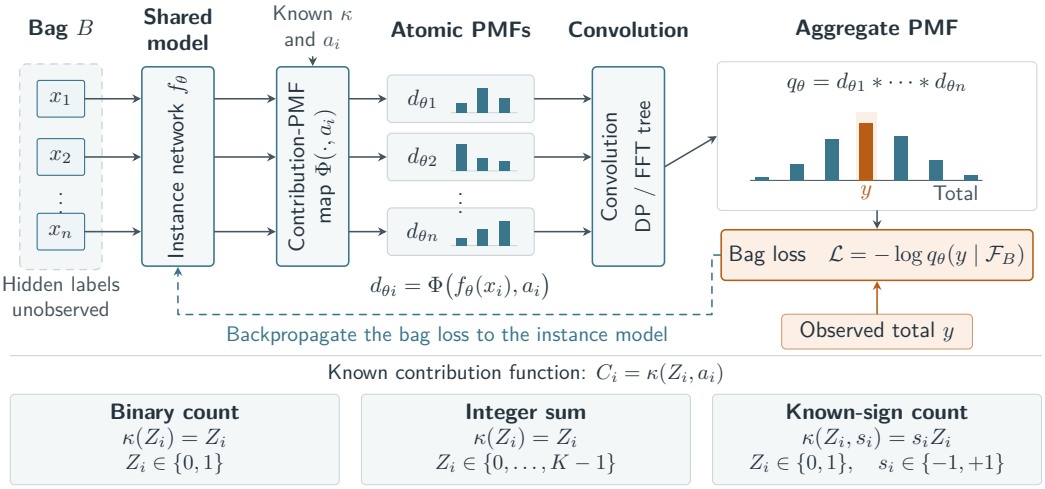


Figure 1: Learning an instance model from an aggregate label. The instance model and contribution-PMF map produce per-instance contribution PMFs. Under conditional factorization given the bag inputs and side information $\mathcal{F}_B$, convolution gives the model-implied aggregate PMF. The loss is the negative log-probability of the observed total y; gradients pass back to f_θ. The lower row shows the three scalar contribution functions. PMF shapes are schematic.

An instance model predicts hidden-state probabilities. The known contribution function induces a map from these probabilities to contribution PMFs. A convolution backend combines them into the distribution of the total. The resulting loss trains the instance model without target-task instance labels (Figure 1). The convolution identity for sums of independent discrete variables is classical; the formulation does not introduce a new convolution rule or exact-inference algorithm.

The digit-sum experiments lead the evaluation because each hidden instance has ten possible values while supervision supplies one total. Across the three vision comparisons, FS-Conv has the lowest reported mean discrete aggregate NLL and highest mean hidden-label accuracy: digit accuracy for sums and binary target accuracy for known-sign counts. The Criteo comparison shows competitive performance without consistent superiority. A signed-count construction in algebraic combinatorics provides additional motivation, but is not evaluated experimentally (Appendix D).

2 Related Work

Multiple-instance learning and count supervision. Classical multiple-instance learning supervises bags rather than instances (Dietterich et al., 1997; Maron & Lozano-Pérez, 1997; Andrews et al., 2002; Carbonneau et al., 2018); neural and attention models learn bag predictors end-to-end (Wang et al., 2018; Ilse et al., 2018). A binary presence label differs from an exact count or integer sum. CountLoss optimizes Bernoulli count probabilities within a count-based weak-supervision framework (Shukla et al., 2023); count supervision also appears in temporal localization (Schroeter et al., 2019).

Learning from label proportions. Learning from label proportions (LLP) uses class proportions or histograms as supervision (Quadrianto et al., 2009; Yu et al., 2014). Approaches include probabilistic models, deep proportion matching, generative methods, and consistency-based training (Sun et al., 2017; Ardehaly & Culotta, 2017; Dulac-Arnold et al., 2019; Liu et al., 2019; Tsai & Lin, 2020). Baručić & Kybic (2022) develop MLE-LLP, which enumerates consistent label assignments for small bags and uses expectation–maximization, and AMLE-LLP, which uses a normal approximation. Our Gaussian-AMLE baseline adapts the latter moment-based objective to general finite-support contributions. LLP-PVC maps instance predictions to count distributions (Ma et al., 2026); Section 3.4 gives the precise relationship between its count log-loss and ours. Feature-defined Criteo bags have precedent in GenBags (Saket et al., 2022) and LLP-Bench (Brahmbhatt et al., 2024); our fixed-size construction is a separate protocol.

Learning digits from sums. DeepProbLog combines neural predicates with probabilistic logic and learns digit classifiers from images supervised by addition results (Manhaeve et al., 2018). These experiments establish that hidden digits can be learned from sum labels. FS-Conv specializes the aggregate computation to additive finite integer contributions and compares the induced discrete likelihood with moment-based losses through a common interface. Our comparison concerns the likelihood formulation and training objectives used for this established task.

Aggregate computation. Dynamic programming, discrete convolution, and polynomial multiplication evaluate the same finite-support sum distribution. CountLoss develops dynamic-programming computation (Shukla et al., 2023); LLP-PVC uses FFT-based pairwise polynomial multiplication in its divide-and-conquer procedure (Ma et al., 2026, Section 3.4 and Algorithm 2). FFT count computation is therefore not a new contribution of this work. Our interface accommodates different execution backends while preserving the likelihood definition.

3 METHOD

3.1 OBSERVED AGGREGATES AND ATOMIC CONTRIBUTIONS

A training example consists of a bag $B = (x_1, \dots, x_n)$, observed side information $a_{1:n}$ when available, and an aggregate label y. Let Z_i be a hidden instance state. A known contribution function κ determines its integer contribution:

$$C_i = \kappa(Z_i, a_i) \in \mathcal{S}_i \subset \mathbb{Z}, \qquad Y = \sum_{i=1}^{n} C_i, \tag{1}$$

where $\mathcal{S}_i$ is finite. For digit sums, $C_i = Z_i \in \{0, \dots, 9\}$. For known-sign counts, $Z_i \in \{0, 1\}$, $a_i = s_i \in \{-1, +1\}$ is observed, and $C_i = s_i Z_i$. A non-target contributes zero and a target contributes its known sign.

Write $\mathcal{F}_B = (x_{1:n}, a_{1:n})$ for the information used to predict the bag label. The instance model f_θ supplies a distribution $p_{\theta i}(z)$ over hidden states. The induced contribution-PMF map Φ gives

$$d_{\theta i} = \Phi\big(f_\theta(x_i), a_i\big) \in \Delta(\mathcal{S}_i), \qquad d_{\theta i}(v) = \sum_{z:\, \kappa(z, a_i) = v} p_{\theta i}(z). \tag{2}$$

Here $\Delta(\mathcal{S}_i)$ is the probability simplex on $\mathcal{S}_i$. We call $d_{\theta i}$ the atomic PMF (or atom): it is a distribution over one instance's contribution, which may differ from its hidden-state distribution. In the evaluated models it depends on $\mathcal{F}_B$ only through (x_i, a_i), so we omit its conditioning argument. The construction does not require iid sampling of raw instances or a fixed bag size, but it does impose the conditional model assumption below.

