INTERACTIVE WEAK SUPERVISION: LEARNING USEFUL HEURISTICS FOR DATA LABELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Obtaining large annotated datasets is critical for training successful machine learning models and it is often a bottleneck in practice. Weak supervision offers a promising alternative for producing labeled datasets without ground truth annotations by generating probabilistic labels using multiple noisy heuristics. This process can scale to large datasets and has demonstrated state of the art performance in diverse domains such as healthcare and e-commerce. One practical issue with learning from user-generated heuristics is that their creation requires creativity, foresight, and expertise from those who hand-craft them, a process which can be tedious and subjective. We develop the first framework for interactive weak supervision in which a method proposes heuristics and learns from user feedback given on each proposed heuristic. Our experiments demonstrate that only a small number of feedback iterations are needed to train models that achieve highly competitive test set performance without access to ground truth training labels. We conduct a real user study, which shows that users are able to effectively provide feedback on heuristics and that test set results track the performance of simulated oracles.

1 INTRODUCTION

The performance of supervised machine learning (ML) hinges on the availability of labeled data in sufficient quantity and quality. However, labeled data for applications of ML can be scarce, and the common process of obtaining labels by having annotators inspect individual samples is often expensive and time consuming. Additionally, this cost is frequently exacerbated by factors such as privacy concerns, required expert knowledge, and shifting problem definitions.

Weak supervision provides a promising alternative, reducing the need for humans to hand label large datasets to train machine learning models (Riedel et al., 2010; Hoffmann et al., 2011; Ratner et al., 2016). A recent approach called data programming (Ratner et al., 2016) combines multiple weak supervision sources by using an unsupervised label model to estimate the latent true class label, an idea that has close connections to modeling workers in crowd-sourcing (Dawid & Skene, 1979; Karger et al., 2011; Dalvi et al., 2013; Zhang et al., 2014). The approach enables subject matter experts to specify *labeling functions (LFs)*—functions that encode domain knowledge and noisily annotate subsets of data, such as user-specified heuristics or external knowledge bases—instead of needing to inspect and label individual samples. These weak supervision approaches have been used on a wide variety of data types such as MRI sequence and unstructured text, and in various domains such as healthcare and e-commerce (Fries et al., 2019; Halpern et al., 2014; Bach et al., 2019; Ré et al., 2020). Not only does the use of multiple sources of weak supervision provide a scalable framework for creating large labeled datasets, but it can also be viewed as a vehicle to incorporate high level, conceptual feedback into the data labeling process.

In data programming, each LF is an imperfect but reasonably accurate heuristic, such as a pre-trained classifier or keyword lookup. For example, for the popular *20 newsgroups* dataset, an LF to identify the class '*sci.space*' may look for the token '*launch*' in documents and would be right about 70% of the time. While data programming can be very effective when done right, experts may spend a significant amount of time designing the weak supervision sources (Varma & Ré, 2018) and must often inspect samples at random to generate ideas (Cohen-Wang et al., 2019). In our *20 newsgroups* example, we may randomly see a document mentioning '*Salman Rushdie*' and realize that the name of a famous atheist could be a good heuristic to identify posts in '*alt.atheism*'. While such a heuristic seems obvious after the fact, we have to chance upon the right documents to generate these ideas. In

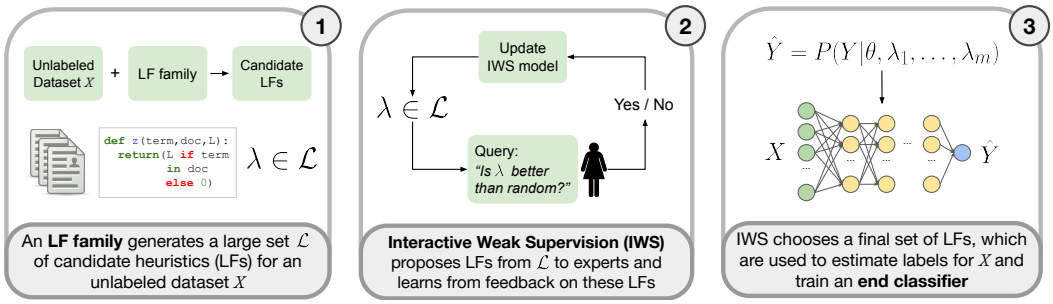

Figure 1: Interactive Weak Supervision (IWS) helps experts discover good labeling functions (LFs).

practice, coming up with effective LFs becomes difficult after the first few. Substantial foresight and creativity are required to create a function that applies to a non-negligible subset of given data, is novel, and adds predictive value.

We propose a new approach for training supervised ML models by interactively discovering weak supervision sources. Our method supports domain experts in fast discovery of good LFs. The method queries users in an active fashion for feedback about candidate LFs, from which a model learns to distinguish LFs likely to have good accuracy. Upon completion, our method produces a final set of LFs. We use this set to create an estimate of the latent class label via an unsupervised label model and train the final supervised ML model using a noise aware loss function on the estimated labels as in Ratner et al. (2016). The approach relies on the observation that many applications allow for heuristics of varying quality to be generated at scale, similar to Varma & Ré (2018), and that experts can provide good judgment by identifying some LFs that have reasonable accuracy. The full pipeline of the proposed approach, termed *Interactive Weak Supervision (IWS)*, is illustrated in Fig. 1. Our contributions are:

1. We propose, to the best of our knowledge, the first interactive method for weak supervision in which queries to be annotated are not data points but labeling functions. This approach automates the design of useful data labeling heuristics.
2. We conduct experiments with real users on two text classification datasets. Results support our modeling assumptions, and show that users can provide accurate feedback on automatically generated LFs.
3. Our method shows superior performance compared to standard active learning, meaning we can achieve better test set performance with a smaller number of queries to users. After 200 IWS iterations annotating LFs, we achieve a mean test set AUC that requires at least three times as many active learning iterations annotating samples. In addition, the average user response time for IWS queries is shorter than for data-point-wise active learning queries.

2 Related Work

Active strategies for weak supervision sources have largely focused on combinations of data programming (Ratner et al., 2016) with traditional active learning where samples are presented to users. In Nashaat et al. (2018), a pool of samples is created on which LFs disagree, and active learning strategies are then applied to obtain labels for some of these data points. In Cohen-Wang et al. (2019), specific samples are selected and presented to users in order to inspire the creation of LFs. In Hancock et al. (2018), natural language explanations provided during text labeling are used to generate heuristics. The proposed system uses a semantic parser to convert explanations into logical forms, which represent labeling functions.

Prior work has emphasized that LFs defined by experts frequently have a recurring structure in which elements are swapped to change the higher level concept a function corresponds to (Varma & Ré, 2018; Varma et al., 2017; Bach et al., 2019). As an example, in tasks involving text documents, LFs often follow a repetitive structure in which key terms or phrases and syntactical relationships change, e.g. mentions of specific words (Varma & Ré, 2018; Cohen-Wang et al., 2019; Varma et al., 2019). Prior work relies on this observation to create heuristic generators (Varma & Ré, 2018), LF templates (Bach et al., 2019), and domain-specific primitives (Varma et al., 2017). In particular, in a semi-supervised data programming setting, Varma & Ré (2018) propose a system for automatic generation of labeling functions without user interaction, by using a small set of labeled data. The

authors motivate their system stating that users frequently perform repetitive steps such as guessing optimal numerical thresholds and developing informative text patterns.

