Learning from Constraints

Zsolt Zombori

HUN-REN Alfréd Rényi Institute of Mathematics, Budapest
KU Leuven, Leuven

2025. November

Learning from Constraints

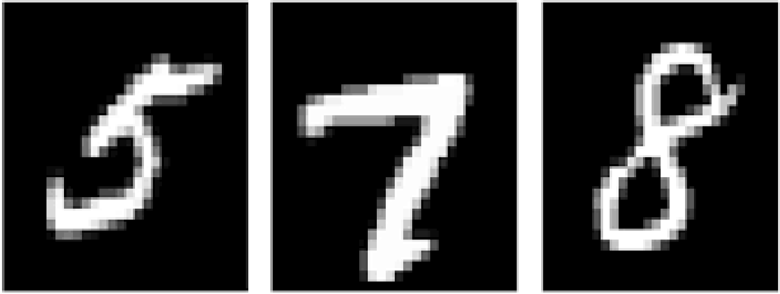
- Train a neural probabilistic classifier with gradient descent
- No direct supervision is available
- Given some constraints restricting the allowed outputs

How should we learn from constraints?

Examples of Constraints – Human Labelling Error

Constraint: True label is one of $\{$horse, mule, donkey$\}$.

Examples of Constraints – Distant Supervision



Constraint: The sum of the three digits is 20.

Illustrative Example

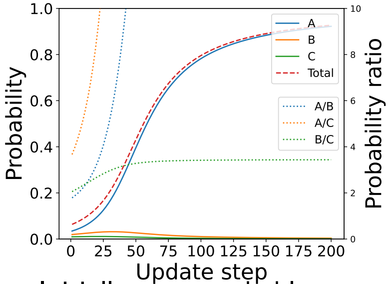
- Single input $\mathbf{x}$
- 10 possible outputs: $\{A, B, C, D, E, F, G, H, I, J\}$
- Unknown true output $\mathbf{y}_{\mathrm{true}}$
- Constraint specifies three *acceptable* outputs $\mathbf{y} = \{A, B, C\}$
- Many ways to perfectly satisfy the constraint: any distribution that assigns all probability to the allowed outputs

Train to Avoid Constraint Violation

- Maximize the likelihood of the supervision
- Probability of the constraint being true: $P_{\text{acc}} = \sum_i \mathbf{y}_i p_i(\mathbf{x})$
- Minimize the *Negative Log Likelihood (NLL)* loss: $L(p(\mathbf{x}), \mathbf{y}) = -\log P_{\text{acc}}$
- Equivalent to minimizing the KL divergence between two Bernoulli distributions: $p_m = (P_{\text{acc}}, 1 - P_{\text{acc}})$, $p_r = (1, 0)$

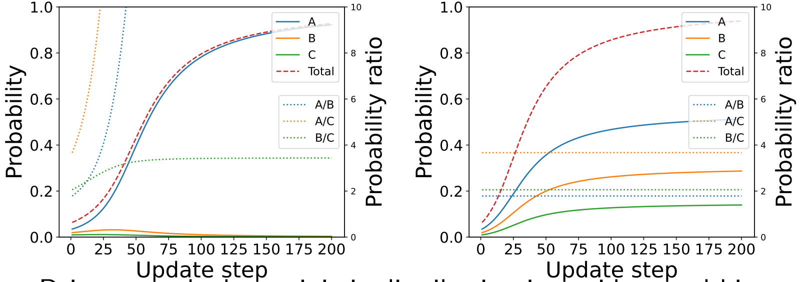
$$
\begin{aligned}
KL(p_r || p_m) =& \sum_{x \in \chi} p_r(x) \log \frac{p_r(x)}{p_m(x)} \\
=& 1 \frac{1}{\log P_{\text{acc}}} + 0 \frac{0}{\log(1 - P_{\text{acc}})} \\
=& -\log P_{\text{acc}}
\end{aligned}
$$

Minimizing the NLL Loss: the Winner Takes All



- Initially most probable output gets all the probability mass
- The constraint is fully satisfied
- But we may have selected the wrong output!
- NLL-loss leads to overconfidence and biased selection
- Can hinder interactions with other training signal