3.2 EXACT LIKELIHOOD UNDER A FACTORIZED MODEL

Definition 1 (Finite-support convolutional likelihood)**.** *The model factorizes the latent contributions conditional on the bag inputs and side information:*

$$P_\theta(C_{1:n} = c_{1:n} \mid \mathcal{F}_B) = \prod_{i=1}^{n} d_{\theta i}(c_i). \tag{3}$$

FS-Conv maps the atomic PMFs to the aggregate PMF

$$q_\theta(y \mid \mathcal{F}_B) = (d_{\theta 1} * \cdots * d_{\theta n})(y) = \sum_{\substack{c_i \in \mathcal{S}_i \ (i=1,\dots,n) \\ \sum_i c_i = y}} \prod_{i=1}^{n} d_{\theta i}(c_i). \tag{4}$$

Here "exact" means finite-support marginalization without a Gaussian or latent-sampling approximation. If the factorized model is correctly specified, q_θ equals the data-generating conditional aggregate PMF; correct specification is sufficient, not necessary, for agreement at the aggregate level. Regardless of specification, q_θ is the model-implied PMF. The factorization holds before conditioning

on $Y = y$, which generally couples the contributions. Floating-point roundoff, underflow, and probability stabilization can still affect numerical losses and gradients.

For a minibatch of M bags, training minimizes

$$\mathcal{L}_{\text{exact}}(\theta) = -\frac{1}{M}\sum_{b=1}^{M} \log q_\theta(y_b \mid \mathcal{F}_{B_b}). \tag{5}$$

We also use FS-Conv to denote training with $\mathcal{L}_{\text{exact}}$ through this layer; the table rows use this convention. Gradients pass through the aggregate computation into the instance model, without requiring hidden instance labels.

How the total supervises an instance. For a scalar bag, suppress the conditioning on $\mathcal{F}_B$ and write $q_{\theta,-i} = *_{j\neq i} d_{\theta j}$ for the distribution of the other instances' sum. For an observed total with $q_\theta(y) > 0$, the posterior responsibility of contribution $v \in \mathcal{S}_i$ is

$$r_{\theta i}(v \mid y) = P_\theta(C_i = v \mid Y = y, \mathcal{F}_B) = \frac{d_{\theta i}(v)\, q_{\theta,-i}(y - v)}{q_\theta(y)}. \tag{6}$$

The residual-sum probability weights each candidate contribution by the model probability that the other instances sum to $y - v$. For an atom parameterized by softmax logits $g_i(v)$ on its allowed values, $d_{\theta i}(v) = \exp(g_i(v))/\sum_{u\in\mathcal{S}_i}\exp(g_i(u))$. Differentiating the scalar bag loss with respect to these logits, holding the other atoms fixed, gives

$$\frac{\partial[-\log q_\theta(y)]}{\partial g_i(v)} = d_{\theta i}(v) - r_{\theta i}(v \mid y). \tag{7}$$

This negative-log-likelihood gradient compares the current atomic prediction with its posterior responsibility under the observed aggregate. Gradients for shared network parameters, or for other parameterizations of the contribution-PMF map, follow by the chain rule. Training differentiates through convolution and need not explicitly construct every leave-one-out PMF. Appendix C.2 gives the derivation. The posterior can remain diffuse across several compatible values, so this identity explains the supervision signal without guaranteeing recovery of the hidden contributions.

3.3 Mean, Gaussian, and discrete objectives

The same atomic probabilities support three training objectives. The atomic PMFs imply the moments

$$m_i = \sum_{v\in\mathcal{S}_i} v\, d_{\theta i}(v), \qquad \mu_B = \sum_i m_i, \tag{8}$$

$$\sigma_B^2 = \sum_i \left[\sum_{v\in\mathcal{S}_i} v^2 d_{\theta i}(v) - m_i^2\right]. \tag{9}$$

Linearity gives the aggregate mean without independence; the variance sum uses the factorized model. We call squared-error training of the aggregate mean *Mean-MSE*; its per-bag loss is $(y - \mu_B)^2$. The Gaussian-AMLE baseline extends the normal approximation in Baručić & Kybic (2022, Section 2.4) to these finite-support moments and minimizes

$$\mathcal{L}_G = \frac{1}{2}\left[\frac{(y-\mu_B)^2}{\sigma_B^2 + \epsilon} + \log(\sigma_B^2 + \epsilon)\right], \qquad \epsilon > 0. \tag{10}$$

This is a stabilized Gaussian density NLL up to an additive constant. The underlying density does not itself assign a discrete probability mass to each attainable integer total. FS-Conv instead minimizes $-\log q_\theta(y \mid \mathcal{F}_B)$.

Mean-MSE uses the aggregate mean, Gaussian-AMLE uses its first two moments under a Gaussian approximation, and FS-Conv uses the induced discrete PMF. Figure 2 illustrates why these objectives differ: two candidate atomic PMFs induce aggregate distributions with equal means and variances but different probabilities at the observed total. This contrast motivates the empirical comparison. It does not isolate the cause of any recovery gain, because the objectives also differ in support, shape, and optimization.

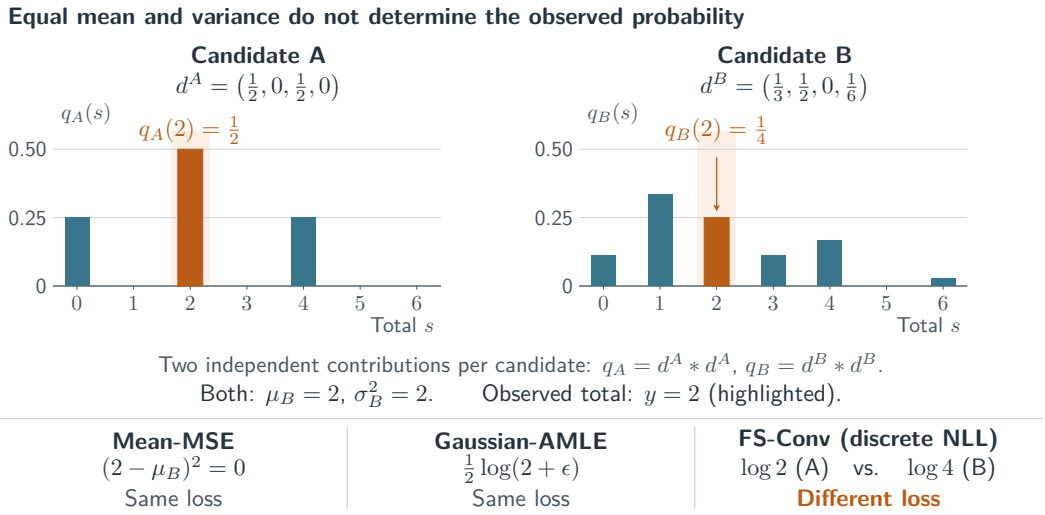


Figure 2: Equal aggregate means and variances need not give equal probabilities at the observed total. Each candidate uses two independent draws from its atomic PMF on $\{0, 1, 2, 3\}$. Both totals have mean 2 and variance 2, but $q_A(2) = 1/2$ and $q_B(2) = 1/4$. At the observed total $y = 2$, Mean-MSE and Gaussian-AMLE give equal losses for the two candidates, while the discrete NLL distinguishes them. The Gaussian loss omits a common additive constant.