Additional related work has focused on the modeling of dependencies among heuristics in data programming (Bach et al., 2017; Varma et al., 2019), on the multi-task setting (Ratner et al., 2019), on handling of multi-resolution sources (Sala et al., 2019), on the use of noisy pairwise labeling functions (Boecking & Dubrawski, 2019), on addressing latent subsets in the data (Varma et al., 2016), and on fast model iteration via the use of pre-trained embeddings (Chen et al., 2020).

3 METHODS

We propose an interactive weak supervision (IWS) approach to assist experts in finding good labeling functions (LFs) for training a classifier on datasets without ground truth labels. We will first describe the general problem setting of learning to classify without ground truth samples by modeling multiple weak supervision sources, as well as the concept of LF families. We then dive into the details of the proposed IWS approach. For brevity, we limit the scope of the end classifier to binary classification, but the presented background and ideas extend to the multi-class settings.

3.1 PRELIMINARIES

Learning with Multiple Weak Supervision Sources Assume each data point $x \in \mathcal{X}$ has a latent class label $y^* \in \mathcal{Y} = \{-1, 1\}$. Given n unlabeled, i.i.d. datapoints $X = \{x_i\}_{i=1}^n$, our goal is to train an end classifier $f : \mathcal{X} \to \mathcal{Y}$ such that $f(x) = y^*$. In data programming (Ratner et al., 2016; 2020), a user provides m LFs $\{\lambda_j\}_{j=1}^m$, where $\lambda_j : \mathcal{X} \to \mathcal{Y} \cup \{0\}$ noisily label the data with $\lambda_j(x) \in \{-1, 1\}$ or abstain with $\lambda_j(x) = 0$. The corresponding LF output matrix is $\Lambda \in \{-1, 0, 1\}^{n \times m}$, where $\Lambda_{i,j} = \lambda_j(x_i)$. In this paper, we assume that each LF λ_j has the same accuracy on each class, $\alpha_j = P(\lambda_j(x) = y^* | \lambda_j(x) \neq 0)$, where accuracy is defined on items where j does not abstain. Further, we denote by $l_j = P(\lambda_j(x) \neq 0)$ the LF propensity (sometimes called LF coverage), i.e. the frequency at which LF j does not abstain.

In data programming, an unsupervised label model $p_\theta(Y, \Lambda)$ produces probabilistic estimates of the latent class labels $Y^* = \{y_i^*\}_{i=1}^n$ using observed Λ by modeling the LF accuracies, propensities, and possibly their dependencies. A number of label model approaches exist in the crowd-sourcing (Dawid & Skene, 1979; Zhang et al., 2014) and the weak supervision literature (Ratner et al., 2020). In this paper, we use a factor graph as proposed in Ratner et al. (2016; 2020) to obtain probabilistic labels by modeling the LF accuracies via factor $\phi_{i,j}^{Lab}(\Lambda, Y) \triangleq \mathbb{1}\{\Lambda_{ij} \neq 0\}$ and labeling propensity by factor $\phi_{i,j}^{Acc}(\Lambda, Y) \triangleq \mathbb{1}\{\Lambda_{ij} = y_i\}$, and for simplicity assume LFs are independent conditional on Y. The label model is defined as

$$p_\theta(Y, \Lambda) \triangleq Z_\theta^{-1} \exp\left(\sum_{i=1}^{n} \theta^\top \phi_i(\Lambda_i, y_i)\right), \tag{1}$$

where Z_θ is a normalizing constant and $\phi_i(\Lambda_i, y_i)$ defined to be the concatenation of the factors for all LFs $j = 1, \ldots, m$ for sample i. We learn θ by minimizing the negative log marginal likelihood given the observed Λ. Finally, following Ratner et al. (2016) the end classifier f is trained using probabilistic labels $p_\theta(Y|\Lambda)$.

Labeling Function Families We define LF families as sets of expert-interpretable weak supervision sources described by functions $z_\phi : \mathcal{X} \mapsto \{-1, 0, 1\}$, for parameters $\phi \in \Phi$. An example are shallow decision trees z_ϕ parameterized by variables or splitting rules ϕ (Varma & Ré, 2018), or a function z_ϕ defining a regular expression for two words where ϕ parameterizes the word choices from a vocabulary Φ. Given such an LF family, we can generate a large set of p candidate heuristics $\mathcal{L} = \{\lambda_j(x) = z_{\phi_j}(x)\}_{j=1}^p$, where $\phi_j \in \Phi$, e.g. by sampling from Φ and pruning low coverage candidates. For text, we can find n-grams within a document frequency range to generate key term lookups, fill placeholders in regular expressions, or generate shallow decision trees (Ratner et al., 2016; Varma & Ré, 2018; Varma et al., 2019). For time series, we can create a large set of LFs based on motifs (Lonardi & Patel, 2002) or graphs of temporal constraints (Guillame-Bert & Dubrawski, 2017). For images, we can create a library of classifiers pre-trained for various specific tasks (e.g. object detectors as used in (Chen et al., 2019)).

3.2 INTERACTIVE WEAK SUPERVISION

Instead of having users provide m good weak supervision sources up front, we want to assist users in discovering them. Successful applications of data programming have established that human experts are able to construct accurate LFs from scratch. Our work leverages the assumption that human experts can also judge these properties when presented with pre-generated LFs of the same form.

Suppose again that we have an unlabeled dataset $X = \{x_i\}_{i=1}^n$, and that our goal is to train an **end classifier** f without access to labels $Y^* = \{y_i^*\}_{i=1}^n$. Assume also that we defined a large pool of p candidate LFs $\mathcal{L} = \{\lambda_j(x)\}_{j=1}^p$ from an LF family (following Sec. 3.1), of varying accuracy and coverage. In IWS, our goal is to identify an optimal subset of LFs $\mathcal{L}^* \subset \mathcal{L}$ to pass to the **label model** in Eq. (1). Below we will quantify how $\mathcal{L}^*$ depends on certain properties of LFs. While we can observe some of these properties—such as coverage, agreement, and conflicts—an important property that we cannot observe is the accuracy of each LF.

Our goal will thus be to infer quantities related to the latent accuracies $\alpha_j \in [0, 1]$ of LFs $\lambda_j \in \mathcal{L}$, given a small amount expert feedback. To do this, we define an **expert-feedback model**, which can be used to infer LF accuracies given a dataset of user feedback. To efficiently train this model, our IWS procedure sequentially chooses an LF $\lambda_j \in \mathcal{L}$ and shows a description of λ_j to an expert, who provides binary feedback about λ_j. We follow ideas from active learning for sequential decision making under uncertainty, in which a probabilistic model guides data collection to efficiently infer quantities of interest within T iterations. After a sequence of feedback iterations, we use the expert-feedback model to provide an estimate $\hat{\mathcal{L}} \subset \mathcal{L}$ of the optimal subset $\mathcal{L}^*$. The label model then uses $\hat{\mathcal{L}}$ to produce a probabilistic estimate of Y^*, which is used to train the end classifier f. The full IWS procedure is shown in the flow chart in Fig. 1 which we describe in detail below.

