

Appendix Organization

Our appendix is organized as follows:

- Appendix A introduces notions related to (robust) optimal transport and discusses the relationship between our notation and the notation used in the relevant NSL literature.
- Appendix B provides the proofs of all the formal statements in Section 3 and a more detailed discussion of our error bounds.
- Appendix C provides the proof of statistical consistency of Algorithm 1 and discusses other technical aspects related to Algorithm 1.
- Appendix D discusses a nonlinear program formulation of NESY. In addition, it presents in detail the steps to derive the optimization objective in (5), as well as an example of (5) in the context of Example 1.1.
- Appendix E presents an extended version of the related work.
- Appendix F provides further details on our empirical analysis and presents results on more benchmarks.
- Tables 6 and 7 summarize our notation.

A Extended Preliminaries

Optimal transport. Let Z_1 and Z_2 be two discrete random variables over $[m_1]$ and $[m_2]$. For $i \in [2]$, vector $\mathbf{b}^i \in \mathbb{R}_+^{m_i}$ denotes the probability distribution of Z_i , i.e., $\mathbb{P}(Z_i = m_j) = b_j^i$, for each $j \in [m_i]$. Let U be the set of matrices defined as $\{\mathbf{Q} \in \mathbb{R}_+^{m_1 \times m_2} \mid \mathbf{Q}\mathbf{1}_{m_1} = \mathbf{b}^2, \mathbf{Q}\mathbf{1}_{m_2} = \mathbf{b}^1\}$. The *optimal transport* (OT) problem [44] asks us to find the matrix $\mathbf{Q} \in U$ that maximizes a linear object subject to marginal constraints, namely

$$\min_{\mathbf{Q} \in U} \langle \mathbf{P}, \mathbf{Q} \rangle \quad (7)$$

Assume that we are strict in enforcing the probability distribution \mathbf{b}^1 , but not in enforcing \mathbf{b}^2 . The *robust semi-constrained optimal transport* (RSOT) problem [23] aims to find:

$$\min_{\mathbf{Q} \in U'} \langle \mathbf{P}, \mathbf{Q} \rangle + \tau \text{KL}(\mathbf{Q}\mathbf{1}_{m_1} \parallel \mathbf{b}^2) \quad (8)$$

where $U' = \{\mathbf{Q} \in \mathbb{R}_+^{m_1 \times m_2} \mid \mathbf{Q}\mathbf{1}_{m_2} = \mathbf{b}^1\}$ and $\tau > 0$ is a regularization parameter. The solution to (8) can be approximated in polynomial time using *robust semi-Sinkhorn* from [23], which generalizes the classical Sinkhorn algorithm [9] for OT.

Other NESY notation. We now show that our notation for NESY samples is equivalent to the notation adopted by previous works on the topic [33, 18, 36].

Let \mathcal{K} be a background logical theory that “sits” on top of f , i.e., it reasons over the predictions of f . In practice, we may have one or more classifiers, f_1, \dots, f_N , each with its own input and output domains. To simplify the description, we focus on the single-classifier case. However, both our notation and the notation in [33, 18, 36] can be trivially extended to support these scenarios.

In previous works, NESY training samples may be denoted by (\mathbf{x}, ϕ) , where \mathbf{x} is a set of elements from \mathcal{X} and ϕ is a logical sentence (or a single target fact in the simplest scenario). The gold labels of the input instances are unknown to the learner. Instead, we only know that the gold labels of the elements in \mathbf{x} satisfy the logical sentence ϕ subject to \mathcal{K} . The sentence ϕ and the logical theory \mathcal{K} allow us to “guess” what the gold labels of the elements in \mathbf{x} might be so that ϕ is logically satisfied subject to \mathcal{K} . This is essentially the process of *abduction* [55]. To align with the terminology in our paper, for a training sample (\mathbf{x}, ϕ) , we use the term *pre-image*¹ to denote a combination of labels of the elements in \mathbf{x} , such that ϕ is logically satisfied subject to \mathcal{K} . The gold pre-image is the one mapping each instance to its gold label. Abduction allows us to “get rid of” ϕ and \mathcal{K} and represent each training sample via \mathbf{x} and its corresponding pre-images, i.e., as $(\mathbf{x}, \{\sigma_i\}_{i=1}^\omega)$, where each pre-image σ_i is a mapping from \mathbf{x} into labels in \mathcal{Y} . By assuming a canonical ordering on the elements in \mathbf{x} , we can view each $\sigma_i(\mathbf{x})$ as a vector of labels, one for each element in \mathbf{x} . Therefore,

¹Pre-images correspond to *proofs* in [55, 18, 10, 33].

we can equivalently see each training sample as a tuple of the form $(\mathbf{x}, \{\sigma_i(\mathbf{x})\}_{i=1}^\omega)$, supporting our claim that the two notations are equivalent.

B Proofs and Details on Section 3

B.1 Proofs

Proposition 3.1 (Class-specific risk bound). *For any $j \in \mathcal{Y}$, we have that $R_j(f) \leq \Phi_{\sigma,j}(R_P(f; \sigma))$.*

Proof. This result follows directly from the definition of the program (2). \square

Proposition 3.3. *Let $d_{[\mathcal{F}]}$ be the Natarajan dimension of $[\mathcal{F}]$. Given a confidence level $\delta \in (0, 1)$, we have that $R_j(f) \leq \Phi_{\sigma,j}(\tilde{R}_P(f; \sigma, \mathcal{T}_P, \delta))$ with probability $1 - \delta$ for any $j \in [c]$, where*

$$\tilde{R}_P(f; \sigma, \mathcal{T}_P, \delta) = \hat{R}_P(f; \sigma, \mathcal{T}_P) + \sqrt{\frac{2 \log(\text{em}_P/2d_{[\mathcal{F}]} \log(6Mc^2d_{[\mathcal{F}]}e))}{m_P/2d_{[\mathcal{F}]} \log(6Mc^2d_{[\mathcal{F}]}e)}} + \sqrt{\frac{\log(1/\delta)}{2m_P}} \quad (3)$$

Proof. Let $L_\sigma \circ [\mathcal{F}]$ be the function space that maps a (training) example (\mathbf{x}, s) to its partial loss defined as follows:

$$L_\sigma \circ [\mathcal{F}] := \{(\mathbf{x}, s) \mapsto L_\sigma([f](\mathbf{x}), s) | f \in \mathcal{F}\} \quad (9)$$

The standard generalization bound with VC dimension (see, for example, Corollary 3.19 of [40]) implies that:

$$R_P(f) \leq \hat{R}_P(f; \mathcal{T}_P) + \sqrt{\frac{2 \log(\text{em}_P/d_{\text{VC}}(L_\sigma \circ [\mathcal{F}]))}{m_P/d_{\text{VC}}(L_\sigma \circ [\mathcal{F}])}} + \sqrt{\frac{\log(1/\delta)}{2m_P}} \quad (10)$$

where $d_{\text{VC}}(\cdot)$ is the VC dimension. For simplicity, let $d = d_{\text{VC}}(L_\sigma \circ [\mathcal{F}])$ and $d_{[\mathcal{F}]}$ be the Natarajan dimension of $[\mathcal{F}]$. Using a similar argument as in [61], given any d samples in $\mathcal{X}^M \times \mathcal{O}$ using $[\mathcal{F}]$, we let N be the maximum number of distinct ways to assign label vectors (in \mathcal{Y}^M) to these d samples. Then, the definition of VC-dimension implies that:

$$2^d \leq N \quad (11)$$

On the other hand, these d samples contain Md input instances in \mathcal{X} . By Natarajan's lemma (see, for example, Lemma 29.4 of [50]), we have that:

$$N \leq (Md)^{d_{[\mathcal{F}]}} c^{2d_{[\mathcal{F}]}} \quad (12)$$

Combining (12) with the above equations, it follows that

$$(Md)^{d_{[\mathcal{F}]}} c^{2d_{[\mathcal{F}]}} \geq N \geq 2^d \quad (13)$$

Taking the logarithm on both sides, we have that:

$$d_{[\mathcal{F}]} \log(Md) + 2d_{[\mathcal{F}]} \log c \geq d \log 2 \quad (14)$$

Taking the first-order Taylor series expansion of the logarithm function at the point $6d_{[\mathcal{F}]}$, we have:

$$\log(d) \leq \frac{d}{6d_{[\mathcal{F}]}} + \log(6d_{[\mathcal{F}]}) - 1 \quad (15)$$

Therefore,

$$\begin{aligned} d \log 2 &\leq d_{[\mathcal{F}]} \log d + d_{[\mathcal{F}]} \log M + 2d_{[\mathcal{F}]} \log c \\ &\leq d_{[\mathcal{F}]} \left(\frac{d}{6d_{[\mathcal{F}]}} + \log(6d_{[\mathcal{F}]}) - 1 \right) + d_{[\mathcal{F}]} \log M + 2d_{[\mathcal{F}]} \log c \\ &= \frac{d}{6} + d_{[\mathcal{F}]} \log(6Mc^2d_{[\mathcal{F}]}e) \end{aligned} \quad (16)$$

