Model Misspecification in Multiple Weak Supervision

Salva Rühling Cachay
Technical University of Darmstadt
salvaruehling@gmail.com

Benedikt Boecking
Carnegie Mellon University

Artur Dubrawski
Carnegie Mellon University

Abstract

Data programming has proven to be an attractive alternative to costly hand-labeling of data. In this paradigm, users encode domain knowledge into *labeling functions* (LF), heuristics that label a subset of the data noisily and may have complex dependencies. The effects on test set performance of a downstream classifier caused by label model misspecification are understudied—presenting a serious knowledge gap to practitioners, in particular since LF dependencies are frequently ignored. In this paper, we focus on modeling errors due to structure over-specification. Based on novel theoretical bounds on the modeling error, we empirically show that this error can be substantial, even when modeling a seemingly sensible structure.

1 Introduction and Problem setup

In data programming users define m labeling functions (LF) which noisily label subsets of the data [1]. These noisy sources are then modeled to obtain an estimate of the latent true label. In practice, the LF often exhibit statistical dependencies amongst each other, such as sources operating on the same or similar input. Defining the correct dependency structure is difficult however, thus a common approach in popular libraries [2; 3] and related research [4; 5; 6] is to ignore it.

Let $(x, y) \sim \mathcal{D}$ be the true data generating distribution and for simplicity assume that $y \in \{-1, 1\}$. As in [1], users provide m LFs $\lambda = \lambda(x) \in \{-1, 0, 1\}$, where 0 means that the LF abstained from labeling. Following [1], we model the joint distribution of y, λ as a factor graph which allows for modeling of higher-order dependencies between LFs. More recent weak supervision models and model fitting approaches often only allow for pairwise correlation dependencies to be modeled [7; 8]. To study model misspecification we compare two label models, p_θ for the conditional independent case and p_μ which models higher-order dependencies:

$$p_\mu(\lambda, y) = \frac{1}{Z_\mu} \exp\left(\mu^T \phi(\lambda, y)\right) = Z_\mu^{-1} \exp\left(\mu_1^T \phi_1(\lambda, y) + \mu_2^T \phi_2(\lambda, y)\right), \qquad \mu \in \mathbb{R}^{m+M} \quad (1)$$

$$p_\theta(\lambda, y) = Z_\theta^{-1} \exp\left(\theta^T \phi_1(\lambda, y)\right), \qquad \theta \in \mathbb{R}^m, \quad (2)$$

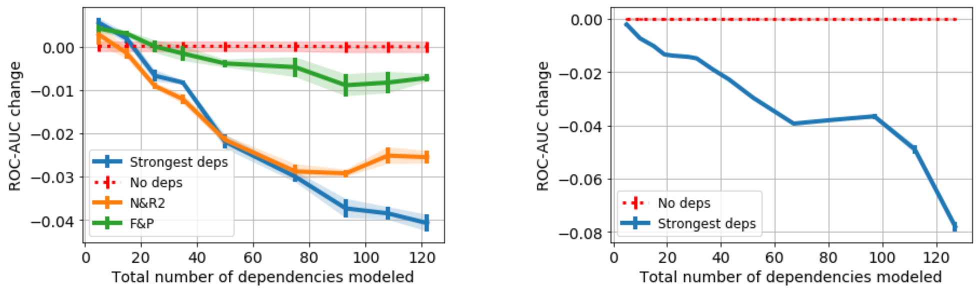
where $\phi_1(\lambda, y) = \lambda y$ are the accuracy factors, $\phi_2(\cdot)$ are arbitrary, higher-order dependencies and $Z_\theta^{-1}, Z_\mu^{-1}$ are normalization constants. We assume w.l.o.g. that factors are bounded ≤ 1. Finally, we extend [1] by introducing *negated*, *bolstering*, *priority* dependencies (definitions in the appendix), e.g. the latter encoding the notion that one LF's vote should be prioritized over the one from a noisier LF.

Bound on the probabilistic label difference due to model misspecification We now state our bound on the probabilistic label difference (which we prove in the appendix):

$$|p_\mu(y \mid \lambda) - p_\theta(y \mid \lambda)| \leq \frac{1}{2}||\mu_1 - \theta||_1 + \frac{1}{4}||\mu_2||_1 \quad (3)$$

The bound naturally involves the accuracy parameter estimation error $||\mu_1 - \theta||_1$ and the learned strength of the dependencies only modeled in p_μ. This is an important quantity of interest since the probabilistic labels are used to train a downstream model. Unsurprisingly then, this quantity reappears as a main factor controlling the generalization risk, see the proof of theorem 1 in [7]. The presented bound is tighter than the one from [7], while in addition accounting for model misspecification.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.