Expert-Feedback Model We first define a generative model of human expert feedback about LFs, given the latent LF accuracies. This model will form the basis for an online procedure that selects a sequence of LFs to show to human experts. We task experts to classify LFs as either useful or not useful $u_j \in \{0, 1\}$, corresponding to their *belief that LF λ_j is predictive of Y^* at better than random accuracy for the samples where λ_j does not abstain*. Note that prior data programming work (Ratner et al., 2016; 2019; Dunnmon et al., 2020; Saab et al., 2020) assumes and demonstrates that experts are able to use their domain knowledge to make this judgment when creating LFs from scratch. We model the generative process for this feedback and the latent LF accuracies as, for $j = 1, \ldots, t$:

$$u_j \sim \text{Bernoulli}(v_j), \quad v_j = h_\omega(\lambda_j), \quad \omega \sim \text{Prior}(\cdot) \tag{2}$$

where v_j can be viewed as the average probability that a human will label a given LF λ_j as $u_j = 1$, and $h_\omega(\lambda_j)$ is a parameterized function (such as a neural network), mapping each LF λ_j to v_j. Finally, to model the connection between accuracy α_j and v_j, we assume that $v_j = g(\alpha_j)$, where $g : [0, 1] \to [0, 1]$ is a monotonic increasing function mapping unknown LF accuracy α_j to v_j.

After t queries of human feedback, we have produced a query dataset $Q_t = \{(\lambda_j, u_j)\}_{j=1}^t$. Given Q_t, we will infer unknown quantities in the above model, which will be used to choose subsequent LFs λ_j to query by constructing an acquisition function $\varphi_t : \mathcal{L} \to \mathbb{R}$ and optimizing it over $\lambda \in \mathcal{L}$.

Acquisition Strategy and Final Set of LFs To derive an online procedure for our expert queries we need to define the properties of the ideal subset of generated LFs $\mathcal{L}^* \subset \mathcal{L}$ which we want to select. Prior data programming work of Ratner et al. (2016; 2019; 2020) with label models as in Eq. (1) does not provide an explicit analysis of ideal metrics of LF sets and their trade-offs to help define this set. We provide the following theorem, which will motivate our definition for $\mathcal{L}^*$.

Theorem 3.1. *Assume a binary classification setting, m independent labeling functions with accuracy $\alpha_j \in [0, 1]$ and labeling propensity $l_j \in [0, 1]$. For a label model as in Eq. (1) with given label model parameters $\hat{\theta} \in \mathbb{R}^{2m}$, and for any $i \in \{1, \ldots, n\}$,*

$$P(\hat{y}_i = y_i^*) \geq 1 - \exp\left(-\frac{(\sum_{j=1}^m \hat{\theta}_j^{(1)}(2\alpha_j - 1)l_j)^2}{2||\hat{\theta}^{(1)}||^2}\right)$$

where $\hat{\theta}^{(1)}$ are the m weights of ϕ^{Acc}, and $\hat{y}_i \in \{-1, 1\}$ is the label model estimate for y_i^*.

Proof. The proof is given in Appendix B. □

This theorem indicates that one can rank LFs according to $(2\alpha_j - 1)\hat{l}_j$ where $\alpha_j, \hat{l}_j$ are the unknown accuracy and observed coverage of LF j, respectively (we provide additional analysis in Appendix B). Our analysis further suggests the importance of obtaining LFs with an accuracy gap above chance. Intuitively, we do not want to add excessive noise by including LFs too close to random. Below, we assume that our final set of LFs is sufficient to accurately learn label model parameters $\hat{\theta}$, and leave analysis of the influence of additional LF properties on learning $\hat{\theta}$ to future work.

To define the ideal final subset of LFs, we distinguish three scenarios: (A) there are no restrictions on the size of the final set and any LF can be included, (B) the final set is limited in size (e.g. due to computational considerations) but any LF can be included, (C) only LFs inspected and validated by experts may be included, e.g. due to security or legal considerations. We first provide full details of our IWS algorithm in scenario (A).

Scenario (A): Unbounded LF Set. When there are no restrictions as described above, our analysis in Appendix B indicates that the ideal subset of LFs $\mathcal{L}^*$ includes all those with accuracy greater than a gap above chance, i.e. $\alpha_j > r > 0.5$. Thus, we define the optimal subset in this scenario as

$$\mathcal{L}^* = \{\lambda_j \in \mathcal{L} \ : \ \alpha_j > r\} \,. \tag{3}$$

This is a variation of the task of active level set estimation (LSE), where the goal is to identify all elements in a superlevel set of $\mathcal{L}$ (Zanette et al., 2018; Gotovos, 2013; Bryan et al., 2006). At each step t, we will use the straddle acquisition function (Bryan et al., 2006) for LSE, defined for a candidate $\lambda_j \in \mathcal{L} \setminus Q_{t-1}$ to score LFs highest that are unknown and near the boundary threshold r:

$$\varphi_t^{\text{LSE}}(\lambda_j) = 1.96 \ \sigma_j(Q_{t-1}) - |\mu_j(Q_{t-1}) - r| \tag{4}$$

where $\sigma_j(Q_{t-1}) = \sqrt{\text{Var}[p(\alpha_j|Q_{t-1})]}$ is the standard deviation and $\mu_j(Q_{t-1}) = \mathbb{E}[p(\alpha_j|Q_{t-1})]$ the mean of the posterior LF accuracy. At the end of Sec. 3.2 we describe how we perform approximate inference of $p(\alpha_j|Q_{t-1})$ via an ensemble model. We maximize the acquisition function over the set of candidate LFs, i.e. compute $\lambda_t = \arg\max_{\lambda \in \mathcal{L} \setminus Q_{t-1}} \varphi_t(\lambda)$ by iterating over all unqueried $\lambda \in \mathcal{L}$. We then query a human expert to obtain (λ_t, u_t), and finally update the query dataset $Q_t = Q_{t-1} \cup \{(\lambda_t, u_t)\}$. After a sequence of T queries we return the following estimate of $\mathcal{L}^*$:

$$\hat{\mathcal{L}} = \{\lambda_j \in \mathcal{L} \ : \ \mu_j(Q_T) > r\} \,. \tag{5}$$

The LF output matrix Λ, comprised of all $\lambda_j \in \hat{\mathcal{L}}$, is then used to produce an estimate $\widehat{Y}$ of the true class labels via the label model $P_\theta(Y|\Lambda)$. Finally, a noise-aware discriminative end classifier f is trained on $(X, \widehat{Y})$. See Algorithm 1 for pseudocode describing this full IWS-LSE procedure. In our experiments, we set $r = 0.7$, though an ablation study shows that IWS-LSE works well for a range of thresholds $r > 0.5$ (Appendix A.3, Fig. 7). Note that the LSE acquisition function aims to reduce uncertainty around r, and therefore tends to explore LFs that have coverage on parts of Y that we are still uncertain about. We denote the algorithm for scenario (A) by **IWS-LSE-a**.

Additional Cases *Scenario (B): Bounded LF Set.* If the final set is restricted in size to m LFs, e.g. due to computational or latency considerations when learning the label model (Eq. (1)), we need to take the trade-off of LF accuracy and LF coverage into account. Let $\hat{l}_j$ be the observed empirical coverage of LF λ_j. We want to identify LFs with accuracy above r and rank them according to their accuracy-coverage trade-off, thus our analysis in the appendix suggests the optimal subset is

$$\mathcal{L}^* = \underset{\mathcal{D} \subseteq \mathcal{L}, |\mathcal{D}|=m}{\arg\max} \sum_{\lambda_j \in \mathcal{D}} \left(\mathbb{1}_{\{\alpha_j > r\}} (2 * \alpha_j - 1) * \hat{l}_j \right) . \tag{6}$$

Since the LF accuracy-coverage trade-off only comes into effect if $\alpha_j > r$, we use the same acquisition function φ_t^{LSE} as in scenario (A), and then select the final set as $\hat{\mathcal{L}} = \{\lambda_j \in \mathcal{D} \ : \ \arg\max_{\mathcal{D} \subseteq \mathcal{L}, |\mathcal{D}|=m} \sum_{\lambda_j \in \mathcal{D}} (\mathbb{1}_{\{\mu_j(Q_T) > r\}} (2 * \mu_j(Q_T) - 1) * \hat{l}_j)\}$ which simply thresholding by $\mu_j(Q_T)$ and then sorting. We denote the algorithm for scenario (B) by **IWS-LSE-ac**.