Rearranging the inequality yields

$$\begin{aligned} d &\leq \frac{d_{[\mathcal{F}]} \log(6Mc^2d_{[\mathcal{F}]}e)}{\log 2 - 1/6} \\ &\leq 2d_{[\mathcal{F}]} \log(6Mc^2d_{[\mathcal{F}]}e) \end{aligned} \quad (17)$$

as claimed. \square

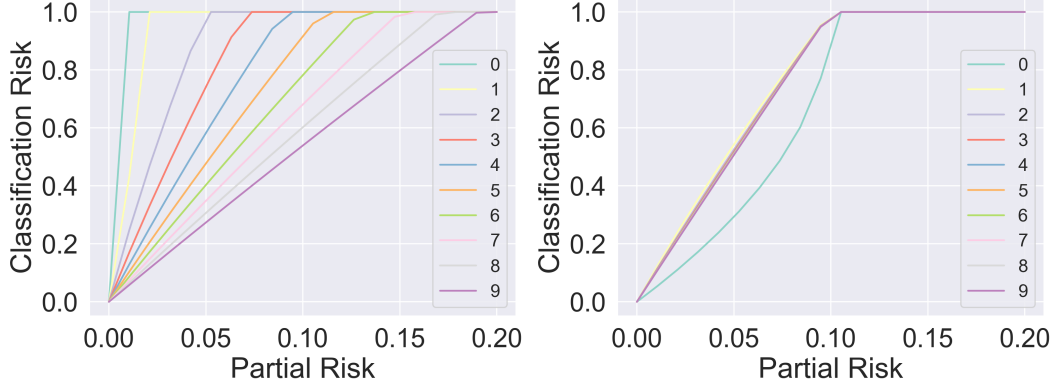


Figure 5: Class-specific upper bounds obtained via (2). (left) \mathcal{D}_Y is uniform. (right) \mathcal{D}_{P_S} is uniform. (Enlarged version of Figure 2).

944 **Proposition 3.4.** *If σ is M -unambiguous, we have*

$$R(f) \leq \sqrt{\mathbf{w}^\top (D(\Sigma_{\sigma, \mathbf{r}}))^\dagger \mathbf{w} R_P(f; \sigma)} = \sqrt{c(c-1) R_P(f; \sigma)} \quad (4)$$

945 *which coincides with Lemma 1 from [61] for $M = 2$, where $\mathbf{w} := \sum_{j=1}^c r_j \mathbf{w}_j$.*

946 *Proof.* Since $\mathbf{w} := \sum_{i=1}^c r_i \mathbf{w}_i$, we have $R(f) = \mathbf{w}^\top \mathbf{h}$. Then, we consider the following relaxed
947 program:

$$\begin{aligned} \max_{\mathbf{h}} \quad & \mathbf{w}^\top \mathbf{h} \\ \text{s.t.} \quad & \mathbf{h}^\top D(\Sigma_{\sigma, \mathbf{r}}) \mathbf{h} \leq R_P \end{aligned} \quad (18)$$

948 where $D(\Sigma_{\sigma, \mathbf{r}})$ is the diagonal part of $\Sigma_{\sigma, \mathbf{r}}$, namely:

$$D(\Sigma_{\sigma, \mathbf{r}}) = [r_i r_j \mathbb{1}\{i = j\} \mathbb{1}\{i \not\equiv j \pmod{c}\}]_{i \in [c^2], j \in [c^2]} \quad (19)$$

949 In other words, $D(\Sigma_{\sigma, \mathbf{r}})$ encodes all the partial risks caused by repeating the same type of mis-
950 classification twice. On the other hand, the M -unambiguity condition ensures that each type of
951 misclassification, when repeated twice, leads to a misclassification of the weak label. Therefore,
952 $\mathbf{w} \in \text{Range}(D(\Sigma_{\sigma, \mathbf{r}}))$.

953 The problem in (18) is a special case of the single-constraint quadratic optimization problem. Then,
954 the fact that $\mathbf{w} \in \text{Range}(D(\Sigma_{\sigma, \mathbf{r}}))$ implies that the dual function of this problem (with dual variable
955 λ) is

$$g(\lambda) = \lambda R_P + \frac{\mathbf{w}^\top (D(\Sigma_{\sigma, \mathbf{r}}))^\dagger \mathbf{w}}{4\lambda} \quad (20)$$

956 where $(D(\Sigma_{\sigma, \mathbf{r}}))^\dagger$ is the pseudo-inverse, namely

$$(D(\Sigma_{\sigma, \mathbf{r}}))^\dagger = [(r_i r_j)^{-1} \mathbb{1}\{i = j\} \mathbb{1}\{i \not\equiv j \pmod{c}\}]_{i \in [c^2], j \in [c^2]} \quad (21)$$

957 Therefore,

$$\mathbf{w}^\top (D(\Sigma_{\sigma, \mathbf{r}}))^\dagger \mathbf{w} = c(c-1) \quad (22)$$

958 According to Appendix B of [1], strong duality holds for this problem. Therefore, the optimal value
959 is given exactly as

$$\inf_{\lambda \geq 0} g(\lambda) = 2\sqrt{\frac{c(c-1)}{4} R_P} = \sqrt{c(c-1) R_P} \quad (23)$$

960 as claimed. \square

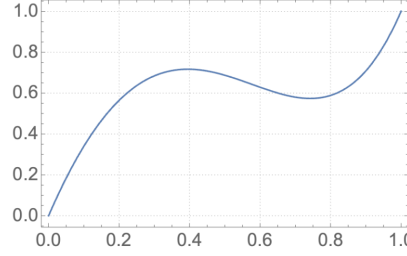


Figure 6: Plot of function $t \mapsto t^4 + 6t^2(1-t)^2 + 4t(1-t)^3$.

961 B.2 Further Discussion on the Proposed Risk Bounds

962 Intuitively, the difficulty of learning is affected by (i) the distribution of weak labels in \mathbf{D}_P and (ii)
 963 the size of the pre-image of σ for each weak label. These two factors are reflected in our risk-specific
 964 bounds. Let us continue with the analysis in Example 3.2.

965 **Example B.1** (Cont’ Example 3.2). *Let us start with CASE 1. In this case, our class-specific error*
 966 *bounds suggest that learning the zero class is more difficult than learning nine class, despite the*
 967 *fact that both hidden labels y_1 and y_2 are uniform in $\{0, \dots, 9\}$, see the left side of Figure B.2. The*
 968 *root cause of this learning imbalance is σ and its characteristics. In particular, the weak labels*
 969 *that result after independently drawing pairs of MNIST digits and applying σ on their gold labels*
 970 *are long-tailed, with $s = 0$ occurring with probability $1/100$ and $s = 9$ occurring with probability*
 971 *$17/100$ in the training data. Hence, we have more supervision to learn class nine than to learn zero.*

972 *Now, let us focus on CASE 2. In this case, our class-specific bounds suggest that learning class zero*
 973 *is the easiest to learn – see right side of Figure B.2. This is due to two reasons. First, the weak labels*
 974 *follow the same uniform distribution. Hence, we have the same amount of supervision to learn all*
 975 *classes. Second, the pre-image of σ for different weak labels is very different. Regarding the second*
 976 *reason, the weak label $s = 0$ provides much stronger supervision than the weak label $s = 9$: when*
 977 *$s = 0$, we have direct supervision ($s = 0$ implies $y_1 = y_2 = 0$); in contrast, when $s = 9$ this only*
 978 *means that $y_1 = 9$ and y_2 is any label in $\{0, \dots, 9\}$, or vice versa.*

979 The above shows that σ (i) can lead to imbalances in the weak labels even if the hidden labels are
 980 uniformly distributed and (ii) may provide supervision signals of very different strengths. Hence,
 981 learning in NESY is *inherently imbalanced* due to σ .

982 B.3 Details on Plotting Figure 2

983 In this subsection, we describe the steps we followed to create the plots in Figure 2. We produced
 984 the curves shown in each figure by plotting 20 evenly spaced points within the partial risk interval
 985 $R_P \in [0, 0.2]$. To obtain the value of the classification risk at each point, we solved the optimization
 986 program (2) using the COBYLA optimization algorithm implemented by the `scipy.optimize`
 987 package. To mitigate numerical instability, for each point, we ran the optimization solver ten times
 988 and then dropped invalid results that were not in the range $[0, 1]$. The median of the remaining valid
 989 results was then taken as the solution to (2).

990 C Further details on Algorithm 1

991 **Proof of statistical consistency of Algorithm 1.** The approximation $\hat{\mathbf{r}}$ given by Algorithm 1 can
 992 be viewed as a method to find the maximum likelihood estimation whose consistency is guaranteed
 993 under suitable conditions. The most critical is the invertibility of Ψ_σ . The invertibility is satisfied by
 994 practical transitions as the one in Example 1.1, but may fail to hold for certain transitions even if the
 995 M -unambiguity condition [61] holds. We will provide one such example later in this section.