3.4 Task-specific contributions and count-likelihood connections

Table 1 gives the scalar contribution functions and atomic PMFs. Here $p_i = p_{\theta i}(1)$ is a binary target probability and $p_{ik} = p_{\theta i}(k)$ a categorical probability for value $k \in \{0, \ldots, K-1\}$. The allowed values of a contribution need not occupy its entire storage grid.

Table 1: Contribution functions and atomic PMFs for the scalar tasks. PMF vectors follow the storage-grid order. For a fixed observed sign, the allowed set is $\{0, s_i\}$; the three-entry signed representation includes a structural zero.

Task	C_i	Allowed set $\mathcal{S}_i$	Storage grid	Atomic PMF
Binary count	Z_i	$\{0, 1\}$	$\{0, 1\}$	$[1 - p_i, p_i]$
Known-sign count	$s_i Z_i$	$\{0, s_i\}$	$\{-1, 0, +1\}$	$s_i = +1$: $[0, 1 - p_i, p_i]$ $s_i = -1$: $[p_i, 1 - p_i, 0]$
Integer sum	Z_i	$\{0, \ldots, K-1\}$	$\{0, \ldots, K-1\}$	$[p_{i0}, \ldots, p_{i,K-1}]$

Bernoulli counts. For $d_{\theta i} = [1 - p_i, p_i]$ on $\{0, 1\}$,

$$q_\theta(y \mid \mathcal{F}_B) = [t^y] \prod_i \big((1 - p_i) + p_i t\big), \tag{11}$$

where $[t^y]P(t)$ denotes the coefficient of t^y in $P(t)$. This is the Poisson-binomial count probability. For matched instance probabilities and observed counts, it is the count-likelihood component of CountLoss (Shukla et al., 2023) and the corresponding cross-entropy component of LLP-PVC (Ma et al., 2026). Equality of complete losses and gradients additionally requires matched reductions, probability handling, and parameterization. A full LLP-PVC pipeline comparison is not included; the equivalence concerns these likelihood components.

Known-sign counts. For $C_i = s_i Z_i$, one positive and one negative target can produce the same total as no targets. The evaluated ± 1 case has a simple count reduction. Let n_- be the number of negative signs and set

$$W_i = \begin{cases} Z_i, & s_i = +1, \\ 1 - Z_i, & s_i = -1, \end{cases} \qquad \pi_i = \begin{cases} p_i, & s_i = +1, \\ 1 - p_i, & s_i = -1. \end{cases} \tag{12}$$

Then $Y_{\pm} = \sum_i W_i - n_-$, so

$$q_\theta(y \mid \mathcal{F}_B) = [t^{y+n_-}] \prod_i \big((1-\pi_i) + \pi_i t\big). \tag{13}$$

An ordinary count backend can therefore evaluate the signed likelihood by complementing probabilities and shifting the target. Its model moments are

$$\mu_B = \sum_i s_i p_i, \qquad \sigma_B^2 = \sum_i s_i^2 p_i(1-p_i) = \sum_i p_i(1-p_i). \tag{14}$$

Known signs illustrate cancellation and a different contribution function, rather than a new probability family. General integer weights also fit Eq. (1), but are not evaluated here.

Categorical integer sums. With $Z_i \in \{0, \dots, K-1\}$, the atomic PMF is the categorical vector. A sum label can be compatible with many digit assignments, and the likelihood adds their probabilities. These contributions are not generally reducible to one Bernoulli variable per instance. They are the only evaluated case that does not reduce to a Bernoulli count likelihood.

Classwise extension. A classwise histogram variant, not evaluated in the experiments below, assumes categorical classes indexed by $c \in \{1, \dots, C\}$. For conditionally independent instance classes, $H_c = \sum_i \mathbf{1}\{Z_i = c\}$ has the marginal count PMF

$$q_c(h_c \mid \mathcal{F}_B) = [t^{h_c}] \prod_i \big((1-p_{ic}) + p_{ic} t\big), \qquad \mathcal{L}_{\text{comp}} = -\frac{1}{C} \sum_{c=1}^{C} \log q_c(h_c \mid \mathcal{F}_B). \tag{15}$$

This is a marginal composite objective (Varin et al., 2011); the dependent class counts do not yield a full joint histogram likelihood by multiplying these marginals. Appendix C.3 gives the joint law, compatibility conditions, and the binary reduction.

3.5 Computation

Support and storage. The aggregate support is the set of sums of positive-mass atomic values:

$$\operatorname{supp}(q_\theta) = \bigoplus_{i=1}^{n} \operatorname{supp}(d_{\theta i}) \subseteq \bigoplus_{i=1}^{n} \mathcal{S}_i, \tag{16}$$

where $\oplus$ denotes the Minkowski sum. Suppose atom i is stored on an integer grid from $\underline{v}_i$ to $\overline{v}_i$ that contains $\mathcal{S}_i$. The dense convolution then occupies an allocated interval of width $T = 1 + \sum_i (\overline{v}_i - \underline{v}_i)$, including any unreachable zero-mass entries. These storage endpoints need not be the smallest and largest allowed values. A common grid width w gives $T = n(w-1)+1$; both atomic and aggregate storage may be wider than their positive-mass supports.

Recurrence and cost. Initialize $q^{(0)} = \delta_0$, the point mass at zero, and update

$$q^{(i)}(s) = \sum_{v \in \mathcal{S}_i} q^{(i-1)}(s-v) d_{\theta i}(v). \tag{17}$$

Sequential convolution, shifted tensor accumulation, grouped convolution, and correctly padded FFT products evaluate this PMF, subject to floating-point error. For a common atomic grid width w and final span T, sequential accumulation has work bounded by $O(nwT)$. A balanced FFT product tree with transform lengths matched to intermediate spans has the conventional bound $O(T \log T \log n)$. Widely separated contribution values can make dense computation expensive even when each allowed set is small. Classwise evaluation repeats the scalar computation for each of C classes. Appendix C.1 gives pseudocode and implementation details. The experiments use Eq. (17) or its Bernoulli dynamic-programming specialization, not FFT (Appendix A.5).

4 Experiments

4.1 Protocol and metrics

The vision comparisons use fixed bags, shared across methods within each seed. Validation images come from the official training split; the official test split is reserved for final evaluation. Training

losses and checkpoint selection use aggregate supervision. Instance labels are used to construct the aggregates and stratify the MNIST image split, but hidden-instance metrics are computed only at final evaluation. Tables report mean $\pm$ sample standard deviation over seeds $0, 1, 2$. The image partitions are fixed across seeds, and SVHN bag sets overlap almost completely; these summaries do not measure variation across independent data splits.