Scenario (C): Validated LF Set. Finally, in some application scenarios only LFs inspected and validated by experts should be used to estimate Y^*, e.g. due to security or legal considerations. This leads to an active search problem (Garnett et al., 2015) where our aim is to identify a maximum

number of useful LFs (i.e. $u = 1$) in $\mathcal{L}$ given a budget of T user queries, i.e. to compute

$$\mathcal{L}^*_{\text{AS}} = \underset{\mathcal{D} \subset \mathcal{L}, |\mathcal{D}|=T}{\arg\max} \sum_{\boldsymbol{\lambda}_j \in \mathcal{D}} u_j, \quad \hat{\mathcal{L}} = \{\lambda_j \in Q_T \ : \ u_j = 1\}. \tag{7}$$

As in (Garnett et al., 2015; Jiang et al., 2017), we use a one-step look ahead active search acquisition function defined for a candidate $\lambda_j \in \mathcal{L} \setminus Q_{t-1}$ to be the posterior probability that the usefulness label u_j is positive, i.e. $\varphi_t^{\text{AS}}(\lambda_j) = \mu_j(Q_{t-1})$. We denote the algorithm for scenario (C) by **IWS-AS**.

Approximate Inference Details We now describe how we use our expert-feedback model in Eq. (2) to infer $p(\alpha_j|Q_t)$. We defined a generative model of human feedback u_j on query LF λ_j with latent variables v_j and ω. We assumed a connection between v_j and the latent LF accuracy α_j via a monotonic increasing function $\alpha_j = g(v_j)$. Similar to existing work on high dimensional uncertainty estimation (Beluch et al., 2018; Chitta et al., 2018), we use an ensemble $\{\tilde{h}_{\omega^{(i)}}\}_{i=1}^{s}$ of s neural networks $\tilde{h}_\omega$ with parameters ω to predict u_j given input λ_j. To perform this prediction, we need a feature representation $\tau(\lambda_j)$ for LFs that is general and works for any data type and task. For these features, we use the LF output over our unlabeled dataset, i.e. $\tau_0(\lambda_j) = (\lambda_j(x_1), \ldots, \lambda_j(x_n))$. We then project this to d' dimensions using PCA for a final feature representation $\tau(\lambda_j)$, which is given as input to each $\tilde{h}_\omega$. Our neural network ensemble can then effectively learn functions $\tilde{h} : \mathbb{R}^{d'} \to [0, 1]$, which map from features $\tau(\lambda_j)$ to $v_j = p(u_j = 1|Q_t)$. This yields an ensemble of estimates for v_j, and through g^{-1}, of α_j. These are treated as approximate samples from $p(\alpha_j|Q_t)$, and used to form sample-estimates for the above acquisition functions.

Algorithm 1: Interactive Weak Supervision (IWS-LSE-a).

Input : $\mathcal{L}$: set of LFs, T: max iterations, $p(u|\lambda, Q)$: usefulness model for ensemble, φ_t: acquisition function.

1 $Q_0 \leftarrow \varnothing$
2 **for** $t = 1, 2, \ldots, T$ **do**
3 $\quad \lambda_t \leftarrow \arg\max_{\lambda \in \mathcal{L} \setminus Q_{t-1}} \varphi_t(\lambda)$ ▷ Eq. (4)
4 $\quad u_t \leftarrow$ *ExpertQuery*(λ_t)
5 $\quad Q_t \leftarrow Q_{t-1} \cup \{(\lambda_t, u_t)\}$
6 **end**
7 $\hat{\mathcal{L}} \leftarrow \{\lambda_j \in \mathcal{L} \ : \ \mathbb{E}[p(\alpha_j|Q_T)] > r\}$ ▷ Eq. (5)

4 EXPERIMENTS

Our experiments show that heuristics obtained via a small number of iterations of IWS can be used to train a downstream end classifier f with highly competitive test set performance. We first present results obtained with a simulated noiseless IWS oracle. Oracle experiments allow us to answer how our method would perform if users had perfect knowledge about LF accuracy. We then show results from a real-user study, involving experiments in which the query feedback is given by humans. We created six binary text classification tasks on three publicly available datasets: Amazon reviews, IMDB reviews, and BiasBios biographies. We provide full dataset and implementation details in Appendix A. The tasks were chosen such that most English speakers can provide sensible expert feedback on LFs, for ease of reproducibility. Similar text classification datasets have been used in related work (Varma & Ré, 2018; Varma et al., 2019). We note that a head to head comparison of user effort required to respond to sample annotation queries vs LF usefulness queries is application dependent. We do however provide a timing study to showcase the effort required to carry out IWS versus labeling of samples in our specific user experiments, see Table 1.

Approaches All approaches train the same downstream end classifier f one the same inputs X. We show results for *IWS-LSE-a* (unbounded LF set), *IWS-LSE-ac* (bounded LF set), and *IWS-AS* (validated LF set). We compare the test set performance of IWS to a set of alternatives including (1) annotation of samples via *active learning* (uncertainty sampling) by a noiseless oracle, (2) the *Snuba* system (Varma & Ré, 2018)[1], and (3) the performance obtained using *all ground truth training labels*. In Appendix A we also compare our results to the IWS procedure using a random acquisition function (*IWS-random*), in the validated LF set setting.

Creating LF Families for Text Related work (Ratner et al., 2020; Varma et al., 2019) previously demonstrated that word and phrase LFs can provide good weak supervision sources on text. We define

[1] In plots, the number of Snuba iterations correspond to the number of labeled samples used for training.

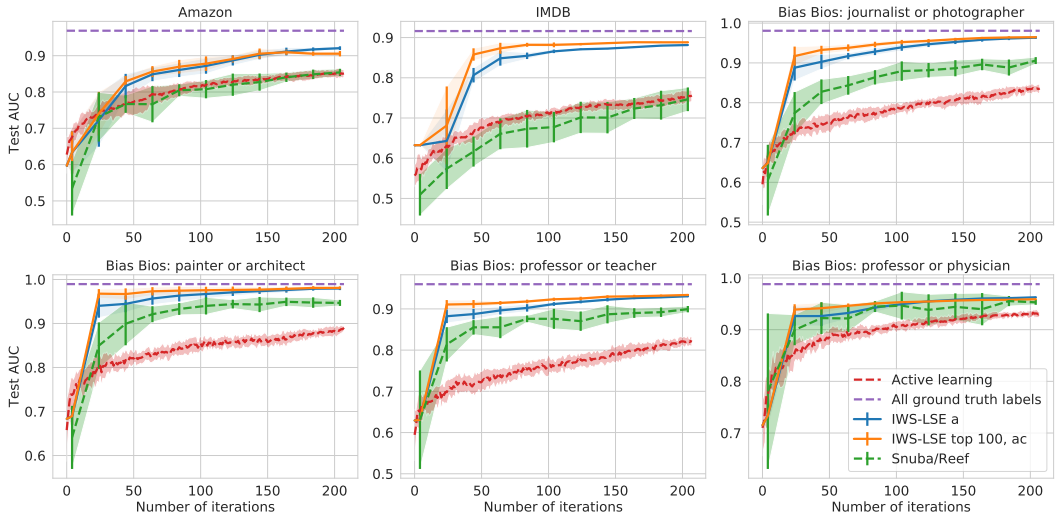

Figure 2: Mean test set AUC of classifiers vs. number of iterations. IWS-LSE is compared to active learning, Snuba, and training using all ground truth labels of the training dataset. Note that, while one iteration corresponds to one expert label, a comparison of effort needed to answer each type of query (label for one sample vs. label for one LF) will vary by application.