996 Suppose that the backpropagation step in Algorithm 1 can find the maximum likelihood estimator. For
 997 a real $\epsilon > 0$, let Δ_c^ϵ be the shrunk probability simplex defined as $\Delta_c^\epsilon := \{\mathbf{r} \in \Delta_c | r_j \geq \epsilon \forall j \in [c]\}$.
 998 Let $\hat{\mathbf{r}}_{m_p}^* := \operatorname{argmin}_{\hat{\mathbf{r}} \in \Delta_c^\epsilon} \sum_{j=1}^{c_S} \bar{p}_j \log[\Psi_\sigma(\hat{\mathbf{r}})]_j$ be the maximum likelihood estimation. We have:

999 **Proposition C.1** (Consistency). *If there exists an $\epsilon > 0$, such that $\mathbf{r} \in \Delta_c^\epsilon$ and Ψ_σ is injective in Δ_c^ϵ ,*
1000 *then $\hat{\mathbf{r}}_{m_p}^* \rightarrow \mathbf{r}$ in probability as $m_p \rightarrow \infty$.*

1001 *Proof.* Let $\Delta_{c_S}^{\sigma, \epsilon} := \{\Psi_\sigma(\mathbf{r}) | \mathbf{r} \in \Delta_c^\epsilon\}$ be the image of Ψ_σ on Δ_c^ϵ . The set $\Delta_{c_S}^{\sigma, \epsilon}$ is a compact subset in
1002 \mathbb{R}^{c_S} . For any weak label $a_j \in \mathcal{S}$, let $H(a_j, \mathbf{r}) := -\log([\Psi_\sigma(\mathbf{r})]_j)$ be the point-wise log-likelihood.
1003 The M -unambiguity condition ensures that each coordinate of every vector in $\Delta_{c_S}^{\sigma, \epsilon}$ should be at
1004 least ϵ^M , and hence the function H is bounded on $\Delta_{c_S}^{\sigma, \epsilon}$. By Theorem 1 of [20], this ensures that
1005 $\sum_s H(s, \mathbf{r})$ converges uniformly to $\mathbb{E}_S[H(S, \mathbf{r})]$. According to [58] (Theorem 5.7), the uniform
1006 convergence further ensures that $\Psi_\sigma(\hat{\mathbf{r}}_{m_p}^*) \rightarrow \mathbf{p}$ in probability as $m_p \rightarrow \infty$. Since Ψ_σ is invertible,
1007 this implies that $\hat{\mathbf{r}}_{m_p}^* \rightarrow \mathbf{r}$ in probability. \square

1008 **Counterexample where the invertibility of Ψ_σ does not hold.** Consider the following transition
1009 function for binary labels ($\mathcal{Y} = \{0, 1\}$) and $M = 4$:

$$\sigma(y_1, y_2, y_3, y_4) = \begin{cases} 1, & \sum_{i=1}^4 y_i \in \{1, 2, 4\} \\ 0, & \text{otherwise} \end{cases} \quad (24)$$

1010 The M -unambiguity condition [61] holds since $\sigma(0, 0, 0, 0) \neq \sigma(1, 1, 1, 1)$. On the other hand, the
1011 probability the weak label is one can be expressed as:

$$\mathbb{P}(s = 1) = r_1^4 + 6r_1^2r_0^2 + 4r_1r_0^3 = r_1^4 + 6r_1^2(1 - r_1)^2 + 4r_1(1 - r_1)^3 \quad (25)$$

1012 which is not an injection, see the plot of function $t \mapsto t^4 + 6t^2(1 - t)^2 + 4t(1 - t)^3$ in Figure 6.

1013 D Details on Section 4.2

1014 D.1 A Nonlinear Program Formulation

1015 A straightforward idea that accommodates the requirements set in Section 4.2 is to reformulate (8) by
1016 (i) extending \mathbf{P} (resp. \mathbf{Q}) to a tensor of size $n \times c \times M$ to store the scores (resp. pseudolabels) of
1017 M -ary tuples of instances and (ii) modifying U' so that the product of the combinations of entries in
1018 \mathbf{Q} corresponding to invalid label assignments is forced to zero. However, modifying U' in this way,
1019 we cannot employ Sinkhorn-like techniques as the one in [28], leaving us only with the option to
1020 employ nonlinear² programming techniques to find \mathbf{Q} .

1021 D.2 Deriving the Linear Program in (5)

1022 Let $(x_{\ell,1}, \dots, x_{\ell,M}, s_\ell)$ denote the ℓ -th NESY training sample, where $\ell \in [n]$. To derive the linear
1023 program in (5), we associate each weak label s_ℓ with a DNF formula Φ_ℓ , a process that is standard in
1024 the neurosymbolic literature [65, 55, 18, 61]. To ease the presentation, we describe how to compute
1025 Φ_ℓ . Let $\{\mathbf{y}_{\ell,1}, \dots, \mathbf{y}_{\ell,R_\ell}\}$ be the set of vectors of labels in $\sigma^{-1}(s_\ell)$. We associate each prediction with
1026 a Boolean variable. Namely, let $q_{\ell,i,j}$ be a Boolean variable that becomes true when $x_{\ell,i}$ is assigned
1027 with label $j \in \mathcal{Y}$. Via associating predictions with Boolean variables, each $\mathbf{y}_{\ell,t}$ can be associated with
1028 a conjunction $\varphi_{\ell,t}$ over Boolean variables from $\{q_{\ell,i,j} | i \in [M], j \in [c]\}$. In particular, $q_{\ell,i,j}$ occurs
1029 in $\varphi_{\ell,t}$ only if the i -th label in $\mathbf{y}_{\ell,t}$ is $j \in \mathcal{Y}$. Consequently, the training sample $(x_{\ell,1}, \dots, x_{\ell,M}, s_\ell)$
1030 is associated with the DNF formula $\Phi_\ell = \bigvee_{t=1}^{R_\ell} \varphi_{\ell,t}$ that encodes all vectors of labels in $\sigma^{-1}(s_\ell)$.
1031 We assume a canonical ordering over the variables occurring in $\varphi_{\ell,t}$, using $\varphi_{\ell,t,j}$ to refer to the j -th
1032 variable, and use $|\varphi_{\ell,t}|$ to denote the number of (unique) Boolean variables occurring $\varphi_{\ell,t}$. Based on
1033 the above, we have $\varphi_{\ell,t} = \bigwedge_{k=1}^{|\varphi_{\ell,t}|} \varphi_{\ell,t,k}$.

1034 Similarly to [51], we use the Iverson bracket $[]$ to map Boolean variables to their corresponding
1035 integer ones, e.g., $[q_{\ell,i,j}]$, denotes the integer variable associated with the Boolean variable $q_{\ell,i,j}$.

1036 We are now ready to construct linear program (5). Notice that the solutions of this program capture
1037 the label assignments that abide by σ , i.e., the labels assigned to each $(x_{\ell,1}, \dots, x_{\ell,M})$ should be
1038 either of $\mathbf{y}_{\ell,1}, \dots, \mathbf{y}_{\ell,R_\ell}$. The steps of the construction are (see [51]):

²Nonlinearity comes from the KL term and by enforcing invalid label combinations to have product equal to zero.

- 1039 • (STEP 1) We translate each Φ_ℓ into a CNF formula Φ'_ℓ via the Tseytin transformation [56]
1040 to avoid the exponential blow up of the (brute force) DNF to CNF conversion.
- 1041 • (STEP 2) We add the corresponding linear constraints out of each subformula in Φ'_ℓ .

1042 Given $\Phi_\ell = \bigvee_{t=1}^{R_\ell} \varphi_{\ell,t}$, the Tseytin transformation associates a fresh Boolean variable $\alpha_{\ell,t}$ with each
1043 disjunction $\varphi_{\ell,t}$ in Φ_ℓ and rewrites Φ_ℓ into the following logically equivalent formula:

$$\Phi'_\ell := \underbrace{\bigvee_{t=1}^{R_\ell} \alpha_{\ell,t}}_{\Psi_\ell} \wedge \bigwedge_{t=1}^{R_\ell} (\alpha_{\ell,t} \leftrightarrow \varphi_{\ell,t}) \quad (26)$$

1044 After obtaining Φ'_ℓ , the construction of (5) proceeds as follows. The first inequality that will be
1045 added to (5) comes from formula Ψ_ℓ . In particular, it will be the inequality $\sum_{t=1}^{R_\ell} [\alpha_{\ell,t}] \geq 1$, due to
1046 Constraint (3) from [51]. The next inequalities come from the subformula $\bigwedge_{t=1}^{R_\ell} (\alpha_{\ell,t} \leftrightarrow \varphi_{\ell,t})$ from
1047 (26). The latter can be rewritten to the following two formulas:

$$\alpha_{\ell,t} \rightarrow \bigwedge_{k=1}^{|\varphi_{\ell,t}|} \varphi_{\ell,t,k} \quad (27)$$

$$\bigwedge_{k=1}^{|\varphi_{\ell,t}|} \varphi_{\ell,t,k} \rightarrow \alpha_{\ell,t} \quad (28)$$

1048 According to Constraint (10) from [51], (27) and (28) are associated with the following inequalities:

$$-|\varphi_{\ell,t}|[\alpha_{\ell,t}] + \sum_{k=1}^{|\varphi_{\ell,t}|} [\varphi_{\ell,t,k}] \geq 0 \quad (29)$$

$$- \sum_{k=1}^{|\varphi_{\ell,t}|} [\varphi_{\ell,t,k}] + [\alpha_{\ell,t}] \geq (1 - |\varphi_{\ell,t}|) \quad (30)$$

1049 which will also be added to the linear program.