Avoiding Winner-take-all: Preserving Probability Ratios



- Drive towards deterministic distribution is a widespread bias
- The winner depends on initial network configuations

Probability Ratio Preserving (PRP) property: gradient update on sample $(\mathbf{x}, \mathbf{y})$ should preserve the probability ratios among outputs $y_i, y_j \in Y$

Avoiding Winner-take-all: Preserving Probability Ratios

- PRP property aims to preserve the model's uncertainty with respect to unspecified details
- Related to *entropy regularization*
  - Both combat overconfidence
  - Entropy regularization alters the target distribution: converges to the same optimum
  - PRP alters the update operation: convergence point depends on the initial configuration, which can be altered by other training signal

Libra-loss

$$L_{libra}(p(\mathbf{x}),\mathbf{y}) = \underbrace{-\frac{1}{k}\sum_{i=1}^{m}\mathbf{y}_i \log(p(\mathbf{x})_i)}_{\text{Allowed term}} + \underbrace{\log\left(1-\sum_{i=1}^{m}\mathbf{y}_i p(\mathbf{x})_i\right)}_{\text{Disallowed term}}$$

1. PRP property holds for Libra-loss if the model is small (softmax regression)
2. adding more layers makes the property just an approximation
3. Applying a continuously differentiable $h : \mathbb{R} \to \mathbb{R}$ function on Libra-loss preserves the PRP property
4. Any loss function with the PRP property can be constructed from Libra-loss via a suitable h function

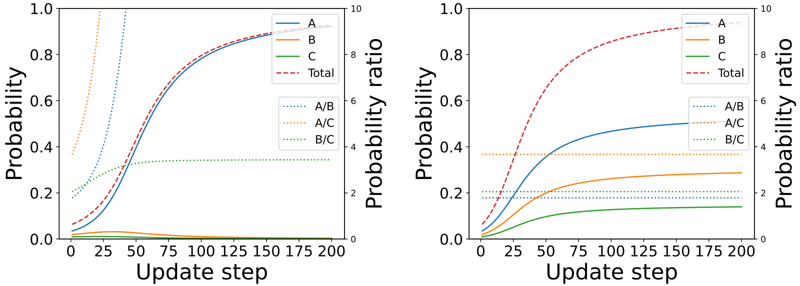
Zombori, Z., Rissaki, A., Szabó, K., Gatterbauer, W., and Benedikt, M. Towards unbiased exploration in partial label learning. Journal of Machine Learning Research, 2024.

Libra-loss vs NLL-loss vs Entropy Regularization

$$
\begin{aligned}
L_{NLL}(p(\mathbf{x}), \mathbf{y}) &= && -\log P_{\mathrm{acc}} \\
L_{libra}(p(\mathbf{x}), \mathbf{y}) &= -\frac{1}{k}\sum_{i=1}^{m} \mathbf{y}_i \log(p(\mathbf{x})_i) && +\log\left(1 - P_{\mathrm{acc}}\right) \\
&= -\sum_{\{i|y_i=1\}} \frac{1}{k} \log(p(\mathbf{x})_i) && +\log\left(1 - P_{\mathrm{acc}}\right) \\
&= H\left(U_{\mathbf{y}}, p(\mathbf{x})\right) && +\log\left(1 - P_{\mathrm{acc}}\right)
\end{aligned}
$$

- $U_{\mathbf{y}}$ is the uniform distribution over the k allowed outputs
- $H(U_{\mathbf{y}}, p(\mathbf{x}))$ is the cross entropy of $p(\mathbf{x})$ relative to $U_{\mathbf{y}}$
- First term maximizes the entropy among accessible outputs
- Second term is similar to the NLL loss

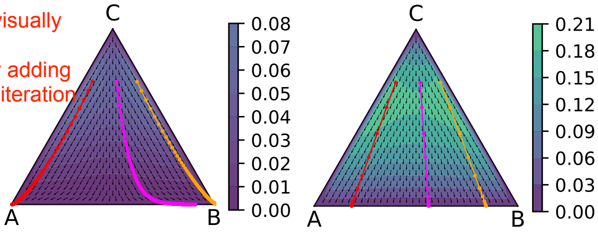
Libra Yields Exact PRP with Shallow Network