a uni-gram vocabulary over all documents and discard very high and low frequency terms. We then exhaustively generate LFs from an LF family z_ϕ which outputs a target label if a uni-gram appears in a document, where ϕ specifies the uni-gram and target label. We also evaluated combinations of higher-order n-grams, but did not observe a significant change in performance.

4.1 Oracle experiments

Our oracle labels an LF as useful if it has an accuracy of at least 0.7. Measured by test-set AUC of final classifier f, IWS-LSE outperforms all other approaches significantly on five out of six datasets, and matches the best performance also attained by Snuba on one dataset (Fig. 2). IWS-AS (Fig. 4, Appendix) performs similarly well on four datasets, and competitively on the other two. Both IWS approaches outperform active learning by a wide margin on all datasets. IWS also quickly approaches the performance achieved by an end model trained on the full ground truth training labels. We provide IWS-LSE annealing results varying the final set size as well as thresholds r in Appendix A.

4.2 User experiments

Table 1: Median (mean) user response time.

Dataset	Annotate LF	Annotate sample
Amazon	4.2s (8.3s)	7.9s (10.3s)
IMDB	3.2s (6.0s)	19.s (24.3s)

We conduct experiments of IWS-AS with real users on the Amazon and IMDB review sentiment classification tasks. The results demonstrate that users judge high accuracy functions as useful and make few mistakes. In our experiments, users are shown a description of the heuristic (the key term pattern) and the intended label. Users are also shown four snippets of random documents where the LF applied, but are instructed to first consider the LF without inspecting these random samples, and to only consider the examples if necessary. See Appendix A.2 for a screenshot of the query interface and more details regarding the user prompts. The top of Fig. 3 shows that mean test set performance of IWS-AS using LFs obtained from real user feedback closely tracks the simulated oracle performance after about 100 iterations. Fig. 3 further shows the queried LFs and corresponding user responses by their true accuracy vs. their non-abstain votes. To match the mean test AUC of IWS-AS obtained after 200 iterations, active learning (uncertainty sampling) requires about 600 iterations. For the IMDB dataset, to achieve the same mean test AUC of IWS-AS obtained after 200 iterations, active learning requires more than 1000 iterations. For both datasets, the average response time to each query was fast. We conducted a manual labeling exercise of samples for the IMDB and Amazon datasets (Table 1) with real users. Users incorrectly classified 9% of IMDB reviews while taking significantly longer compared to the response times to LF queries. For the Amazon dataset, users mislabeled 2% of samples and were also slower at labeling samples than LFs.

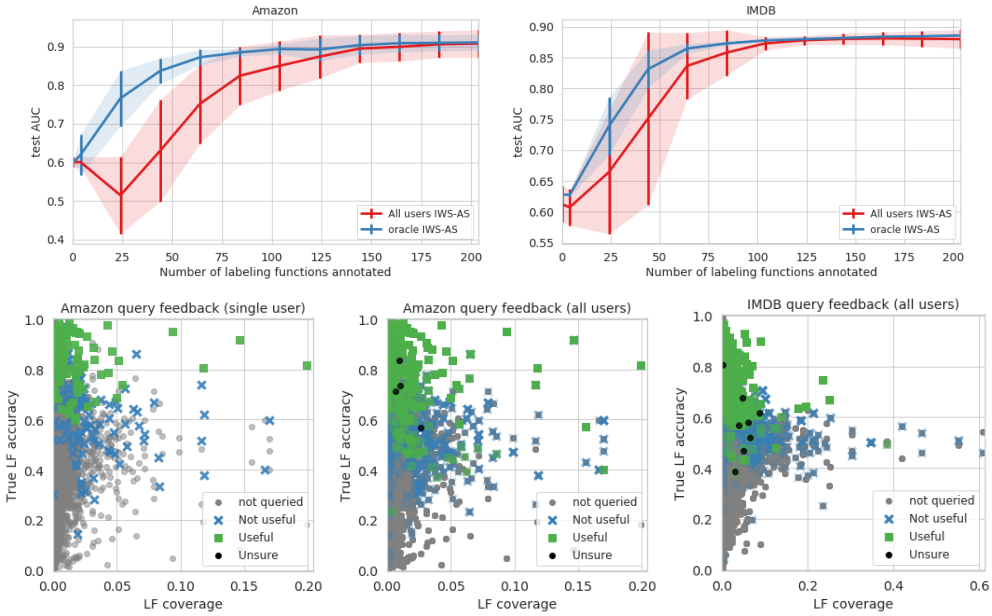

Figure 3: **Human user study.** *Top:* Test set AUC of end classifiers trained on probabilistic labels obtained via IWS-AS. On these datasets, test set performance of humans closely tracks performance using a simulated oracle after 100 iterations. *Bottom:* scatter plots of human responses to queries showing the true LF accuracy vs LF coverage by one user (lower left) and all users (lower middle and lower right). An "unsure" response does not receive a label but is counted as an iteration/query.

The user-study experiments involved nine persons with a computer science background. Neither the true accuracy of each heuristic nor the end model train or test set results were revealed to the users at any stage of the experiment.

5 Conclusion

We have proposed methods for interactively discovering weak supervision sources. Our results show that a small number of expert interactions can suffice to select good weak supervision sources from a large pool of candidates, leading to competitive end classifiers. The proposed methodology shows promise as a way to significantly speed up the process of weak supervision source definition by domain experts as an alternative to devising such sources from scratch. We obtain superior predictive performance on a downstream test set compared to the automatic selection of LFs with Snuba (Varma & Ré, 2018) and standard active learning (where users annotate samples instead of LFs), when measured with respect to the number of user queries. We conduct experiments with real users on two popular benchmark datasets and find that humans recognize and approve high accuracy LFs, yielding models that match performance attainable with a simulated oracle. Further, in our experiments we find that users annotate heuristics faster than individual samples on the datasets we used.

There are limitations to the approaches we propose. LF families that generate heuristics (such as shallow decision trees) appear easy to define for many text classification tasks, and are reusable thereafter for similar tasks and datasets. However, in future work we need to show that this process works similarly well for other data types and tasks. Furthermore, we rely on experts to be able to judge heuristics pre-generated from an LF family in the same way experts judge heuristics they create from scratch, which we cannot assume will be possible for all LF families. The approach proposed in this paper is not meant to replace active learning or manually created weak supervision sources. For many important tasks, standard active learning or crowd sourcing may be entirely sufficient. However, when a large number of labels is necessary to train good models while experts' effort is precious, an interactive framework for weak supervision can be the right choice.