1050 Lastly, according to Constraint (5) from [51], we have an equality $\sum_{j=1}^c [q_{\ell,i,j}] = 1$, for each $\ell \in [n]$
1051 and $i \in [M]$. The above equality essentially requires the scores of all pseudolabels for a given
1052 instance $x_{\ell,i}$ to sum up to one. Finally, we require each pseudolabel $[q_{\ell,i,j}]$ to be in $[0, 1]$, for each
1053 $\ell \in [n]$, $i \in [M]$, and $j \in [c]$.

1054 Putting everything together, we have the following linear program:

$$\begin{aligned} \text{minimize} \quad & \min_{(\mathbf{Q}_1, \dots, \mathbf{Q}_m)} \sum_{i=1}^M \langle \mathbf{Q}_i, -\log(\mathbf{P}_i) \rangle, \\ \text{subject to} \quad & \begin{aligned} & \sum_{t=1}^{R_\ell} [\alpha_{\ell,t}] \geq 1, & \ell \in [n], \\ & -|\varphi_{\ell,t}|[\alpha_{\ell,t}] + \sum_{k=1}^{|\varphi_{\ell,t}|} [\varphi_{\ell,t,k}] \geq 0, & \ell \in [n], t \in [R_\ell] \\ & - \sum_{k=1}^{|\varphi_{\ell,t}|} [\varphi_{\ell,t,k}] + [\alpha_{\ell,t}] \geq -1(1 - |\varphi_{\ell,t}|), & \ell \in [n], t \in [R_\ell] \\ & \sum_{j=1}^c [q_{\ell,i,j}] = 1, & \ell \in [n], i \in [M] \\ & [q_{\ell,i,j}] \in [0, 1], & \ell \in [n], i \in [M], j \in [c] \end{aligned} \end{aligned} \quad (31)$$

1055 Program (5) results after adding to the above program constraints enforcing the hidden label ratios $\hat{\mathbf{r}}$.

1056 **Example D.1.** We demonstrate an example of (5) in the context of Example 1.1. We assume $n = 2$.
1057 We also assume that the weak labels s_1 and s_2 of the two NESY samples in the batch are equal to 0
1058 and 1, respectively. Due to the properties of the max, we have:

$$\sigma^{-1}(0) = \{(0, 0)\} \quad (32)$$

$$\sigma^{-1}(1) = \{(0, 1), (1, 0), (1, 1)\} \quad (33)$$

1059 and formulas Φ_1 and Φ_2 are defined as:

$$\Phi_1 = \underbrace{q_{1,1,0} \wedge q_{1,2,0}}_{\varphi_{1,1}} \quad (34)$$

$$\Phi_2 = \underbrace{q_{2,1,0} \wedge q_{2,2,1}}_{\varphi_{2,1}} \vee \underbrace{q_{2,1,1} \wedge q_{2,2,0}}_{\varphi_{2,2}} \vee \underbrace{q_{2,1,1} \wedge q_{2,2,1}}_{\varphi_{2,3}} \quad (35)$$

1060 The Tseytin transformation associates the fresh Boolean variables $\alpha_{1,1}$, $\alpha_{2,1}$, $\alpha_{2,2}$, and $\alpha_{2,3}$ to $\varphi_{1,1}$,
 1061 $\varphi_{2,1}$, $\varphi_{2,2}$, and $\varphi_{2,3}$, respectively, and rewrites Φ_1 and Φ_2 to the following logically equivalent
 1062 formulas:

$$\Phi'_1 = \alpha_{1,1} \wedge (\alpha_{1,1} \leftrightarrow \varphi_{1,1}) \quad (36)$$

$$\Phi'_2 = (\alpha_{2,1} \vee \alpha_{2,2} \vee \alpha_{2,3}) \wedge (\alpha_{2,1} \leftrightarrow \varphi_{2,1}) \wedge (\alpha_{2,2} \leftrightarrow \varphi_{2,2}) \wedge (\alpha_{2,3} \leftrightarrow \varphi_{2,3}) \quad (37)$$

1063 The linear constraints that are added due to Φ'_1 are:

$$\begin{aligned} -|\varphi_{1,1}|[\alpha_{1,1}] + [q_{1,1,0}] + [q_{1,2,0}] &\geq 1 \\ -(|\varphi_{1,1}|[\alpha_{1,1}] + [q_{1,1,0}] + [q_{1,2,0}]) + [\alpha_{1,1}] &\geq 0 \\ -(|\varphi_{1,1}|[\alpha_{1,1}] + [q_{1,1,0}] + [q_{1,2,0}]) + [\alpha_{1,1}] &\geq -1(1 - |\varphi_{1,1}|) \end{aligned} \quad (38)$$

1064 The linear constraints that are added due to Φ'_2 are:

$$\begin{aligned} [\alpha_{2,1}] + [\alpha_{2,2}] + [\alpha_{2,3}] &\geq 1 \\ -|\varphi_{2,1}|[\alpha_{2,1}] + [q_{2,1,0}] + [q_{2,2,1}] &\geq 0 \\ -|\varphi_{2,2}|[\alpha_{2,2}] + [q_{2,1,1}] + [q_{2,2,0}] &\geq 0 \\ -|\varphi_{2,3}|[\alpha_{2,3}] + [q_{2,1,1}] + [q_{2,2,1}] &\geq 0 \\ -(|\varphi_{2,1}|[\alpha_{2,1}] + [q_{2,1,0}] + [q_{2,2,1}]) + [\alpha_{2,1}] &\geq -1(1 - |\varphi_{2,1}|) \\ -(|\varphi_{2,2}|[\alpha_{2,2}] + [q_{2,1,1}] + [q_{2,2,0}]) + [\alpha_{2,2}] &\geq -1(1 - |\varphi_{2,2}|) \\ -(|\varphi_{2,3}|[\alpha_{2,3}] + [q_{2,1,1}] + [q_{2,2,1}]) + [\alpha_{2,3}] &\geq -1(1 - |\varphi_{2,3}|) \end{aligned} \quad (39)$$

1065 Finally, the requirement that the pseudolabels for each instance $x_{\ell,i}$ to sum up to one, for $\ell \in [2]$ and
 1066 $i \in [2]$, and to lie in $[0, 1]$ introduces the following linear constraints:

$$\begin{aligned} \sum_{j=0}^9 [q_{1,1,j}] &= 1 \\ \sum_{j=0}^9 [q_{1,2,j}] &= 1 \\ \sum_{j=0}^9 [q_{2,1,j}] &= 1 \\ \sum_{j=0}^9 [q_{2,2,j}] &= 1 \\ [q_{1,i,j}] &\in [0, 1], \quad i \in [2], j \in \{0, \dots, 9\} \\ [q_{2,i,j}] &\in [0, 1], \quad i \in [2], j \in \{0, \dots, 9\} \end{aligned} \quad (40)$$

1067 E Extended Related Work

1068 **NESY.** NESY quite often arises in NSL [32, 63, 10, 66, 55, 34, 18, 25, 19]. However, we are the
 1069 first to study the phenomenon of learning imbalances. Below we discuss some recent theoretical
 1070 results [36, 35, 61]. The work in [36, 35] deals with the problem of characterizing and mitigating
 1071 *reasoning shortcuts*. Intuitively, a reasoning shortcut is a classifier that has small partial risk but high
 1072 classification risk. For example, a reasoning shortcut is a classifier that has good accuracy in the overall
 1073 task of returning the maximum of two MNIST digits, but has low accuracy in classifying MNIST
 1074 digits. The work in [36] showed that current NESY techniques are vulnerable to reasoning shortcuts.
 1075 However, the work does not provide (class-specific) error bounds or any theoretical characterization
 1076 of learning imbalances. The authors in [61] proposed necessary and sufficient conditions that ensure
 1077 learnability of MI-PLL and provided error bounds for a state-of-the-art neurosymbolic loss under
 1078 approximations [18]. Our theoretical analysis extends the analysis in [61] by providing (i) class-
 1079 specific risk bounds (in contrast to [61], which only bounds $R(f)$) and (ii) stricter bounds for $R(f)$.
 1080 In particular, as we show in Proposition 3.4, we can recover the bound from Lemma 1 in [61] by
 1081 relaxing (2).

1082 **Long-tailed learning.** The term *long-tailed learning* has been used to describe settings in which
 1083 instances of some classes occur very frequently in the training set, with other classes being underrep-
 1084 resented. The problem has received considerable attention in supervised learning, with the proposed

techniques operating at training or testing time. Techniques in the former category typically work by reweighting the losses computed using the original training samples [4, 54, 53] or by over- or under-sampling during training [7, 2]. The techniques in the latter category work by modifying the classifiers' scores at testing time and using the modified scores for classification [22, 42], with LA being one of the most well-known techniques [38]. LA modifies the classifier's scores during testing time by subtracting the (unknown) gold ratios. In particular, the prediction of the classifier f given input x is given by $\arg \max_{j \in [c]} f^j(x) - \ln(r_j)$. Our empirical analysis shows that CAROT is more effective than LA.