MNIST (LeCun et al., 1998) uses 1,000 training bags and a randomly initialized convolutional network. SVHN (Netzer et al., 2011) uses 5,000 training bags and an ImageNet-1K-pretrained ResNet-18 (He et al., 2016), with all parameters fine-tuned under sum supervision. Bag lengths are rounded normal draws with mean 10 and standard deviation 2, clipped below at 1 for MNIST and to $[5, 15]$ for SVHN. Gaussian-AMLE uses $\epsilon = 10^{-4}$ in Eq. (10). Its checkpoints minimize Gaussian integer-bin NLL on MNIST and Gaussian density loss on SVHN; Mean-MSE selects by MAE and MSE, respectively. FS-Conv selects by discrete NLL on both. These comparisons reflect both the training objectives and the checkpoint-selection criteria in Table 6. Appendix A specifies the splits, bag construction, architectures, optimization, and numerical implementation.

Metrics. Expected-sum MAE averages $|\mu_{B_b} - y_b|$ over test bags; signed MAE uses the signed expectation. Discrete aggregate NLL scores every trained model using its convolved PMF, with observed-total probability floored at 10^{-12}. It is distinct from the Gaussian density training loss and Gaussian-bin score. Hidden accuracy uses direct digit argmax or the binary decision $p_i \geq 0.5$. MNIST counts every valid test-bag occurrence, including repeats; SVHN counts each distinct image appearing in its test bags once. Criteo ROC-AUC covers the selected test impressions. None of these instance predictions conditions on the observed test aggregate.

Discrete NLL measures held-out fit under FS-Conv's training objective; it alone does not establish instance recovery. MAE evaluates aggregate point predictions, while hidden accuracy and ROC-AUC assess instance predictions. We interpret these metrics jointly.

4.2 DIGIT SUMS

Digit-sum supervision observes $Y = \sum_i Z_i$, with hidden digit values $Z_i \in \{0, \ldots, 9\}$. Table 2 compares the three objectives. On MNIST, Gaussian-AMLE improves the mean of each metric over Mean-MSE, and FS-Conv improves them further. Thus the difference between the exact and Gaussian objectives appears in both aggregate prediction and hidden-digit recovery.

Table 2: Digit sums: mean $\pm$ sample standard deviation over three seeds. Discrete NLL evaluates each model's induced aggregate PMF. MNIST uses 1,000 training bags; SVHN uses 5,000 and a fine-tuned ImageNet-pretrained ResNet-18. Accuracy follows the dataset-specific populations in Section 4.1.

Method	Expected-sum MAE ↓	Discrete NLL ↓	Hidden digit accuracy ↑
MNIST digit sums			
Mean-MSE	2.615 ± 0.081	3.251 ± 0.024	0.199 ± 0.008
Gaussian-AMLE	2.147 ± 0.169	3.019 ± 0.416	0.616 ± 0.074
FS-Conv	1.498 ± 0.366	1.658 ± 0.727	0.878 ± 0.083
SVHN digit sums with a pretrained ResNet-18			
Mean-MSE	2.165 ± 0.149	2.350 ± 0.239	0.781 ± 0.079
Gaussian-AMLE	2.342 ± 0.248	2.204 ± 0.340	0.875 ± 0.054
FS-Conv	1.983 ± 0.359	1.777 ± 0.465	0.916 ± 0.049

SVHN has the same mean ordering in discrete NLL and hidden-digit accuracy. Gaussian-AMLE has worse expected-sum MAE than Mean-MSE, however, so the objectives do not have a uniform ordering across metrics. FS-Conv has the best mean on all three metrics, with substantial across-seed variation. These summaries are not paired significance tests. Per seed (Table 7), FS-Conv has the highest hidden-digit accuracy and lowest discrete NLL in all three seeds on both datasets; its SVHN expected-sum MAE is lower than Gaussian-AMLE's in every seed but lower than Mean-MSE's only for seed 0. These results favor the evaluated discrete objective over this Gaussian approximation without establishing which difference between the losses causes the recovery gains.

4.3 KNOWN-SIGN COUNTS

The random-sign MNIST task uses $Z_i = \mathbf{1}\{\text{digit}_i = 9\}$ and observes $Y_\pm = \sum_i s_i Z_i$. Signs are independent, equiprobable ± 1 draws, independent of the digits; they enter the contribution function, not the image classifier. Sampling preserves the natural class imbalance: digit 9 accounts for 10.1–10.5% of test-bag occurrences across seeds. Positive and negative target contributions can cancel.

Table 3: Random-sign MNIST: mean $\pm$ sample standard deviation over three seeds. MAE uses the expected signed total. Hidden target accuracy concerns binary target recognition, not ten-class digit classification.

Method	Expected signed MAE ↓	Discrete NLL ↓	Hidden target accuracy ↑
Mean-MSE	0.354 ± 0.292	3.093 ± 4.736	0.957 ± 0.054
Gaussian-AMLE	0.198 ± 0.046	0.398 ± 0.091	0.984 ± 0.004
FS-Conv	0.110 ± 0.005	0.237 ± 0.015	0.992 ± 0.001

FS-Conv has the lowest expected signed MAE and discrete NLL and the highest hidden-target accuracy in each seed (Table 7). Its mean accuracy is 0.992, versus 0.984 for Gaussian-AMLE; predicting no targets would give 0.895–0.899. Mean-MSE's large NLL spread comes from seed 0 (NLL 8.56), whose accuracy matches that seed's negative-class rate (0.895), consistent with predicting no targets.

4.4 FEATURE-DEFINED CRITEO CLICK COUNTS

We use the Criteo Display Advertising Challenge data (Criteo, 2014) through the public RecZoo `Criteo_x1` partitions used in BARS (Zhu et al., 2022), with bags formed within each partition. Records with the same values of the anonymized categorical features C4 and C11 are ordered deterministically without using labels and divided into disjoint bags of 128; incomplete bags are discarded. Each seed uses 12,000 training, 3,000 validation, and 5,000 test bags. Training observes only each bag's total click count. All methods share an embedding–MLP predictor; categorical vocabulary sizing uses feature IDs from all three partitions without click labels (Appendix A.4). This construction follows the use of feature-defined Criteo bags in LLP (Saket et al., 2022; Brahmbhatt et al., 2024), but does not reproduce those studies' bag protocols.

The DLLP-BCE baseline applies binary cross-entropy to the observed click proportion $\rho_B = y/n$ and the mean instance probability $\bar{p}_B = n^{-1} \sum_i p_i$:

$$\mathcal{L}_{\text{DLLP}} = -\rho_B \log \bar{p}_B - (1 - \rho_B) \log(1 - \bar{p}_B). \tag{18}$$

This is the binary proportion-matching objective (Ardehaly & Culotta, 2017), rather than the complete co-training procedure. Mean-MSE uses squared count error, $(\sum_i p_i - y)^2$. The GenBags-style baseline trains on weighted linear combinations of bag-count equations, adapting the generalized-bag principle (Saket et al., 2022); it is not a reproduction of the full published pipeline. FS-Conv uses the Bernoulli count likelihood in Eq. (11).

Table 4: Criteo feature-defined bags of 128 impressions: mean $\pm$ sample standard deviation over three seeds. Discrete count NLL uses the model-implied Bernoulli count PMF for each trained instance predictor.