Promising directions for future work are to explore hybrid methods of querying for heuristics and labels for samples, interactive learning of dependencies, and acquisition functions which optimize for additional properties of sets of heuristics such as LF diversity.

REFERENCES

Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. Learning the structure of generative models without labeled data. In *ICML*, pp. 273–282, 2017.

Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, Alex Ratner, Braden Hancock, Houman Alborzi, et al. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proceedings of the 2019 International Conference on Management of Data*, pp. 362–375, 2019.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9368–9377, 2018.

Benedikt Boecking and Artur Dubrawski. Pairwise feedback for data programming. *NeurIPS Workshop on Learning with Rich Experience (LIRE)*, 2019.

Brent Bryan, Robert C Nichol, Christopher R Genovese, Jeff Schneider, Christopher J Miller, and Larry Wasserman. Active learning for identifying function threshold boundaries. In *Advances in neural information processing systems*, pp. 163–170, 2006.

Mayee F Chen, Daniel Y Fu, Frederic Sala, Sen Wu, Ravi Teja Mullapudi, Fait Poms, Kayvon Fatahalian, and Christopher Ré. Train and you'll miss it: Interactive model iteration with weak supervision and pre-trained embeddings. *arXiv preprint arXiv:2006.15168*, 2020.

Vincent Chen, Sen Wu, Alexander J Ratner, Jen Weng, and Christopher Ré. Slice-based learning: A programming model for residual learning in critical data slices. In *Advances in neural information processing systems*, pp. 9392–9402, 2019.

Kashyap Chitta, Jose M Alvarez, and Adam Lesnikowski. Large-scale visual active learning with deep probabilistic ensembles. *arXiv preprint arXiv:1811.03575*, 2018.

Benjamin Cohen-Wang, Stephen Mussmann, Alex Ratner, and Chris Ré. Interactive programmatic labeling for weak supervision. *KDD Data Collection, Curation, and Labeling for Mining and Learning Workshop*, 2019.

Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *Proceedings of the 22nd international conference on World Wide Web*, pp. 285–294, 2013.

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28 (1):20–28, 1979.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman 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*, pp. 120–128, 2019.

Jared A. Dunnmon, Alexander J. Ratner, Khaled Saab, Nishith Khandwala, Matthew Markert, Hersh Sagreiya, Roger Goldman, Christopher Lee-Messer, Matthew P. Lungren, Daniel L. Rubin, and Christopher Re. Cross-modal data programming enables rapid medical machine learning. *Patterns*, 1(2):100019, 2020.

Jason A Fries, Paroma Varma, Vincent S Chen, Ke Xiao, Heliodoro Tejeda, Priyanka Saha, Jared Dunnmon, Henry Chubb, Shiraz Maskatia, Madalina Fiterau, et al. Weakly supervised classification of aortic valve malformations using unlabeled cardiac mri sequences. *Nature communications*, 10 (1):1–10, 2019.

Roman Garnett, Thomas Gärtner, Martin Vogt, and Jürgen Bajorath. Introducing the 'active search'method for iterative virtual screening. *Journal of Computer-Aided Molecular Design*, 29(4):305–314, 2015.

Alkis Gotovos. Active learning for level set estimation. Master's thesis, Eidgenössische Technische Hochschule Zürich, Department of Computer Science,, 2013.

Mathieu Guillame-Bert and Artur Dubrawski. Classification of time sequences using graphs of temporal constraints. *The Journal of Machine Learning Research*, 18(1):4370–4403, 2017.

Yoni Halpern, Youngduck Choi, Steven Horng, and David Sontag. Using anchors to estimate clinical state without labeled data. In *AMIA Annual Symposium Proceedings*, volume 2014, pp. 606. American Medical Informatics Association, 2014.

Braden Hancock, Martin Bringmann, Paroma Varma, Percy Liang, Stephanie Wang, and Christopher Ré. Training classifiers with natural language explanations. In *ACL*, volume 2018, pp. 1884. NIH Public Access, 2018.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pp. 507–517. International World Wide Web Conferences Steering Committee, 2016.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 541–550. Association for Computational Linguistics, 2011.

Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1714–1723. JMLR. org, 2017.

David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pp. 1953–1961, 2011.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Hongwei Li, Bin Yu, and Dengyong Zhou. Error rate analysis of labeling by crowdsourcing. In *ICML Workshop: Machine Learning Meets Crowdsourcing. Atalanta, Georgia, USA*. Citeseer, 2013.

JLEKS Lonardi and Pranav Patel. Finding motifs in time series. In *Proc. of the 2nd Workshop on Temporal Data Mining*, pp. 53–68, 2002.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, June 2011.

Mona Nashaat, Aindrila Ghosh, James Miller, Shaikh Quader, Chad Marston, and Jean-Francois Puget. Hybridization of active learning and data programming for labeling large industrial datasets. In *2018 IEEE International Conference on Big Data (Big Data)*, pp. 46–55. IEEE, 2018.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4763–4771, 2019.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, 29(2):709–730, 2020.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In *NIPS*, pp. 3567–3575, 2016.

Christopher Ré, Feng Niu, Pallavi Gudipati, and Charles Srisuwananukorn. Overton: A data system for monitoring and improving machine-learned products. *In Proceedings of the 10th Annual Conference on Innovative Data Systems Research*, 2020.

Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 148–163. Springer, 2010.

Khaled Saab, Jared Dunnmon, Christopher Ré, Daniel Rubin, and Christopher Lee-Messer. Weak supervision as an efficient approach for automated seizure detection in electroencephalography. *npj Digital Medicine*, 3(1):1–12, 2020.

Frederic Sala, Paroma Varma, Jason Fries, Daniel Y Fu, Shiori Sagawa, Saelig Khattar, Ashwini Ramamoorthy, Ke Xiao, Kayvon Fatahalian, James Priest, and Chris Re. Multi-resolution weak supervision for sequential data. In *NeurIPS'19*, NeurIPS'19, 2019.

Paroma Varma and Christopher Ré. Snuba: automating weak supervision to label training data. *Proceedings of the VLDB Endowment*, 12(3):223–236, 2018.

Paroma Varma, Bryan He, Dan Iter, Peng Xu, Rose Yu, Christopher De Sa, and Christopher Ré. Socratic learning: Augmenting generative models to incorporate latent subsets in training data. *arXiv preprint arXiv:1610.08123*, 2016.

Paroma Varma, Bryan D He, Payal Bajaj, Nishith Khandwala, Imon Banerjee, Daniel Rubin, and Christopher Ré. Inferring generative model structure with static analysis. In *Advances in neural information processing systems*, pp. 240–250, 2017.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Ré. Learning dependency structures for weak supervision models. *ICML*, 2019.

Andrea Zanette, Junzi Zhang, and Mykel J Kochenderfer. Robust super-level set estimation using gaussian processes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 276–291. Springer, 2018.

Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. In *Advances in neural information processing systems*, pp. 1260–1268, 2014.

A Additional Details on Experiments and Results

Datasets We use three publicly available datasets [2] and define six binary text classification tasks. We use a subset of the Amazon Review Data (He & McAuley, 2016) for sentiment classification, aggregating all categories with more than $100k$ reviews from which we sample $200k$ reviews and split them into $160k$ training points and $40k$ test points. We use the IMDB Movie Review Sentiment dataset (Maas et al., 2011) which has $25k$ training samples and $25k$ test samples. In addition, we use the Bias in Bios (De-Arteaga et al., 2019) dataset from which we create binary classification tasks to distinguish difficult pairs among frequently occurring occupations. Specifically, we create the following subsets with equally sized train and test sets: journalist or photographer ($n = 32\,258$), professor or teacher ($n = 24\,588$), painter or architect ($n = 12\,236$), professor or physician ($n = 54\,476$).