The most relevant to our work is the study in [42]. Unlike CAROT, the authors in [42] focus on PLL and use an optimal transport formulation [44] to adjust the scores of the classifier assuming that the marginal \mathbf{r} is known. In contrast, CAROT relies on the assumption that $\hat{\mathbf{r}}$ may be noisy, resorting to a robust optimal transport formulation [23] to improve the classification accuracy in these cases.

PLL. In PLL [8, 30, 12], each training sample is a tuple of the form $(x, \{l_1, \dots, l_n\})$, where $x \in \mathcal{X}$ and l_1, \dots, l_n is a set of candidate, mutually exclusive labels for x that includes the gold label of x . Since (1) each NESY training sample is represented as a tuple of the form $\mathbf{x}, \sigma^{-1}(s)$, see Section 2, where each element in $\sigma^{-1}(s)$ is a vector of candidate labels for the elements in \mathbf{x} , (2) the vectors in $\mathbf{x}, \sigma^{-1}(s)$ are mutually exclusive, and (3) $\sigma^{-1}(s)$ includes the gold labels \mathbf{y} for the elements in \mathbf{x} , we can see that PLL reduces NESY (and MI-PLL) when restricting to input vectors of one label only.

The observation that certain classes are harder to learn than others dates back to the work of [8] in the context of PLL. We are the first to provide such results for NESY, also unveiling the relationship between σ and class-specific risks.

Long-tailed PLL. A few recently proposed papers lie in the intersection of long-tailed learning and standard PLL, namely [29], RECORDS [16] and SOLAR [59], with the first one focusing on non-deep learning settings. RECORDS modifies the classifier's scores following the same idea with LA and uses the modified scores for training. However, it uses a momentum-updated prototype feature to estimate $\hat{\mathbf{r}}$. RECORDS's design allows it to be used with any loss function and be trivially extended to support NESY. Our empirical analysis shows that RECORDS is less effective than CAROT, leading to lower classification accuracy when the same loss is adopted during training.

SOLAR shares some similarities with LP. In particular, given single-instance PLL samples of the form $\{(x_1, S_1), \dots, (x_n, S_n)\}$, where each $S_\ell \subseteq \mathcal{Y}$ is the weak label of the ℓ -th PLL sample³, SOLAR finds pseudolabels \mathbf{Q} by solving the following linear program:

$$\begin{aligned} & \min_{\mathbf{Q} \in \Delta} \langle \mathbf{Q}, -\log(\mathbf{P}) \rangle \\ \text{s.t. } & \Delta := \{[q_{\ell,j}]_{n \times c} \mid \mathbf{Q}^\top \mathbf{1}_n = \hat{\mathbf{r}}, \mathbf{Q} \mathbf{1}_c = \mathbf{c}, q_{\ell,j} = 0 \text{ if } j \notin S_\ell\} \subseteq [0, 1]^{n \times c} \end{aligned} \quad (41)$$

The program (41) shows that the information of each weak label S_ℓ is strictly encoded into Δ . To directly extend (41) to NESY, we have two options:

- Use an $n \times c^M$ tensor \mathbf{P} to store the scores of the classifier, where the cell $P[\ell, j_1, \dots, j_c]$ stores the scores of the classifier for the label vector (j_1, \dots, j_c) associated with the ℓ -th training NESY sample, for $1 \leq \ell \leq n$. However, that formulation would require an excessively large tensor, especially when M becomes larger.
- Use separate tensors $\mathbf{P}_1, \dots, \mathbf{P}_M$ to represent the model's scores of the M instances, and set for each $1 \leq \ell \leq n$, the product $P_1[\ell, j_1] \times \dots \times P_M[\ell, j_c]$ to be 0 if (j_1, \dots, j_c) does not belong to $\sigma^{-1}(s_\ell)$. However, that formulation would lead to a non-linear program.

Neither choice is scalable for NESY when M is large⁴. Our work overcomes these issues by translating the information in the weak labels into linear constraints, leading to an LP formulation. Another difference between SOLAR and our work is that we developed Algorithm 1 to estimate the ratios of the hidden labels, while SOLAR employs a window averaging technique to estimate \mathbf{r} based on the model's scores [59]. Finally, although CAROT also uses a linear programming formulation with a Sinkhorn-style procedure, it differs from SOLAR in that it adjusts the classifier's scores at testing time rather than assigning pseudolabels at training time.

³In standard PLL, each weak label is a subset of classes from \mathcal{Y} .

⁴Yet another non-linear formulation is presented in Section D based on RSOT (see Section A).

Listing 1 Theory for the Smallest Parent benchmark.

```
land_transportation :- automobile, truck
other_transportation :- airplane, ship
transportation :- land_transportation, other_transportation
home_land_animal :- cat, dog
wild_land_animal :- deer, horse
land_animal :- home_land_animal, wild_land_animal
other_animal :- bird, frog
animal :- land_animal, other_animal
entity :- transportation, animal
```

1132 **Constrained learning.** NESY is closely related to constrained learning, in the sense that the predicted
1133 label vector \mathbf{y} should adhere to the constraint $\sigma(\mathbf{y}) = s$. Training classifiers under constraints has
1134 been well studied in NLP [52, 45, 43, 39, 57, 62, 14, 37]. The work in [46] proposes a formulation
1135 for training under linear constraints; [49] proposes a Unified Expectation Maximization (UEM)
1136 framework that unifies several techniques, including CoDL [5] and Posterior Regularization [13].
1137 The UEM framework was also adopted by [25] for NSL. Our LP formulation is orthogonal to UEM –
1138 it could be integrated with UEM, though.

1139 The theoretical framework for constrained learning in [60] provides a generalization theory that
1140 suggests that encoding the constraints during both training and testing results in a better model
1141 compared to encoding the constraints only during testing. This theory could be extended to explain
1142 the advantages of LP-based techniques and to characterize the necessary conditions for CAROT to
1143 improve model performance.

1144 **Other weakly supervised settings.** Another well-known weakly supervised learning setting is that
1145 of Multi-Instance Learning (MIL). In MIL, instances are not individually labelled but grouped into
1146 sets that contain at least one positive instance, or only negative instances, and the aim is to learn a
1147 *bag classifier* [48, 47]. In contrast, in NESY, instances are grouped into tuples, with each tuple of
1148 instances being associated with a set of mutually exclusive label vectors, and the aim is to learn an
1149 *instance classifier*.

1150 F Further Experiments and Details

1151 **Why using SL and Scallop.** SL [65, 33] has become the state-of-the-art approach to train deep
1152 classifiers in NSL settings. Training under SL requires computing a Boolean formula ϕ encoding all
1153 the possible label vectors in $\sigma^{-1}(s)$ for each NESY training sample (\mathbf{x}, s) and then computing the
1154 weighted model counting [6] of ϕ given the softmax scores of f . SL has been effective in several tasks,
1155 including visual question answering [18], video-to-text alignment [27], and fine-tuning language
1156 models [26], and has nice theoretical properties [61, 36]. Due to its effectiveness, SL is now adopted
1157 by several NSL engines, such as DeepProbLog [33], DeepProbLog’s successors [34], and Scallop
1158 [18, 27].

1159 Our empirical analysis only uses Scallop, since it is the only engine that provides a scalable SL
1160 implementation that can support our scenarios when $M \geq 3$. The computation of $\sigma^{-1}(s)$ is generally
1161 required by NSL techniques [25, 33, 10, 66]. This computation can become a bottleneck when the
1162 space of candidate label vectors grows exponentially, as in our MAX- M , SUM- M , and HWF- W
1163 scenarios. As also experimentally shown by [55, 61], the NESY techniques from [33, 34, 10, 25, 66]
1164 either time out after several hours while trying to compute $\sigma^{-1}(s)$, or lead to deep classifiers of much
1165 worse accuracy than Scallop. So, Scallop was the only engine that could support our experiments,
1166 balancing runtime with accuracy.

1167 A further discussion about scalability issues in NESY can be found in Sections 3.2 and 6 at [61].

1168 **Additional scenarios.** We additionally carried experiments with two other scenarios that have been
1169 widely used as NESY benchmarks, namely SUM- M [32, 18] and HWF- M [25, 27]. SUM- M is
1170 similar to MAX- M , however, instead of taking the maximum, we take the sum of the gold labels. The
1171 HWF- M scenario⁵ was introduced in [24]. In this scenario, each training sample $((x_1, \dots, x_M), s)$

⁵The benchmark is available at <https://liqing.io/NGS/>.

consists of a sequence (x_1, \dots, x_M) of digits in $\{0, \dots, 9\}$ and mathematical operators in $\{+, -, *\}$, corresponding to a valid mathematical expression, where s is the result of the mathematical expression. As in SUM- M , the goal is to train a classifier to recognize digits and mathematical operators. Notice that this benchmark is not i.i.d. since only specific types of input sequences are valid. The benchmark comes with a list of training samples. However, we created our own samples, to introduce imbalances in the distributions of the digits and operators.

Computational infrastructure. The experiments ran on an 64-bit Ubuntu 22.04.3 LTS machine with Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz, 3.16TB hard disk and an NVIDIA GeForce RTX 2080 Ti GPU with 11264 MiB RAM. We used CUDA version 12.2.