Method	Instance ROC-AUC ↑	Discrete count NLL ↓
DLLP-BCE	0.7522 ± 0.0011	3.2845 ± 0.0121
Mean-MSE	0.7472 ± 0.0012	3.2975 ± 0.0288
GenBags-style	0.7463 ± 0.0012	3.4804 ± 0.1521
FS-Conv	0.7491 ± 0.0004	3.2838 ± 0.0356

FS-Conv has lower mean ROC-AUC than DLLP-BCE and higher mean ROC-AUC than Mean-MSE and GenBags-style (Table 4). This ROC-AUC ordering holds in each seed (Table 8). Its discrete count NLL is numerically close to DLLP-BCE's. These results show competitive performance, without

establishing statistical equivalence or consistent superiority. Unobserved group-level effects may leave labels dependent even conditional on the instance features, but this experiment does not vary that dependence in a controlled way.

5 Discussion and conclusion

FS-Conv has higher hidden-digit accuracy than the evaluated moment objectives in every seed, on MNIST and on SVHN with an ImageNet-pretrained ResNet-18; its mean expected-sum MAE is also lowest, although on SVHN the advantage over Mean-MSE comes from one of three seeds. The SVHN comparison combines supervised ImageNet pretraining with aggregate-only fine-tuning; it does not isolate the effect of pretraining. In the known-sign task, FS-Conv is best on every metric in each seed, although accuracies should be read against a 0.895–0.899 majority-class rate; Criteo shows competitive performance without consistent superiority.

Dependence and recovery. The likelihood is exact for the factorized model. For a general conditional distribution,

$$\mathrm{Var}(Y \mid \mathcal{F}_B) = \sum_i \mathrm{Var}(C_i \mid \mathcal{F}_B) + 2 \sum_{i<j} \mathrm{Cov}(C_i, C_j \mid \mathcal{F}_B). \tag{19}$$

Even with correct atomic marginals, nonzero conditional covariance can make the model-implied aggregate variance differ from the data-generating variance. A misspecified likelihood can also distort the learned atomic probabilities. Our experiments do not measure robustness as a function of dependence strength.

Ambiguity within one bag does not by itself establish non-identifiability of an instance predictor across many bags. Recovery depends on bag composition, representation quality, and the hypothesis class. Hidden-label accuracies measure recovery in the reported settings; they do not establish identifiability in general. The unevaluated classwise extension has an additional limitation: its composite objective omits cross-class count dependence.

The evaluation covers one contribution type beyond Bernoulli count reductions—digit sums—and three seeds per setting. It does not establish broader gains across finite-support tasks. FS-Conv supplies a common likelihood interface whose value should be judged jointly by aggregate prediction, distributional fit, and hidden-label recovery. Distributional fit must be read in light of the factorized model and objective alignment; recovery must also account for any supervision used in pretraining.

AI Use Statement

We used ChatGPT/Codex to assist with manuscript restructuring and editing, literature identification and checking, inspection of source code and saved experiment records, and drafting and reviewing mathematical explanations. We also used these tools to construct the analytical example in Figure 2, create the TikZ figures, and prepare and compile the LaTeX source. The reported experimental values come from existing experiment records.

References

Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems*, 2002.

Ehsan Mohammady Ardehaly and Aron Culotta. Co-training for demographic classification using deep learning from label proportions. In *IEEE International Conference on Data Mining Workshops*, pp. 1017–1024, 2017. doi: 10.1109/ICDMW.2017.144. URL https://arxiv.org/abs/1709.04108.

Denis Baručić and Jan Kybic. Fast learning from label proportions with small bags. In *IEEE International Conference on Image Processing*, pp. 3156–3160, 2022. doi: 10.1109/ICIP46576.2022.9897895. URL https://arxiv.org/abs/2110.03426.

Anand Brahmbhatt, Mohith Pokala, Rishi Saket, and Aravindan Raghuveer. LLP-Bench: A large scale tabular benchmark for learning from label proportions. In *Proceedings of the 33rd ACM*

International Conference on Information and Knowledge Management, pp. 4374–4381, 2024. doi: 10.1145/3627673.3680032. URL https://doi.org/10.1145/3627673.3680032.

Marc-André Carbonneau, Veronika Cheplygina, Eric Granger, and Ghyslain Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77: 329–353, 2018.

Criteo. Criteo display advertising challenge dataset. Kaggle, 2014. URL https://www.kaggle.com/c/criteo-display-ad-challenge/data.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1–2):31–71, 1997.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90. URL https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

Maximilian Ilse, Jakub M. Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International Conference on Machine Learning*, pp. 2127–2136, 2018.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. URL https://bottou.org/papers/lecun-98h.

Jiabin Liu, Bo Wang, Zhiquan Qi, Yingjie Tian, and Yong Shi. Learning from label proportions with generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2019.

Tianhao Ma, Wei Wang, Ximing Li, Gang Niu, and Masashi Sugiyama. Learning from label proportions via proportional value classification. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=JkFBc9anLi.

I. G. Macdonald. *Symmetric Functions and Hall Polynomials*. The Clarendon Press, Oxford University Press, New York, second edition, 1995.

Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. DeepProbLog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/dc5d637ed5e62c36ecb73b654b05ba2a-Abstract.html.

Oded Maron and Tomás Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems*, volume 10, 1997. URL https://proceedings.neurips.cc/paper/1997/hash/82965d4ed8150294d4330ace00821d77-Abstract.html.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. URL https://ai.stanford.edu/~twangcat/papers/nips2011_housenumbers.pdf.

Novi Quadrianto, Alex J. Smola, Tibério S. Caetano, and Quoc V. Le. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10:2349–2374, 2009.

Rishi Saket, Aravindan Raghuveer, and Balaraman Ravindran. On combining bags to better learn from label proportions. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 5913–5927, 2022. URL https://proceedings.mlr.press/v151/saket22a.html.

Julien Schroeter, Kirill Sidorov, and David Marshall. Weakly-supervised temporal localization via occurrence count learning. In *International Conference on Machine Learning*, pp. 5649–5659, 2019.

Vinay Shukla, Zhe Zeng, Kareem Ahmed, and Guy Van den Broeck. A unified approach to count-based weakly supervised learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 38709–38722, 2023.

Richard P. Stanley. Some combinatorial properties of Jack symmetric functions. *Advances in Mathematics*, 77:76–115, 1989.

Tao Sun, Dan Sheldon, and Brendan O'Connor. A probabilistic approach for learning with label proportions applied to the US presidential election. In *IEEE International Conference on Data Mining*, pp. 445–454, 2017.

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

Cristiano Varin, Nancy Reid, and David Firth. An overview of composite likelihood methods. *Statistica Sinica*, 21(1):5–42, 2011. URL `https://www3.stat.sinica.edu.tw/statistica/oldpdf/A21n11.pdf`.

Xinggang Wang, Yongluan Yan, Peng Tang, Xiang Bai, and Wenyu Liu. Revisiting multiple instance neural networks. *Pattern Recognition*, 74:15–24, 2018.

Felix X. Yu, Krzysztof Choromanski, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. On learning from label proportions. arXiv preprint arXiv:1402.5902, 2014. URL `https://arxiv.org/abs/1402.5902`.