(a) IMDB (0.81)

(b) Bias Bios: professor or teacher (0.91)

Figure 1: Modeling more than a handful dependencies (as the ones in Table 1) significantly deteriorates ROC-AUC downstream performance as compared to simply ignoring them ("No deps"), by up to 4 and 8 points. This effect intensifies as we model more dependencies. In brackets, the baseline score for the independent model.

2 Experiments

2.0.1 Proxy for finding true dependencies in real datasets

The underlying *true* structure between two real labeling functions λ_j, λ_k is, of course, unknown. However, by using true training labels (solely for this purpose) together with the observed LF votes, we can compute resulting factor values for each data point i, to observe empirical strength of dependency factor l over a training set: $v^l_{j,k} = \sum_i \phi^l(\lambda(x_i)_j, \lambda(x_i)_k, y_i)$. Sorting dependencies l according to $v^l_{j,k}$ in descending order, we then choose to model the top d dependencies. These are the dependencies for which the true labels provide the most evidence of being correct.

2.0.2 Downstream model performance deterioration due to structure over-specification

For the following experiment we use the IMDB Movie Review Sentiment dataset consisting of $n = 25$k training and test samples each [9] and manually select a set of $m = 135$ sensible LFs that label on the presence of a single word or a pair of words (i.e. uni-/bi-gram LFs). In addition we use the Bias in Bios dataset [10] from which we create a binary classification task to distinguish the frequently occurring occupations professor or teacher ($n = 12294, m = 85$). We deliberately choose unigram and bigram LFs so as to create dependencies we expect to help with downstream model performance.

Table 1: The strongest and weakest dependencies for the IMDB LFs

LF$_j$	LF$_k$	factor type l	factor value $v^l_{j,k}$
best	great	bolstering	801
original	bad	priority	327
special	not special	negated	8
bad	absolutely horrible	reinforcing	7

We choose different $d \in \{1, 3, 5, \ldots, 40\}$ and then model the strongest $\leq d$ dependencies of each factor l according to v^l. An example of the strongest and weakest dependencies for the IMDB dataset is shown in Table 1. For the Bias in Bios experiment, an example of a strong reinforcing dependency is that the term ‘phd’ appears in addition to the term ‘university’. We report the test set performance of a simple 3-layer neural network trained on the probabilistic labels, averaged out over 100 runs. While for IMDB we observe a marginal boost (< 0.005) in performance when modeling the strongest $d = 1, 3$ dependencies of each factor $(5, 15$ in total), the main take-away is the following:

We find that modeling more than a handful of dependencies significantly *deteriorates* the downstream end classifier performance (by up to 8 ROC-AUC points) as compared to simply ignoring them (Fig. 1). The performance worsens as we increase d, i.e. as we model more, slightly weaker, dependencies. We reiterate that these additional dependencies still, semantically, make sense (as depicted in Table 1, where the weakest ones are modeled only for the case where the total number of dependencies $= 122$).

Discussion Even though this result and insight is highly relevant for practitioners, it has, to the best of our knowledge, not been explored in detail. It may come as a surprise that modeling seemingly sensible dependencies can significantly deteriorate the targeted downstream model performance. We hypothesize that this is due to the true model being close to the conditionally independent case and in Eq. 3 we see that the bound becomes looser as more incorrect dependencies are modeled. Also, as [8] briefly note, more complex models often suffer of a higher sample complexity. We can conclude from this that ignoring potential dependencies will often be a reasonable baseline for practitioners.

References

[1] A. Ratner, C. De Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," *Advances in neural information processing systems*, vol. 29, 05 2016.

[2] S. H. Bach, D. Rodriguez, Y. Liu, C. Luo, H. Shao, C. Xia, S. Sen, A. Ratner, B. Hancock, H. Alborzi, R. Kuchhal, C. Ré, and R. Malkin, "Snorkel drybell: A case study in deploying weak supervision at industrial scale," in *Proceedings of the 2019 International Conference on Management of Data*, ser. SIGMOD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 362–375.