Implementation Details Our **probabilistic ensemble** in IWS, which is used in all acquisition functions to learn $p(u_j = 1|Q_t)$, is a bagging ensemble of $s = 50$ multilayer perceptrons with two hidden layers of size 10, RELU activations, sigmoid output and logarithmic loss. To create features for the p candidate LFs in $\mathcal{L}$, we use singular value decomposition (SVD) to project from n to $d' = 150$. Thus, at iteration t, given a query dataset $Q_{t-1} = \{(\lambda_j, u_j)\}_{j=1}^{t-1}$, the ensemble is trained on pairs $\{(\tau(\lambda_j), u_j)\}_{j=1}^{t-1}$ where $\tau(\lambda_j)$ are the SVD features and u_j the binary expert responses. The output of the ensemble on LFs not in the query dataset is used to compute $\sigma_j(Q_{t-1}) = \sqrt{\mathrm{Var}[g^{-1}(p(u_j = 1|Q_{t-1}))]}$ and $\mu_j(Q_{t-1}) = \mathbb{E}[g^{-1}(p(u_j = 1|Q_{t-1}))]$. While g, which maps α_j to v_j, could be fine-tuned from data, we set g as the identity function in our experiments, which we find works well empirically. Finally, to allow human experts to express some level of confidence about their decision on u_j, we also collect corresponding uncertainty weights $b_j \in \{1, 0.5\}$, and we multiply the contribution to the loss of each u_j by the respective weight b_j. Users can also skip queries if they are unsure, indicated in black in Fig. 3. These unsure responses are still counted as an iteration/query in our plots.

Our downstream **end classifier** f is a multilayer perceptron with two hidden layers of size 20 and RELU activations, sigmoid output and logarithmic loss. Each model in the ensemble as well as f are optimized using Adam (Kingma & Ba, 2014). To avoid conflating performance gains by using pre-trained representations, we fit the end models f to low dimensional projections of a large bag-of-words matrix via truncated Singular Value Decomposition (SVD), fixing the embedding size to $d = 300$. We repeat each experiment ten times. We assume that the class balance is known when fitting the label model, as common in related work. When class balance is unknown, (Ratner et al., 2019) discuss an unsupervised approach to estimate it.

The first 8 iterations of IWS are initialized with queries of four LFs known to have accuracy between 0.7 and 0.75 drawn at random and four randomly drawn LFs with arbitrary accuracy. Subsequently, IWS chooses the next LFs to query. Active learning is initialized with the same number of known samples.

A.1 Full IWS Results and All Baselines

Fig. 4 provides the full results of all IWS settings to all baselines, including IWS with a random acquisition function (IWS-random). IWS LSE-a corresponds to scenario (A) where there are no restrictions on the size of the final set and any LF can be included]. IWS LSE-ac corresponds to scenario (B) where the final set LFs is limited in size (e.g. due to computational considerations) but any LF can be included. IWS-AS corresponds to Scenario (C), where only LFs in our query dataset Q_t can be used, which are LFs that were inspected and validated by experts, e.g. due to security or legal considerations.

[2] Amazon: `https://nijianmo.github.io/amazon/index.html`, IMDB: `https://ai.stanford.edu/~amaas/data/sentiment/`, BiasBios: `http://aka.ms/biasbios`

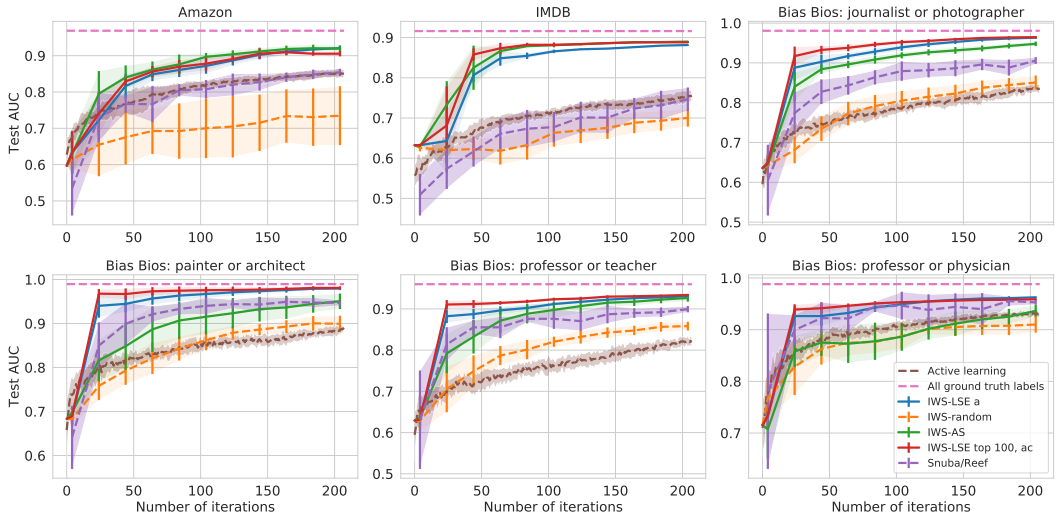

Figure 4: Mean test set AUC vs. number of iterations for classifiers trained on probabilistic labels. IWS-LSE and IWS-AS are compared to active learning, Snuba, training on all labels, and IWS with a random acquisition function. Note that, while one iteration corresponds to one expert label, a comparison of effort needed to answer each type of query (label for sample vs label for LF) will vary by application.

A.2 User experiments

A.2.1 Interface and experiment prompt

Fig. 5 shows an example of the prompt that was shown to users at each iteration of the IWS user experiments. Before the experiment started, users were first instructed on the interface they would see and the task they would be given, i.e. to label a heuristic as good if they would expect it to label samples at better than random accuracy and as bad otherwise. Users were also instructed about the response options, including the option to not answer a query if they were unsure ('I don't know').

Users were given a description of the classification task and domain of the documents for which heuristics were being acquired. Users were also provided with a description of the heuristic generated which labeled samples with a target label if a document contained a certain term. Finally, users were given two examples of a better than random heuristic, and two examples of an arbitrary heuristic.

During the experiments, users were also provided with 4 random examples of documents documents where the queried LF applied. Users were instructed to first consider the LF without inspecting these random samples, and to only consider the examples if necessary.

While LFs receive binary labels, users were allowed to express uncertainty about their decision, which was used as a sample weight (1 if certain else 0.5) of LFs during training of the probabilistic model of user feedback.