Software packages. Our source code was implemented in Python 3.9. We used the following python libraries: scallopy⁶, highspy⁷, or-tools⁸, PySDD⁹, PyTorch and PyTorch vision. Finally, we used part of the code¹⁰ available at [16] to implement RECORDS and part of the code¹¹ available at [59] to implement the sliding window approximation for marginal estimation.

Classifiers. For MAX- M and SUM- M , we used the MNIST CNN also used in [18, 32]. For HWF- M , we used the CNN also used in [25, 27]. For Smallest Parent, we used the ResNet model also used in [59, 16].

Data generation. To create datasets for MAX- M , Smallest Parent, SUM- M , and HWF- M we adopted the approach followed in previous work [10, 55, 61, 33, 18]. In particular, to create each training sample, we drew instances x_1, \dots, x_M independently by MNIST or CIFAR-10. Then, we applied the function σ over the gold labels y_1, \dots, y_M to obtain the weak label s . To create samples for HWF- M , we followed similar steps to the above. However, to ensure that the input vectors of images represent a valid mathematical expression, we split the training instances into operators and digits, drawing instances of digits for odd i s and instances of operators for even i s, for $i \in [M]$. Before sample creation, the images in HWF were split into training and testing ones with ratio 70%/30%, as the benchmark does not offer such splits. As we state in Section 5, to simulate long-tail phenomena (denoted as **LT**), we vary the imbalance ratio ρ of the distributions of the input instances as in [4, 59]: $\rho = 0$ means that the hidden label distribution is unmodified and balanced. In each scenario, the test data follows the same distribution as the hidden labels in the training NESY data, e.g., when $\rho = 0$, the test data is balanced; otherwise, it is imbalanced under the same ρ .

Further details. For the Smallest Parent scenarios, we computed SL and (5) using the whole pre-image of each weak label. For the MAX- M scenarios, we only consider the top-1 proof [61] both when running Scallop and in (5) as the space of pre-images is very large. For the Smallest Parent benchmark, we created the hierarchical relations shown in Listing 1 based on the classes from CIFAR-10.

Tables and plots. To assess the robustness of our techniques, we focus on scenarios with high imbalances, large number of input instances, and few NESY training samples. Table 2 shows results for SUM- M , for $M \in \{5, 6, 7\}$, $\rho = \{50, 70\}$, and $m_P = 2000$. Table 3 shows results for HWF- M , for $M \in \{5, 6, 7\}$, $\rho = \{15, 50\}$, and $m_P = 250$, while Table 4 shows results for the same experiment, but $m_P = 1000$. In Tables 3 and 4, LP(ALG1) refers to running LP using the gold ratios—Algorithm 1 cannot be applied, as the data is not i.i.d. in this scenario. Tables 3 and 4 focuses on training time mitigation. RECORDS was not considered, as it led to substantially lower accuracy in the MAX- M and Smallest Parent scenarios. Figure 7 shows the marginal estimates computed by Algorithm 1 for different scenarios. Last, Table 5 presents all the results for the MAX- M scenarios. The tables follow the same notation with the ones in the main body of the paper.

Conclusions. The conclusions that we can draw from Table 2, 3, 4, and Figure 7 are very similar to the ones drawn in the main body of our paper. When LP is adopted jointly with the estimates obtained by Algorithm 1, we can see that the accuracy improvements are substantial on multiple occasions. For example, in SUM-6 with $\rho = 50$, the accuracy of classification increases from 67% under SL to 80% under LP(ALG1); in HWF-7 with $\rho = 15$, classification accuracy increases from

⁶<https://github.com/scallop-lang/scallop> (MIT license).

⁷<https://pypi.org/project/highspy/> (MIT license).

⁸<https://developers.google.com/optimization/> (Apache-2.0 license).

⁹<https://pypi.org/project/PySDD/> (Apache-2.0 license).

¹⁰<https://github.com/MediaBrain-SJTU/RECORDS-LTPLL> (MIT license).

¹¹<https://github.com/hbzju/SoLar>.

Table 2: Experimental results for SUM- M using $m_P = 2000$. Results over six runs.

Algorithms	$\mathbf{LT} \rho = 50$			$\mathbf{LT} \rho = 70$		
	$M = 5$	$M = 6$	$M = 7$	$M = 5$	$M = 6$	$M = 7$
SL	82.28 \pm 15.87	67.60 \pm 13.43	68.42 \pm 25.66	75.43 \pm 22.49	79.60 \pm 19.36	69.05 \pm 13.31
+ LA	81.74 \pm 16.27	67.04 \pm 13.27	68.33 \pm 25.61	75.38 \pm 22.58	79.47 \pm 19.49	68.95 \pm 12.91
+ CAROT	82.21 \pm 15.94	68.82 \pm 12.61	69.54 \pm 24.46	76.12 \pm 21.80	80.47 \pm 18.37	66.08 \pm 17.70
LP(EMP)	75.31 \pm 23.49	62.86 \pm 6.97	62.89 \pm 34.47	78.18 \pm 20.74	64.66 \pm 33.95	63.64 \pm 35.32
+ LA	74.94 \pm 23.86	62.36 \pm 6.71	62.55 \pm 34.81	78.11 \pm 20.81	64.02 \pm 34.66	63.08 \pm 35.87
+ CAROT	72.19 \pm 17.50	64.13 \pm 8.37	65.26 \pm 32.24	77.25 \pm 21.48	66.36 \pm 27.43	67.95 \pm 30.85
LP(ALG1)	89.86 \pm 8.54	80.10 \pm 18.45	77.94 \pm 20.72	91.64 \pm 7.62	91.52 \pm 7.24	63.79 \pm 12.97
+ LA	89.72 \pm 8.68	79.43 \pm 19.15	77.61 \pm 21.05	91.66 \pm 7.60	91.52 \pm 7.24	63.70 \pm 12.87
+ CAROT	89.14 \pm 9.16	78.85 \pm 19.55	67.74 \pm 29.69	91.29 \pm 7.86	91.97 \pm 6.80	67.06 \pm 9.78

Table 3: Experimental results for HWF- M using $m_P = 250$. Results over six runs.

Algorithms	$\mathbf{LT} \rho = 15$			$\mathbf{LT} \rho = 50$		
	$M = 3$	$M = 5$	$M = 7$	$M = 3$	$M = 5$	$M = 7$
SL	38.03 \pm 44.91	44.83 \pm 5.22	37.02 \pm 10.89	39.94 \pm 46.83	50.40 \pm 17.31	36.83 \pm 20.94
LP(EMP)	41.66 \pm 23.00	44.16 \pm 7.33	38.66 \pm 6.90	45.56 \pm 39.70	50.29 \pm 25.65	34.38 \pm 16.60
LP(GOLD)	48.31 \pm 26.72	44.72 \pm 6.73	41.06 \pm 8.05	50.73 \pm 34.19	51.63 \pm 14.00	35.55 \pm 15.17

Table 4: Experimental results for HWF- M using $m_P = 1000$. Results over six runs.

Algorithms	$\mathbf{LT} \rho = 15$			$\mathbf{LT} \rho = 50$		
	$M = 3$	$M = 5$	$M = 7$	$M = 3$	$M = 5$	$M = 7$
SL	94.01 \pm 0.49	95.34 \pm 0.14	48.23 \pm 6.91	27.42 \pm 25.62	80.81 \pm 15.36	83.87 \pm 13.00
LP(EMP)	84.27 \pm 10.01	84.86 \pm 10.80	50.90 \pm 12.17	49.26 \pm 45.98	66.44 \pm 19.62	47.04 \pm 8.58
LP(GOLD)	94.39 \pm 0.27	95.72 \pm 0.34	55.73 \pm 6.12	41.09 \pm 52.57	81.28 \pm 14.43	88.85 \pm 27.89

1221 37% under SL to 41% under LP(ALG1). The accuracy under LP(EMP) is lower than the accuracy
1222 under LP(ALG1) in SUM- M . We argue that this is due to the low quality of the empirical estimates
1223 of \mathbf{r} , a phenomenon that gets magnified due to the adopted approximations— recall that we run for SL
1224 and LP using the top-1 proofs, in order to make the computation tractable. The lower accuracy of
1225 LP(ALG1) for SUM-7 and $\rho = 70$ is attributed to the fact that the marginal estimates computed by
1226 Algorithm 1 diverge from the gold ones – see Figure 7. In fact, computing marginals for this scenario
1227 is particularly challenging due to the very large pre-image of σ when $M = 7$, the high imbalance ratio
1228 ($\rho = 70$), and the small number of NESY samples ($m_P = 2000$). Tables 3 and 4 also suggest that
1229 SOLAR’s empirical ratio estimation technique may harm the accuracy of our LP-based formulation,
1230 supporting a claim that we also made in the main body of the paper, namely that *the computation of*
1231 *the marginals for training time mitigation is an important direction for future research.*

1232 Figure 7 shows the robustness of Algorithm 1 in computing marginals. Figure 8 shows the hidden
1233 label ratios and the corresponding class-specific classification accuracies under the MAX- M and the
1234 Smallest Parent scenarios for $\rho = 50$.

Table 5: Experimental results for MAX- M using $m_P = 3000$.