[3] A. Ratner, S. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," *The VLDB Journal*, vol. 29, 07 2019.

[4] P. Varma and C. Ré, "Snuba: Automating weak supervision to label training data," in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 12, no. 3. NIH Public Access, 2018, p. 223.

[5] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models," *Journal of Machine Learning Research*, vol. 15, pp. 2773–2832, 2014.

[6] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the em algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 20–28, 1979.

[7] A. Ratner, B. Hancock, J. Dunnmon, F. Sala, S. Pandey, and C. Ré, "Training complex models with multi-task weak supervision," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 4763–4771, 07 2019.

[8] D. Y. Fu, M. F. Chen, F. Sala, S. Hooper, K. Fatahalian, and C. Ré, "Fast and three-rious: Speeding up weak supervision with triplet methods," *ICML*, 2020.

[9] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 142–150.

[10] M. De-Arteaga, A. Romanov, H. Wallach, J. Chayes, C. Borgs, A. Chouldechova, S. Geyik, K. Kenthapadi, and A. T. Kalai, "Bias in bios: A case study of semantic representation bias in a high-stakes setting," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019, pp. 120–128.

3 Appendix

3.1 Problem setup recap

Let $(x, y) \sim \mathcal{D}$ be the true data generating distribution and for simplicity assume that $y \in \mathcal{Y} = \{-1, 1\}$. Users provide m labeling functions (LFs) $\lambda = \lambda(x) \in \{-1, 0, 1\}$, where 0 means that the LF abstained from labeling. We compare two label models, p_θ for the conditional independent case, and p_μ which models higher-order dependencies:

$$p_\mu(\lambda, y) = \frac{1}{Z_\mu} \exp\left(\mu^T \phi(\lambda, y)\right) = Z_\mu^{-1} \exp\left(\mu_1^T \phi_1(\lambda, y) + \mu_2^T \phi_2(\lambda, y)\right), \qquad \mu \in \mathbb{R}^{m+M} \tag{4}$$

$$p_\theta(\lambda, y) = Z_\theta^{-1} \exp\left(\theta^T \phi_1(\lambda, y)\right), \qquad \theta \in \mathbb{R}^m, \tag{5}$$

where $\phi_1(\lambda, y) = \lambda y$ are the accuracy factors, $\phi_2(\cdot)$ are arbitrary, higher-order dependencies and $Z_\theta^{-1}, Z_\mu^{-1}$ are normalization constants. We assume w.l.o.g. that factors are bounded ≤ 1.

3.2 Proof of the bound

Bound Our bound on the probabilistic label difference between the two models above is:

$$|p_\mu(y \mid \lambda) - p_\theta(y \mid \lambda)| \leq \frac{1}{2}||\mu_1 - \theta||_1 + \frac{1}{4}||\mu_2||_1 \tag{6}$$

Proof First note that the posterior of the label models as above can be rewritten as follows:

$$\begin{aligned}
p_\mu(y \mid \lambda) &= \frac{p_\mu(\lambda, y)}{p_\mu(\lambda)} = \frac{p_\mu(\lambda, y)}{\sum_{\tilde{y} \in \mathcal{Y}} p_\mu(\lambda, \tilde{y})} \\
&= \frac{Z_\mu^{-1} \exp\left(\mu^T \phi(\lambda, y)\right)}{\sum_{\tilde{y} \in \mathcal{Y}} Z_\mu^{-1} \exp\left(\mu^T \phi(\lambda, \tilde{y})\right)} \\
&= \frac{\exp\left(\mu^T \phi(\lambda, y)\right)}{\sum_{\tilde{y} \in \mathcal{Y}} \exp\left(\mu^T \phi(\lambda, \tilde{y})\right)} \\
&= \frac{\exp\left(\mu^T \phi(\lambda, y)\right)}{\exp\left(\mu^T \phi(\lambda, y)\right) + \exp\left(\mu^T \phi(\lambda, -y)\right)} \\
&= \frac{1}{1 + \exp\left(\mu^T \left(\phi(\lambda, -y) - \phi(\lambda, y)\right)\right)} \\
&= \sigma\left(\mu^T \left(\phi(\lambda, y) - \phi(\lambda, -y)\right)\right) \\
&= \sigma\left(2\mu_1^T \phi_1(\lambda, y) + \mu_2^T \left(\phi_2(\lambda, y) - \phi_2(\lambda, -y)\right)\right),
\end{aligned}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function and we used the fact that the accuracy factors are odd functions, i.e. $\phi_1(\lambda, -y) = -\lambda y = -\phi_1(\lambda, y)$. Analogously, $p_\theta(y \mid \lambda) = \sigma\left(2\theta^T \phi_1(\lambda, y)\right)$. Therefore we have that

