End-to-End Weak Supervision

Salva Rühling Cachay[1,2], Benedikt Boecking[1], Artur Dubrawski[1]

Code: https://github.com/autonlab/weasel

1 Auton Lab @ Carnegie Mellon University

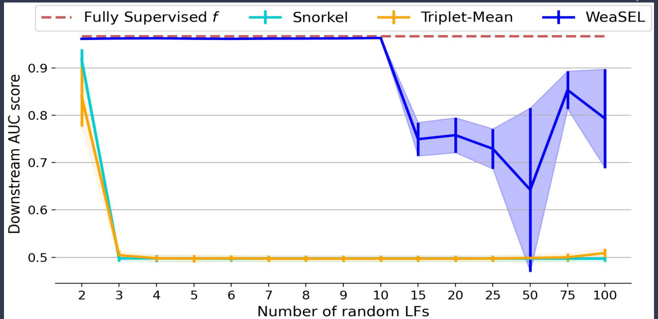
2 Technische Universität Darmstadt

Problem setting

Alleviate the *data labeling bottleneck* by aggregating multiple noisy labeling heuristics/functions (LFs), as in [1]

Contributions & Results

- Introduce **WeaSEL**: A **flexible, end-to-end** method for multi-source weak supervision
- Empirically demonstrate WeaSEL's **robustness** to noisy and highly correlated LFs
- Gains of up to 6.1 F1 points over two-step approaches
- Beats state-of-the-art **crowdsourcing** methods on a crowdsourcing dataset (with a mutual-information based loss)



When the LF set consists of a perfect LF (true labels) and up to 10 independent (!), no better-than-random LFs, WeaSEL recovers the test performance of the same downstream model *f* trained directly on the ground truth labels while related methods collapse (even though no independence assumption is violated).
Similar behavior occurs when the LFs are highly correlated.

Issues with the usual approach

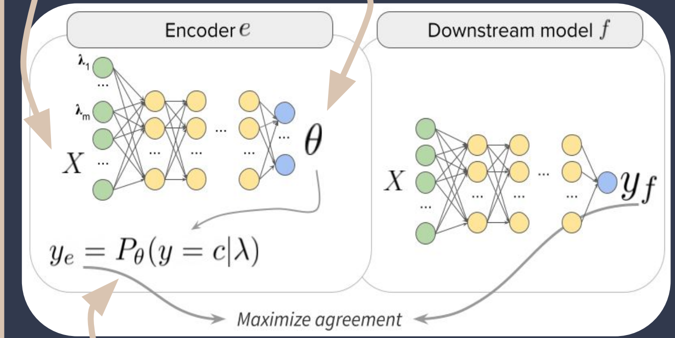
- Two-step approaches like Snorkel **ignore the features** in LF-label modeling (step 1), use the same LF accuracy parameter for each sample, and ignore the LFs when learning the downstream model (step 2)
  → *We streamline this process into a joint optimization step, do not ignore features/LFs, and have sample-dependent weights*
- Statistical **dependencies** between LFs hard to model and often **ignored.** This makes the label model Naive-Bayes like and violates assumptions needed for theory
  → *Our neural approach is shown to be considerably less susceptible to dependencies than prior work*

Algorithm 1 `WeaSEL`: The proposed Weakly Supervised End-to-end Learning algorithm for learning from multiple weak supervision sources.

input: batch size n, networks e, f, inverse temperatures τ_1, τ_2, noise-aware loss function L, class balance $P(y)$.
for sampled minibatch $\{z^{(k)} = (\mathbf{x}^{(k)}, \boldsymbol{\lambda}^{(k)})\}_{k=1}^{n}$ **do**
  for all $k \in \{1, \ldots, n\}$ **do**
    # Produce accuracy scores for all weak sources
    $\theta\left(z^{(k)}\right) = \text{softmax}\left(e(z^{(k)})\tau_1\right)$
    # Generate probabilistic labels
    define $\mathbf{s}^{(k)}$ **as** $\mathbf{s}^{(k)} = \theta(z^{(k)})^T \bar{\boldsymbol{\lambda}}^{(k)}$
    $y_e^{(k)} = P_\theta(y|\boldsymbol{\lambda}^{(k)}) = \text{softmax}\left(\mathbf{s}^{(k)}\tau_2\right) \odot P(y)$
    # Downstream model forward pass
    $y_f^{(k)} = f(\mathbf{x}^{(k)})$
  end for
  $\mathcal{L}_f = \frac{1}{n}\sum_{k=1}^{n} L\left(y_f^{(k)}, \texttt{stop-grad}\left(y_e^{(k)}\right)\right)$
  $\mathcal{L}_e = \frac{1}{n}\sum_{k=1}^{n} L\left(y_e^{(k)}, \texttt{stop-grad}\left(y_f^{(k)}\right)\right)$
  update e to minimize $\mathcal{L}_e$, and f to minimize $\mathcal{L}_f$
end for
return downstream network $f(\cdot)$

Include the features, **X,** in LF-label modeling

Predict **sample-dependent accuracy scores** (the PGM parameters of prior work) for each of the *m* LFs



Reparameterize PGM posterior of prior work with a neural net (a normalized linear combination, which avoids overly trivial solutions).

Use predictions of other model as **pseudo-labels** (stop-grad operation) for end-to-end **joint optimization.**

[1] Alexander J Ratner, et al.. "Data programming: Creating large training sets, quickly." In: NeurIPS, 2016