Algorithms	Original $\rho = 0$			$\text{LT } \rho = 5$			$\text{LT } \rho = 15$			$\text{LT } \rho = 50$		
	$M = 3$	$M = 4$	$M = 5$	$M = 3$	$M = 4$	$M = 5$	$M = 3$	$M = 4$	$M = 5$	$M = 3$	$M = 4$	$M = 5$
SL + LA + CAROT	84.15 \pm 11.92	73.82 \pm 2.36	59.88 \pm 5.58	55.48 \pm 23.23	66.24 \pm 1.22	55.13 \pm 4.20	71.25 \pm 4.48	66.98 \pm 3.2	55.06 \pm 5.21	66.74 \pm 5.42	67.71 \pm 11.58	55.74 \pm 2.58
	84.17 \pm 11.95	73.82 \pm 2.36	59.88 \pm 5.58	55.48 \pm 23.23	65.63 \pm 1.75	55.13 \pm 4.20	70.80 \pm 4.52	66.98 \pm 3.20	54.53 \pm 5.74	66.57 \pm 5.09	61.10 \pm 3.95	52.47 \pm 8.06
	84.57 \pm 11.50	73.08 \pm 3.10	60.26 \pm 5.20	56.52 \pm 21.70	66.70 \pm 0.76	55.91 \pm 3.42	74.95 \pm 3.45	67.44 \pm 2.74	55.80 \pm 4.47	68.16 \pm 4.00	68.25 \pm 6.14	57.29 \pm 14.17
RECORDS + LA + CAROT	85.56 \pm 7.25	75.11 \pm 0.77	59.43 \pm 6.61	77.98 \pm 3.13	65.85 \pm 0.62	55.07 \pm 4.24	55.47 \pm 20.45	53.34 \pm 16.66	52.40 \pm 7.95	70.20 \pm 7.65	66.05 \pm 13.90	59.93 \pm 4.86
	87.63 \pm 5.11	75.11 \pm 0.77	59.28 \pm 6.76	77.98 \pm 3.13	65.43 \pm 0.87	54.40 \pm 4.44	54.90 \pm 20.16	54.46 \pm 15.54	51.25 \pm 9.09	70.09 \pm 7.26	65.78 \pm 14.18	59.93 \pm 4.86
	90.97 \pm 2.03	75.94 \pm 0.91	60.45 \pm 7.78	78.31 \pm 4.00	67.57 \pm 1.74	55.46 \pm 3.94	54.32 \pm 21.85	62.74 \pm 8.14	55.85 \pm 4.61	71.46 \pm 6.4	71.25 \pm 8.70	63.64 \pm 5.92
LP(EMP) + LA + CAROT	94.97 \pm 1.32	77.86 \pm 4.22	55.27 \pm 11.27	80.15 \pm 1.69	70.73 \pm 1.85	56.28 \pm 2.03	75.83 \pm 5.26	69.67 \pm 5.47	59.25 \pm 7.27	77.16 \pm 3.46	70.06 \pm 10.73	56.79 \pm 1.58
	94.69 \pm 1.60	77.91 \pm 4.16	55.34 \pm 11.19	80.08 \pm 1.55	70.54 \pm 1.82	55.31 \pm 3.27	75.77 \pm 5.32	68.92 \pm 3.96	58.49 \pm 5.74	77.1 \pm 3.52	69.76 \pm 10.31	56.81 \pm 1.56
	95.07 \pm 1.20	75.53 \pm 7.42	53.07 \pm 12.99	80.29 \pm 2.33	70.88 \pm 2.22	57.85 \pm 4.05	76.38 \pm 4.72	69.74 \pm 5.51	59.56 \pm 8.14	77.58 \pm 3.04	70.11 \pm 10.34	57.09 \pm 1.90
LP(ALG1) + LA + CAROT	96.09 \pm 0.41	78.34 \pm 4.80	59.91 \pm 6.63	78.56 \pm 1.52	69.71 \pm 0.03	57.61 \pm 3.09	74.51 \pm 9.13	69.14 \pm 1.82	56.81 \pm 3.74	72.23 \pm 11.49	69.28 \pm 11.78	63.67 \pm 7.04
	95.81 \pm 0.74	78.97 \pm 4.09	59.98 \pm 6.56	78.48 \pm 1.53	69.71 \pm 0.03	57.47 \pm 3.09	74.26 \pm 9.06	68.73 \pm 2.23	56.37 \pm 3.13	72.23 \pm 11.49	69.21 \pm 11.86	63.67 \pm 7.04
	96.13 \pm 0.38	80.78 \pm 2.36	59.71 \pm 6.35	78.93 \pm 1.85	70.32 \pm 0.86	57.62 \pm 3.08	77.05 \pm 7.00	69.19 \pm 1.81	59.76 \pm 7.24	74.82 \pm 10.18	74.30 \pm 7.54	64.39 \pm 6.43

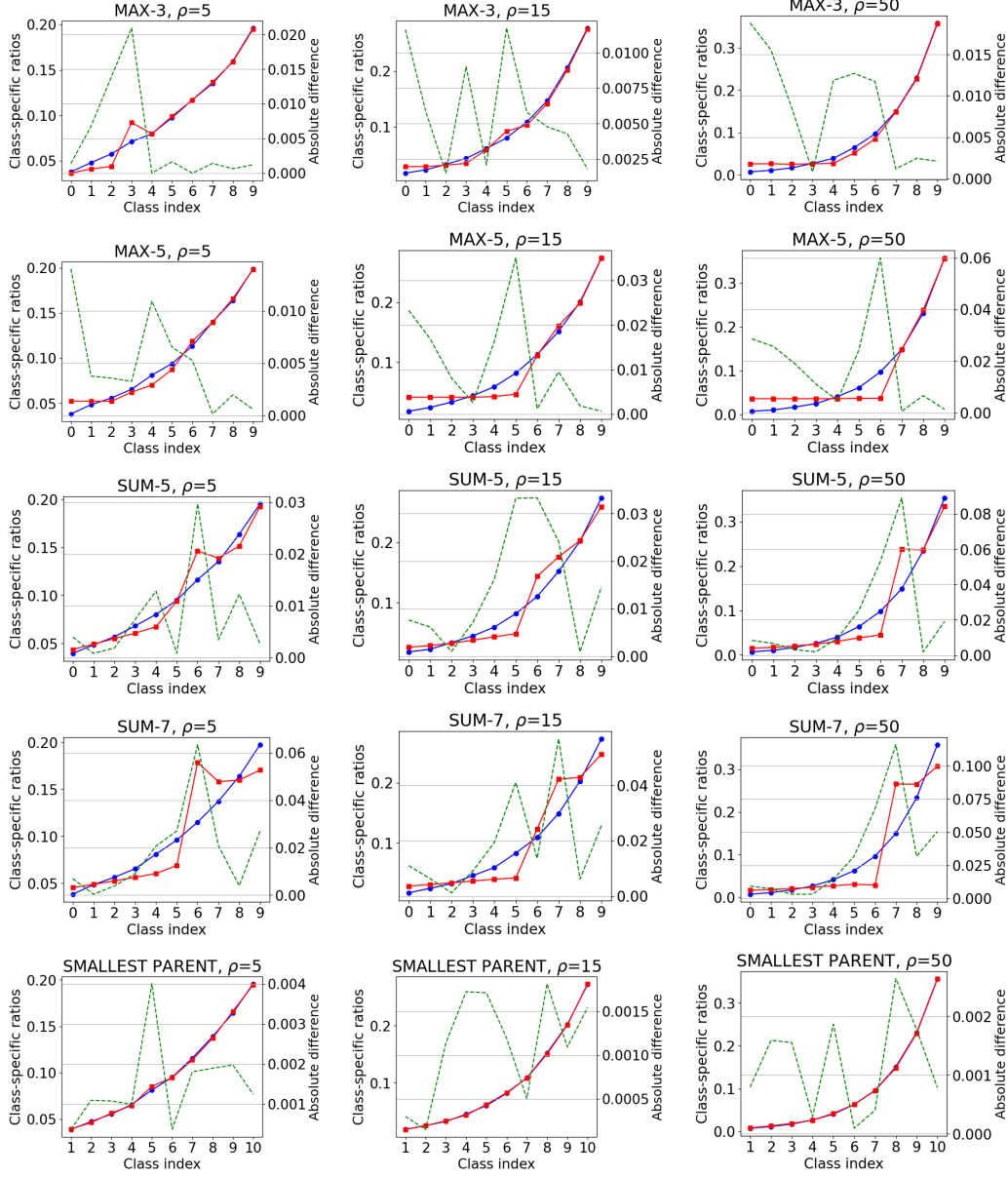


Figure 7: Accuracy of the marginal estimates computed by Algorithm 1 for different scenarios. Blue denotes the gold ratios, red the estimated ones, and green the absolute difference between the gold and estimated ratios.