Jieming Zhu, Quanyu Dai, Liangcai Su, Rong Ma, Jinyang Liu, Guohao Cai, Xi Xiao, and Rui Zhang. BARS: Towards open benchmarking for recommender systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022. doi: 10.1145/3477495.3531723.

A Experimental details

A.1 Training and checkpoint selection

Table 5 lists the optimization settings. All models use Adam with a constant learning rate. MNIST and SVHN run for their full epoch budgets and then reload the selected checkpoint; there is no early stopping. Criteo stops after three consecutive validation evaluations without improvement, up to 12 epochs. Methods within a task and seed share the same bag sets. Labels are used offline to construct aggregate targets; training losses do not use individual target labels.

Table 5: Training settings. Batch size counts bags. The Criteo epoch budget is a maximum; its gradient clip is an ℓ_2-norm bound.

Setting	MNIST sum / signed	SVHN sum	Criteo count
Train / val / test bags	1,000 / 1,000 / 1,000	5,000 / 600 / 600	12,000 / 3,000 / 5,000
Instance model	LeNet-style CNN	ResNet-18	Embedding–MLP
Initialization	Random	ImageNet-1K supervised	Random
Learning rate	5×10^{-4}	10^{-4}	10^{-3}
Weight decay	10^{-4}	5×10^{-4}	10^{-6}
Batch size	192	16	32
Epoch budget	50	40	12
Gradient clipping	None	None	Norm 5

The vision checkpoint criteria differ by task (Table 6). Each criterion is averaged over validation bags and minimized; an exact tie retains the earlier checkpoint. Gaussian-bin NLL uses the Gaussian probability of $[y - \frac{1}{2}, y + \frac{1}{2}]$, whereas Gaussian density loss is Eq. (10). Both differ from the discrete

NLL reported in the result tables. Criteo selects by validation expected-count MAE for every method, breaking an exact MAE tie by the native validation objective. That objective is averaged across validation minibatches; MAE weights bags equally. Exact ties on both quantities retain the earlier checkpoint. Hidden-instance metrics are not evaluated during selection.

Table 6: Vision checkpoint criteria, all minimized. Mean-MSE trains on squared error in both tasks.

Method	MNIST sum / signed	SVHN sum
Mean-MSE	Expected-total MAE	Expected-total MSE
Gaussian-AMLE	Gaussian integer-bin NLL	Gaussian density loss
FS-Conv	Discrete aggregate NLL	Discrete aggregate NLL

A.2 MNIST DIGIT SUMS AND KNOWN-SIGN COUNTS

The official MNIST training images are split by a digit-stratified shuffle into 51,000 training and 9,000 validation images, using split seed 314159. The official test images form the final evaluation pool. This image partition is fixed across run seeds. For each bag, its length is a rounded normal draw with mean 10 and standard deviation 2, clipped below at 1 with no upper cap. Image indices are sampled uniformly with replacement from the designated pool, so images can repeat within and across bags. Bag-generation seeds are s, $s + 20{,}000$, and $s + 10{,}000$ for training, validation, and test, respectively, where $s \in \{0, 1, 2\}$. The resulting manifests are fixed throughout training and paired across methods.

The digit-sum target adds the digit values. For signed counts, the hidden target is $Z_i = \mathbf{1}\{\text{digit}_i = 9\}$ and signs are independent equiprobable ± 1 draws, independent of image labels. Only the contribution computation receives the signs. The training image pool contains 5,057 digit-9 images out of 51,000; sampling is not balanced between positive and negative targets. The three signed test manifests contain 1,055/10,025, 1,025/9,962, and 1,002/9,897 positive/total occurrences.

Images are grayscale values divided by 255, without augmentation. The network has 5×5 convolutions with 20 and 50 channels, each followed by ReLU and 2×2 max pooling, then a 500-unit ReLU layer and a final output layer. Digit sums use ten softmax outputs; signed counts use one sigmoid output. Each run starts from a fresh random initialization. Hidden accuracy is direct argmax classification for digits, or $p_i \geq 0.5$ for the digit-9 indicator, with equal weight for every valid test-bag occurrence. An exact argmax tie selects the lowest class index.

A.3 SVHN DIGIT SUMS

The instance network is torchvision ResNet-18 initialized with the supervised ImageNet-1K weights `IMAGENET1K_V1`; its classifier is replaced by a randomly initialized ten-output linear layer. All parameters are fine-tuned, including the backbone and BatchNorm affine parameters. BatchNorm running statistics update during training and are fixed for validation and testing. There is no separate supervised SVHN pretraining stage.

A CPU permutation of the official SVHN training images, with split seed 1729, assigns its first $\mathrm{round}(0.15N)$ images to validation and the remainder to training; this split is not stratified. Test bags use the official test images. Bag lengths are rounded normal draws with mean 10 and standard deviation 2, clipped to $[5, 15]$. Images are sampled uniformly with replacement. SVHN label 10 is mapped to digit 0. The fixed budgets are given in Table 5.

For run seed s, bag index j uses generator seed $s + j$ in training, $s + 20{,}000 + j$ in validation, and $s + 10{,}000 + j$ in testing. Thus adjacent run seeds produce shifted bag sequences: 4,999 of 5,000 training bags and 599 of 600 validation/test bags coincide after a one-position shift. Augmentation is also deterministic for each training bag and repeats across epochs. Across-seed standard deviations therefore describe these runs with strongly overlapping bags and a fixed image split, rather than independently resampled datasets.

RGB values are divided by 255. Training augmentation applies reflection padding of four pixels, a random 32×32 crop, and brightness and contrast factors drawn uniformly from $[0.8, 1.2)$, followed by clipping to $[0, 1]$; there is no horizontal flip. Contrast uses the crop's spatial mean separately for

each channel. The network wrapper normalizes by ImageNet channel means $(0.485, 0.456, 0.406)$ and standard deviations $(0.229, 0.224, 0.225)$, then resizes to 224×224 by bilinear interpolation. Validation and test images receive no augmentation. Only valid bag images are forwarded through the backbone, excluding padded entries from BatchNorm.

Table 2 uses argmax accuracy over the distinct images appearing in each seed's test bags. There are 5,344, 5,341, and 5,340 such images for seeds 0, 1, and 2, respectively, compared with 5,959, 5,954, and 5,953 bag occurrences. Each distinct image is scored once without conditioning on a bag total. This differs from the occurrence-weighted MNIST metric and from accuracy over the full official SVHN test set. Aggregate MAE and NLL weight the 600 test bags equally.

A.4 Criteo preprocessing and baselines

We use the preprocessed RecZoo `Criteo_x1` release at revision `4ecadde5eb8473e6ed6c570d9fb57e71079c0ef4`. Its supplied training, validation, and test partitions contain 33,003,326, 8,250,124, and 4,587,167 rows, respectively. Numeric fields I1–I13 and categorical IDs C1–C26 are used as supplied, without further normalization or imputation. Each partition is grouped separately by the values of C4 and C11.