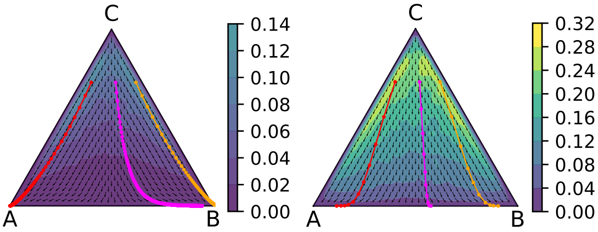
Ratios are preserved exactly only when the network has a single layer!

Approximate PRP with Deep Networks

maybe remind visually about the faster convergence by adding something like "iteration 50"



(a) 1 layer network. **Left:** NLL-loss, **Right:** Libra-loss



(b) 10 layer network. **Left:** NLL-loss, **Right:** Libra-loss

Interaction among Training Samples

	Leonardo	Galileo	Leonardo	Darwin
Init Prob	0.1	0.2	0.2	0.4

Winner-take-all: training on single images yields wrong prediction
Winner-take-all: training on both images can yield any prediction

Libra Loss makes Interaction Smooth

Direct update Implicit change				
	Leonardo	Galileo	Leonardo	Darwin
Init Prob	0.1	0.2	0.2	0.4
	0.15	0.3	0.25	0.35
	0.2	0.28	0.3	0.42
	0.25	0.35	0.35	0.4
	0.3	0.33	0.4	0.46
	0.35	0.39	0.45	0.44
	0.4	0.36	0.5	0.49
	⋮	⋮	⋮	⋮
	1.0	0.0	1.0	0.0

Conclusion

- Constraints restrict the range of options
- Learning to avoid disallowed configurations often introduces unwanted bias
- Highly dependent on the model and the learning method
- The introduced bias can hinder optimisation
- Removing bias can greatly increase model performance

Acknowledgement

This work has been supported by Hungarian National Excellence Grant 2018-1.2.1-NKP-00008, the Hungarian Artificial Intelligence National Laboratory (RRF-2.3.1-21-2022-00004) and the ELTE TKP 2021-NKTA-62 funding scheme. It has also been supported by the UK's Engineering and Physical Science Research Center under Oxford's EPSRC Impact Acceleration Account Award EP/R511742/1 and EPSRC EP/T022124/1, as well as the National Science Foundation (NSF) under award numbers IIS-1762268 and IIS-1956096.

Cour, T., Sapp, B., , and Taskar, B. (2011).
Learning from partial labels.
Journal of Machine Learning Research, 12(5):1501–1536.

Feng, L. and An, B. (2019).
Partial label learning with self-guided retraining.
In *AAI*.

Guu, K., Pasupat, P., Liu, E., and Liang, P. (2017).
From language to programs: Bridging reinforcement learning and maximum marginal likelihood.
In *ACL*.

Jin, R. and Ghahramani, Z. (2002).
Learning with multiple labels.
In *Neurips*.

Liu, L. and Dietterich, T. (2012).
A conditional multinomial mixture model for superset label learning.
In *NEURIPS*.

Liu, L. and Dietterich, T. G. (2014).
Learnability of the superset label learning problem.
In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1629–1637. JMLR.org.

Nguyen, N. and Caruana, R. (2008).
Classification with partial labels.
In *KDD*.

Tian, Y., Yu, X., and Fu, S. (2023).
Partial label learning: Taxonomy, analysis and outlook.
Neural Networks, 161:708–734.

Wen, H., Cui, J., Hang, H., Liu, J., Wang, Y., and Lin, Z. (2021).
Leveraged weighted loss for partial label learning.
In *ICML*.

Zombori, Z., Rissaki, A., Szabó, K., Gatterbauer, W., and Benedikt, M. (2024).
Towards unbiased exploration in partial label learning.
JMLR, 25(1).