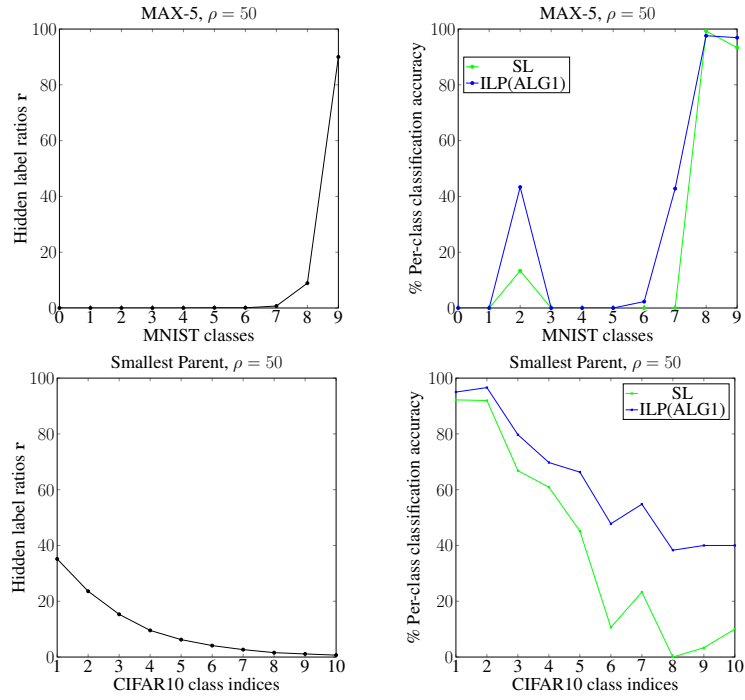


Figure 8: (Up left) hidden label ratios r for MAX-5 with $\rho = 50$. (Up right) Class-specific classification accuracies under SL and ILP(ALG1) for MAX-5 with $\rho = 50$. (Down left) hidden label ratios r for Smallest parent with $\rho = 50$. (Down right) Corresponding class-specific classification accuracies under SL and ILP(ALG1) for Smallest parent with $\rho = 50$.

Table 6: The notation in the preliminaries and the theoretical analysis.

Supervised learning	
$1\{\cdot\}$ $[n] := \{1, \dots, n\}$ $\mathcal{X}, \mathcal{Y} = [c]$ x, y X, Y $\mathcal{D}, \mathcal{D}_X, \mathcal{D}_Y$ $r_j = \mathbb{P}(Y = j)$ $\bar{\mathcal{D}}_Y := \mathbf{r} = (r_1, \dots, r_c)$ Δ_c $f : \mathcal{X} \rightarrow \Delta_c$ $f^j(x)$ $[f] : \mathcal{X} \rightarrow \mathcal{Y}$ $\mathcal{F}, [\mathcal{F}]$ $d_{[\mathcal{F}]}$ $L(y', y) := 1\{y' \neq y\}$ $R(f)$ $R_j(f) := P([f](x) \neq j Y = j)$ $D(\mathbf{A})$	Indicator function Set notation Input instance space and label space Elements from \mathcal{X} and \mathcal{Y} Random variables over \mathcal{X} and \mathcal{Y} Joint distribution of (X, Y) and marginals of X and Y probability of occurrence (or ratio) of label $j \in \mathcal{Y}$ in \mathcal{D} Marginal of Y Space of probability distributions over \mathcal{Y} Scoring function Score of f upon x for class $j \in \mathcal{Y}$ Argmax classifier induced by f Space of scoring functions and corresponding space of classifiers Natarajan dimension of $[\mathcal{F}]$ Zero-one loss given $y, y' \in \mathcal{Y}$ Zero-one risk of f Risk of f for the j -th class in \mathcal{Y} The diagonal matrix that shares the same diagonal with square matrix \mathbf{A}
NESY	
$M > 0$ $\mathbf{x} = (x_1, \dots, x_M), \mathbf{y} = (y_1, \dots, y_M)$ $\mathcal{S} = \{a_1, \dots, a_{c_S}\}$ S $\sigma : \mathcal{Y}^M \rightarrow \mathcal{S}$ $s = \sigma(\mathbf{y})$ $\sigma^{-1}(s)$ (\mathbf{x}, s) \mathcal{D}_P \mathcal{D}_{P_S} \mathcal{T}_P $[f](\mathbf{x})$ $L_\sigma(\mathbf{y}, s) := L(\sigma(\mathbf{y}), s)$ $R_P(f; \sigma) := E_{(X_1, \dots, X_M, S) \sim \mathcal{D}_P} [L_\sigma([f](\mathbf{X}), S)]$ $\hat{R}_P(f; \sigma, \mathcal{T}_P)$	Number of input instances per NESY sample Vector of input instances and their (hidden) gold label Space of c_S weak labels Random variable over \mathcal{S} Symbolic component (known to the learner) Weak label Pre-image of s , i.e., set of all vectors $\mathbf{y} \in \mathcal{Y}^M$ s.t. $\sigma(\mathbf{y}) = s$ NESY sample Distribution of NESY samples over $\mathcal{X}^M \times \mathcal{S}$ Marginal of \mathcal{S} Set of m_P NESY samples Short for $([f](x_1), \dots, [f](x_M))$ Zero-one partial loss subject to σ Zero-one partial risk subject to σ Empirical zero-one partial risk subject to σ given set \mathcal{T}_P of NESY samples
Notation in Section 3	
$\mathbf{1}_n, \mathbf{0}_n$ \mathbf{I}_n \mathbf{e}_j $\mathbf{H}(f)$ $\mathbf{h}(f) := \text{vec}(\mathbf{H}(f))$ $\mathbf{w}_j := \text{vec}(\mathbf{W}_j)$ $\Sigma_{\sigma, \mathbf{r}}$ $\Phi_{\sigma, j}(R_P(f; \sigma))$ $\hat{R}_P(f; \sigma, \mathcal{T}_P, \delta)$	All-one and all-zero vectors Identity matrix of size $n \times n$ c -dimensional one-hot vector, where the j -th element is one $c \times c$ matrix where the (i, j) cell is the probability of f classifying an instance with label $i \in \mathcal{Y}$ to $j \in \mathcal{Y}$. Vectorization of $\mathbf{H}(f)$ Vectorization of matrix $\mathbf{W}_j := (\mathbf{1}_c - \mathbf{e}_j)\mathbf{e}_j^\top$, where $j \in \mathcal{Y}$ Symmetric matrix in $\mathbb{R}^{c^2 \times c^2}$ depending on σ and \mathbf{r} Optimal solution to program (2) and upper bound to $R_j(f)$ Generalization bound of $R_P(f; \sigma)$ for probability $1 - \delta$

Table 7: The notation used in our proposed algorithms.
Notation in Section 4.1

$p_j := \mathbb{P}(S = a_j)$	Probability of occurrence (or ratio) of $a_j \in \mathcal{S}$ in $\mathcal{D}_{\mathbf{p}}$
P_σ	System of polynomials $[p_j]_{j \in [c\mathcal{S}]}^\top = [\sum_{(y_1, \dots, y_M) \in \sigma^{-1}(a_j)}]_{j \in [c\mathcal{S}]}^\top$
Ψ_σ	Mapping of each $r_j \in \mathcal{Y}$ to its solution in P_σ , assuming \mathbf{p} is known
$\hat{\mathbf{r}}, \hat{\mathbf{p}}$	Estimates of \mathbf{r} and \mathbf{p}
$\bar{p}_j := \sum_{k=1}^{m_{\mathbf{p}}} \mathbb{1}\{s_k = a_j\}/m_{\mathbf{p}}$	Estimate of p_j given NESY dataset $\mathcal{T}_{\mathbf{p}}$

Notation in Section 4.2

$n > 0$	Size of each batch of NESY samples
i	Index over $[M]$
j	Index over $[c]$
ℓ	Index over $[n]$
$(x_{\ell,1}, \dots, x_{\ell,M}, s_\ell)$	ℓ -th NESY training sample in the input batch
R_ℓ	Size of $\sigma^{-1}(s_\ell)$
t	Index over $[R_\ell]$
\mathbf{P}_i	Matrix in $[0, 1]^{n \times c}$, where $P_i[\ell, j] = f^j(x_{\ell,i})$
\mathbf{Q}_i	Matrix in $[0, 1]^{n \times c}$, where $Q_i[\ell, j]$ is the pseudo-label assigned with label $j \in \mathcal{Y}$ for instance $x_{\ell,i}$
$q_{\ell,i,j}$	A Boolean variable that is true if $x_{\ell,i}$ is assigned with label $j \in \mathcal{Y}$ and false otherwise
$\varphi_{\ell,t}$	Conjunction over the $q_{\ell,i,j}$ Boolean variables that encodes the t -th label vector in $\sigma^{-1}(s_\ell)$
$\Phi_\ell = \varphi_{\ell,1} \vee \dots \vee \varphi_{\ell,R_\ell}$	DNF formula encoding the label vectors in $\sigma^{-1}(s_\ell)$
$\alpha_{\ell,t}$	A fresh Boolean variable associated with each $\varphi_{\ell,t}$ by the Tseytin transformation

Notation in Section 4.3

$n > 0$	Size of each batch of test input instances from \mathcal{X}
\mathbf{P}	Matrix in $R^{n \times c}$ of the f 's scores on the test instances of the input batch
\mathbf{P}'	Matrix in $R^{n \times c}$ storing the CAROT's adjusted scores for \mathbf{P}
$H(\mathbf{P}')$	Entropy of \mathbf{P}'
$\eta, \tau > 0$	Parameters of robust semi-constrained optimal transport problem [23]