Rows receive their original within-partition row IDs. Within a group, an MD5 key built from the protocol identifier, partition, seed, category pair, and row ID orders the rows before chunking into complete bags of 128. A second MD5 key using the chunk index in place of the row ID orders candidate bags; the first requested number is selected without replacement. A final MD5 ordering fixes positions within each bag. The protocol identifier is `criteo_x1_c4_c11_k128_strictv1`. All ordering and selection keys are label-blind. Bags are disjoint within a partition and seed; different seeds may select overlapping impressions. All methods use the same selected bags within each seed. ROC-AUC pools the 640,000 selected test impressions per seed, each once.

The predictor uses one shared categorical embedding table with dimension 8. The 26 retrieved embeddings are concatenated with 13 numerical inputs, giving 221 inputs to an MLP with hidden widths 256 and 128, ReLU activations, and one sigmoid output. There is no dropout or BatchNorm. Vocabulary layout and cardinality are determined from the selected train, validation, and test feature IDs, without their labels; this is not a vocabulary fitted only on training data. The table cardinality is 2,086,971 for seeds 0 and 1 and 2,086,974 for seed 2.

DLLP-BCE uses Eq. (18), with $\bar{p}_B$ clamped to $[10^{-7}, 1 - 10^{-7}]$. Mean-MSE uses squared count residuals. For the GenBags-style adaptation, let $e_G \in \mathbb{R}^4$ contain the count residuals $\sum_{i \in B} p_i - y_B$ of four consecutive bags in the shuffled minibatch. For each group G, draw $R = 60$ weight vectors independently from $\mathcal{N}(0, \Sigma)$ and minimize

$$\mathcal{L}_{\text{mix}} = \frac{1}{|\mathcal{G}|R} \sum_{G \in \mathcal{G}} \sum_{r=1}^{R} (\omega_{Gr}^{\mathsf{T}} e_G)^2, \qquad \Sigma = 1.331 I_4 - 0.33\, \mathbf{1}_4 \mathbf{1}_4^{\mathsf{T}}. \tag{20}$$

The covariance has diagonal 1.001 and off-diagonal -0.33. Weights may be negative and are not normalized; residuals stay in count units. A batch of 32 bags gives eight groups and 480 weighted equations. Weight sampling uses the run-seeded torch RNG. Validation uses the deterministic expectation $e_G^{\mathsf{T}} \Sigma e_G$, averaged over groups. This Gaussian mixing rule specifies our adaptation of GenBags; it is not a reproduction of its full published optimization procedure.

A.5 Numerical computation and reporting

The vision experiments use sequential probability-domain convolution through Eq. (17); Criteo uses the Bernoulli dynamic-programming specialization. These runs do not use FFT or grouped-convolution backends. The active data, model, atomic-PMF, and aggregate-PMF paths use float32 tensors without automatic mixed precision. Hardware arithmetic modes such as TF32 were not recorded.

Padded positions are point masses at zero. With maximum padded bag length L, digit sums allocate $9L + 1$ entries for $0, \ldots, 9L$; signed counts allocate $2L + 1$ entries for $-L, \ldots, L$. The convolution is not renormalized or clipped. Only the observed-total probability is clamped below at 10^{-12} before its negative natural logarithm is taken. Consequently a per-bag discrete NLL cannot exceed

$-\log(10^{-12}) \simeq 27.63$, and probabilities below that floor receive zero gradient through the clamp. The likelihood and gradient identities in Section 3 describe the unstabilized mathematical objective.

Gaussian-AMLE clamps each numerically computed atomic variance to be nonnegative, sums across valid instances, and adds 10^{-4} to the nonnegative aggregate variance. MNIST checkpoint selection evaluates Gaussian-bin probabilities with a stable log-CDF difference; SVHN selects the density objective directly. Training losses average over bags, without bag-size normalization. Aggregate evaluation weights bags equally; the instance populations are specified above. Each reported mean and sample standard deviation is computed from the three per-seed metrics, not from pooled predictions across seeds.

B PER-SEED RESULTS

Tables 7 and 8 give the saved per-seed results underlying the main tables, rounded here to six decimal places. Epoch denotes the selected checkpoint. Main-text means and sample standard deviations are computed before rounding. For signed counts, the Mean-MSE seed-0 NLL of 8.562513 accounts for its large across-seed standard deviation.

Table 7: Per-seed vision results. Accuracy counts test-bag occurrences for MNIST and distinct images appearing in test bags for SVHN. MAE uses the expected total; NLL uses the induced discrete PMF.

Method	Seed	Epoch	MAE	Discrete NLL	Hidden accuracy
MNIST digit sums					
Mean-MSE	0	48	2.695354	3.224757	0.206493
	1	50	2.616704	3.272275	0.191494
	2	50	2.534325	3.254484	0.198116
Gaussian-AMLE	0	50	2.045907	3.356718	0.576690
	1	43	2.342756	3.144692	0.570506
	2	50	2.052812	2.554122	0.701172
FS-Conv	0	50	1.837276	2.178009	0.812072
	1	50	1.109694	0.827117	0.970685
	2	50	1.548276	1.968973	0.850987
SVHN digit sums					
Mean-MSE	0	37	2.215139	2.559467	0.690120
	1	37	1.996825	2.089663	0.838045
	2	39	2.282136	2.401619	0.813483
Gaussian-AMLE	0	25	2.331007	2.283961	0.884731
	1	8	2.099766	1.831198	0.923797
	2	3	2.595134	2.496962	0.817603
FS-Conv	0	5	1.609204	1.417255	0.948728
	1	4	2.016701	1.612485	0.938776
	2	3	2.324465	2.302078	0.859925
Random-sign MNIST					
Mean-MSE	0	11	0.691005	8.562513	0.894763
	1	49	0.191603	0.383117	0.986449
	2	50	0.178275	0.334616	0.989795
Gaussian-AMLE	0	50	0.180704	0.345424	0.985935
	1	27	0.249925	0.503725	0.979321
	2	44	0.163821	0.345974	0.986359
FS-Conv	0	49	0.115120	0.220124	0.992319
	1	50	0.109308	0.244999	0.991166
	2	50	0.106252	0.245499	0.992220

Table 8: Per-seed Criteo results, evaluated on 5,000 disjoint test bags of 128 impressions.

Method	Seed	Epoch	Instance ROC-AUC	Discrete count NLL
DLLP-BCE	0	12	0.752207	3.288903
	1	11	0.753291	3.293689
	2	12	0.750994	3.270780
Mean-MSE	0	10	0.747415	3.265156
	1	9	0.748186	3.306896
	2	9	0.745910	3.320476
GenBags-style	0	12	0.747495	3.349223
	1	5	0.745175	3.647131
	2	12	0.746344	3.444829
FS-Conv	0	11	0.749336	3.261973
	1	11	0.749291	3.264525
	2	11	0.748586	3.324833

C Implementation and derivations

C.1 Aggregate computation

Algorithm 1 summarizes the scalar and classwise losses. Padding or masked positions use deterministic zero-contribution PMFs: $[1, 0]$ for binary counts, $[0, 1, 0]$ on the signed storage grid, and $[1, 0, \ldots, 0]$ for nonnegative integer values. They do not enlarge the positive-mass support.

Algorithm 1 Aggregate loss for a minibatch

Input: M bags with aggregate labels, instance model f_θ, contribution-PMF map Φ, scalar convolution backend Conv.