$$|p_\mu(y \mid \lambda) - p_\theta(y \mid \lambda)| = \left|\sigma\left(2\mu_1^T \phi_1(\lambda, y) + \mu_2^T \left(\phi_2(\lambda, y) - \phi_2(\lambda, -y)\right)\right) - \sigma\left(2\theta^T \phi_1(\lambda, y)\right)\right|$$

By the mean value theorem it follows that for some c between the arguments of σ above

$$\begin{aligned}
&= \sigma'(c) \left|\left(2\mu_1^T \phi_1(\lambda, y) + \mu_2^T \left(\phi_2(\lambda, y) - \phi_2(\lambda, -y)\right)\right) - 2\theta^T \phi_1(\lambda, y)\right| \\
&= \sigma'(c) \left|2\left(\mu_1 - \theta\right)^T \phi_1(\lambda, y) + \mu_2^T \left(\phi_2(\lambda, y) - \phi_2(\lambda, -y)\right)\right|
\end{aligned}$$

Using the triangle inequality and the fact that $\max_x \sigma'(x) = \max_x \sigma(x)(1 - \sigma(x)) = \frac{1}{4}$, we can now bound this expression as follows

$$\leq \frac{1}{2}\left|\left(\mu_1 - \theta\right)^T \phi_1(\lambda, y)\right| + \frac{1}{4}\left|\mu_2^T \left(\phi_2(\lambda, y) - \phi_2(\lambda, -y)\right)\right|$$

finally, since the defined higher-order dependencies are indicator functions $\neq 0$ for only one $y \in \mathcal{Y}$, and if $||q||_\infty \leq 1$ then $|x^T q| = |\sum_i x_i q_i| \leq \sum_i |x_i q_i| \leq \sum_i |x_i| = ||x||_1$, this reduces to

$$\leq \frac{1}{2}||\mu_1 - \theta||_1 + \frac{1}{4}||\mu_2||_1.$$

3.3 Factor Definitions

We supplement the factor definitions of the used higher-order dependencies (the first two stem from [1], the rest we defined ourselves for the conducted experiments). Whenever a factor $\phi_{j,k}(\lambda, y)$ is not symmetric (all factors, besides *bolstering*), we define it so that LF_k acts on (e.g. *negates*) LF_j.
For the *fixing* dependency we have:

$$\phi_{j,k}^{Fix}(\lambda, y) = \begin{cases} +1 & \text{if } \lambda_j = -y \wedge \lambda_k = y \\ -1 & \text{if } \lambda_j = 0 \wedge \lambda_k \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

for the *reinforcing* one:

$$\phi_{j,k}^{Rei}(\lambda, y) = \begin{cases} +1 & \text{if } \lambda_j = \lambda_k = y \\ -1 & \text{if } \lambda_j = 0 \wedge \lambda_k \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

for the *priority* factor:

$$\phi_{j,k}^{Pri}(\lambda, y) = \begin{cases} +1 & \text{if } \lambda_j = -y \wedge \lambda_k = y \\ -1 & \text{if } \lambda_j = y \wedge \lambda_k = -y \\ 0 & \text{otherwise} \end{cases}$$

for the *bolstering*:

$$\phi_{j,k}^{Bol}(\lambda, y) = \begin{cases} +1 & \text{if } \lambda_j = \lambda_k = y \\ -1 & \text{if } \lambda_j = \lambda_k \neq y \vee \lambda_j = -\lambda_k \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

and, finally, for the *negated* factor:

$$\phi_{j,k}^{Neg}(\lambda, y) = \begin{cases} +1 & \text{if } \lambda_j = -y \wedge \lambda_k = y \\ -1 & \text{if } (\lambda_j = y \wedge \lambda_k = -y) \vee \lambda_j = \lambda_k \neq 0 \\ 0 & \text{otherwise} \end{cases}$$