1. For each bag, compute instance scores and atomic probabilities using f_θ and Φ.
2. For scalar supervision, compute $q_b = \mathrm{Conv}(d_{b1}, \ldots, d_{bn_b})$ and $\ell_b = -\log q_b(y_b)$.
3. Alternatively, for classwise supervision, compute $q_{bc} = \mathrm{Conv}(\{[1 - p_{bic}, p_{bic}]\}_i)$ for each class and $\ell_b = -C^{-1} \sum_c \log q_{bc}(h_{bc})$.
4. Return $M^{-1} \sum_b \ell_b$ and differentiate through the selected computation.

For compatible categorical predictions with $C = 2$, the classwise loss reduces to the binary count NLL (Appendix C.3).

The sequential backend accumulates shifted copies of the current PMF. For grouped signed convolution, a mathematical PMF on $[-1, 0, +1]$ is stored in the reversed orientation required by a cross-correlation operator such as `conv1d`:

$$[p_i \mathbf{1}\{s_i = +1\},\ 1 - p_i,\ p_i \mathbf{1}\{s_i = -1\}]. \tag{21}$$

This changes the stored kernel orientation, not the contribution probabilities in Table 1. A signed buffer must include every reachable partial sum. FFT products require enough zero padding for linear rather than circular convolution. These are alternative implementations of the same convolution. The experiments use the sequential backends and probability handling specified in Appendix A.5.

C.2 Learning-signal derivation

Fix a bag and suppress $\mathcal{F}_B$. Under Eq. (3), separating instance i from the convolution gives

$$q_\theta(y) = \sum_{u \in \mathcal{S}_i} d_{\theta i}(u)\, q_{\theta,-i}(y - u). \tag{22}$$

For a one-instance bag, the empty convolution is $q_{\theta,-i} = \delta_0$. When $q_\theta(y) > 0$, Bayes' rule yields Eq. (6); Eq. (22) also shows that the responsibilities sum to one over $\mathcal{S}_i$.

For a softmax parameterization on $\mathcal{S}_i$, the atomic Jacobian is

$$\frac{\partial d_{\theta i}(u)}{\partial g_i(v)} = d_{\theta i}(u)\big(\mathbf{1}\{u = v\} - d_{\theta i}(v)\big). \tag{23}$$

Holding all other atomic vectors fixed and substituting into Eq. (22) gives

$$\frac{\partial q_\theta(y)}{\partial g_i(v)} = d_{\theta i}(v)\left[q_{\theta,-i}(y-v) - q_\theta(y)\right], \tag{24}$$

$$\frac{\partial[-\log q_\theta(y)]}{\partial g_i(v)} = d_{\theta i}(v) - \frac{d_{\theta i}(v) q_{\theta,-i}(y-v)}{q_\theta(y)}. \tag{25}$$

The last expression is Eq. (7). This is a per-bag derivative at the atomic logits. Shared model parameters receive the sum of the corresponding chain-rule contributions, with the minibatch factor $1/M$ from Eq. (5). Structural zeros outside an atom's allowed values are not softmax coordinates. If κ maps several latent states to the same contribution, differentiation also passes through the induced PMF map Φ. The identity characterizes the scalar likelihood gradient; it does not require an explicit posterior-estimation step in the training algorithm.

C.3 Classwise versus joint likelihood

The classwise construction assumes $p_i = (p_{i1}, \ldots, p_{iC}) \in \Delta(\{1, \ldots, C\})$. The counts satisfy $\sum_c H_c = n$ and are dependent. Their marginal product, or its geometric mean, is generally not a normalized joint PMF over histograms.

For $C = 2$, categorical compatibility gives $p_{i2} = 1 - p_{i1}$ and $h_2 = n - h_1$, hence $q_2(h_2) = q_1(h_1)$. The class-averaged loss in Eq. (15) therefore reduces to the binary count NLL; the unaveraged product instead squares the count probability. Equation (7) is a scalar atomic-logit identity: applying it to a shared categorical model with a classwise objective also requires summing the per-class chain-rule terms.

For compatible categorical instance probabilities and conditionally independent instance classes, the full histogram law is

$$P_\theta(H = h \mid \mathcal{F}_B) = [t_1^{h_1} \cdots t_C^{h_C}] \prod_i \left(\sum_{c=1}^{C} p_{ic} t_c \right), \qquad \sum_c h_c = n. \tag{26}$$

This is a Poisson-multinomial distribution on a multidimensional support. For binary labels and small bags, MLE-LLP evaluates the corresponding likelihood by enumerating consistent label assignments within EM (Baručić & Kybic, 2022). The classwise construction uses C one-dimensional marginals instead of computing Eq. (26). If one-vs-rest outputs are not compatible categorical probabilities, their scalar count objectives can still be formed, but Eq. (26) and the categorical binary reduction do not follow without an additional compatibility condition.

D Motivating signed-count construction

The signed formulation was motivated in part by a factorization problem for (q, t)-Littlewood–Richardson coefficients in the theory of Macdonald polynomials (Macdonald, 1995). Related factorization questions for Jack symmetric functions are studied by Stanley (1989). This appendix records the connection to additive observations. It is motivation and future scope, not an evaluated application or a proof of the motivating factorization.

A partition λ has a Young diagram. For a cell u, its arm length $a_\lambda(u)$ counts cells to its right in the same row, and its leg length $\ell_\lambda(u)$ counts cells below it in the same column (Figure 3). Define

$$b(a, \ell) = \frac{1 - q^a t^{\ell+1}}{1 - q^{a+1} t^\ell}, \qquad b_\lambda(u) = b(a_\lambda(u), \ell_\lambda(u)). \tag{27}$$

The motivating representation seeks numerator and denominator products of these factors associated with selected cells of partitions λ, μ, ν. Assign $s_u = +1$ to numerator cells and $s_u = -1$ to denominator cells, and let $Z_u \in \{0, 1\}$ indicate selection. Cells with the same arm–leg pair contribute the same factor. If $\mathcal{B}_{a,\ell}$ is the corresponding group of cells, a selected product has the form

$$\prod_u b(a_u, \ell_u)^{s_u Z_u} = \prod_{a,\ell} b(a, \ell)^{\beta_{a,\ell}}, \qquad \beta_{a,\ell} = \sum_{u \in \mathcal{B}_{a,\ell}} s_u Z_u. \tag{28}$$

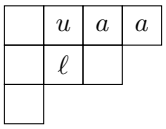


Figure 3: Young diagram of $(4, 3, 1)$. The cell $u = (1, 2)$ has arm length 2 and leg length 1. The marked cells illustrate the two counts.

Given aggregate exponents in such a representation, the observed exponent is a signed total, the sign is known, and the cell selections are hidden. Different selections can produce the same exponents, so this observation does not alone identify a unique selection. The groups are defined by the mathematical object rather than by an iid bag sampler.

If each contribution is uniquely fixed by information actually supplied to a model, its conditional distribution is a point mass and factorization holds trivially. That conditional observation does not establish that a particular cell model has sufficient context, that a factorization is unique, or that learning recovers the intended selection. Problem-specific architectures, the factorization questions, and empirical diagnostics remain future work.