A.2.2 Additional statistics of user experiments

In Fig. 6 we provide more details about our user experiments. The top row displays the test set performance of downstream model f for each individual user. The middle row shows how the number of LFs determined by the user to be useful $u = 1$ increases with the number of iterations. The bottom row displays the maximum positive correlation between a new LF with $u = 1$ at iteration t and all previously accepted LFs with $u = 1$ up to iteration t. Note that we take abstains into account by computing correlation between and LF i and j only on entries where at least one of them is nonzero

A.2.3 Heuristics found by IWS-AS

We here provide some concrete examples of heuristics found during the IWS-AS procedure, that real users annotated as useful during the experiments. For the IMDB dataset:

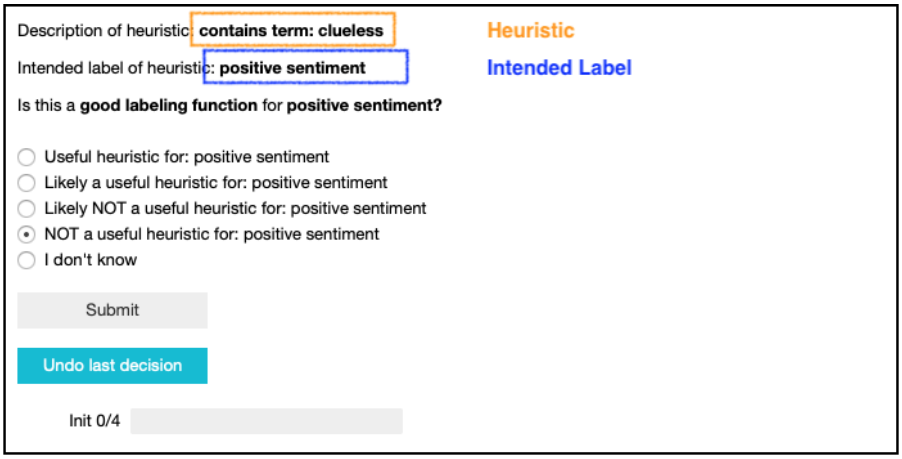

Figure 5: An example of the prompt and answer options that users were shown during the user study. Before starting the experiment, users were provided with a description of the task and the labeling function family.

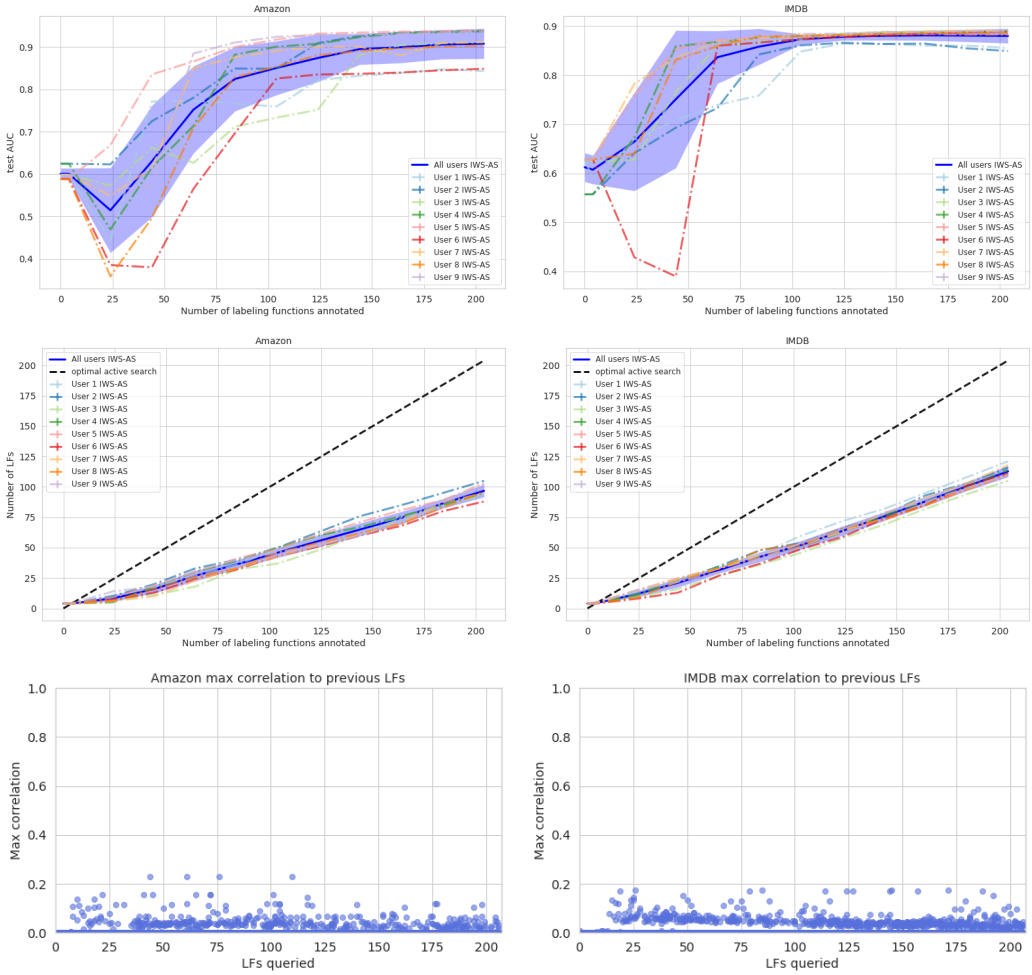

Figure 6: Test AUC vs. IWS iteration shown for individual user experiments with IWS-AS (*top*). Number of LFs labeled as useful vs. IWS iterations (*middle*). Maximum correlation to previously accepted LFs vs. number of iterations (*bottom*).

- Ten terms most frequently annotated as useful by users
  - Positive: beautiful, wonderful, perfect, enjoyed, amazing, brilliant, fantastic, superb, excellent, masterpiece.
  - Negative: worst, poor, awful, bad, waste, terrible, horrible, boring, crap, stupid.
- Ten terms annotated as useful with highest underlying accuracy
  - Positive: flawless, superbly, perfection, wonderfully, captures, refreshing, breathtaking, delightful, beautifully, underrated.
  - Negative: stinker, dreck, unwatchable, unfunny, waste, atrocious, pointless, redeeming, laughable, lousy.
- Ten terms annotated as useful by users, selected at random
  - Positive: favorites, joy, superbly, delight, wonderfully, art, intelligent, terrific, light, finest.
  - Negative: reason, failed, atrocious, decent, unfunny, lame, ridiculous, mistake, worst, dull.

For the Amazon dataset:

- Ten terms most frequently annotated as useful by users
  - Positive: wonderful, beautiful, amazing, fantastic, favorite, awesome, love, best, perfect, easy.
  - Negative: worst, terrible, horrible, awful, worse, boring, poor, bad, waste, garbage.
- Ten terms annotated as useful with highest underlying accuracy
  - positive: compliments, delightful, pleasantly, stars, captivating, excellent, awesome, beautifully, comfy, perfect.
  - negative: poorly, worthless, disappointing, refund, waste, yuck, garbage, unusable, useless, junk.
- Ten terms annotated as useful by users, selected at random
  - Positive: interesting, beautifully, value, loves, strong, expected, gorgeous, perfectly, durable, great.
  - Negative: sent, zero, money, mess, crap, refund, wasted, joke, unusable, beware.

A.3 ORACLE EXPERIMENTS ABLATION

In this section we provide results for all datasets of ablation experiments we conducted. The IWS-LSE algorithm requires us to set a threshold above random accuracy r around which our model aims to partition the dataset. Fig. 7 provides results for different threshold settings and shows that the algorithm performs well across a wide range. While there is no clear, distinct performance difference discernible, the results suggest that a high threshold r 0.8 causes the algorithm to under-perform, potentially as it stifles exploration of LFs.

In Scenario (B), which corresponds to the LSE-ac and where we aim to find a final set of LFs of limited size, Fig. 8 shows that a wide range (50 to 200) of final set sizes produce good results.

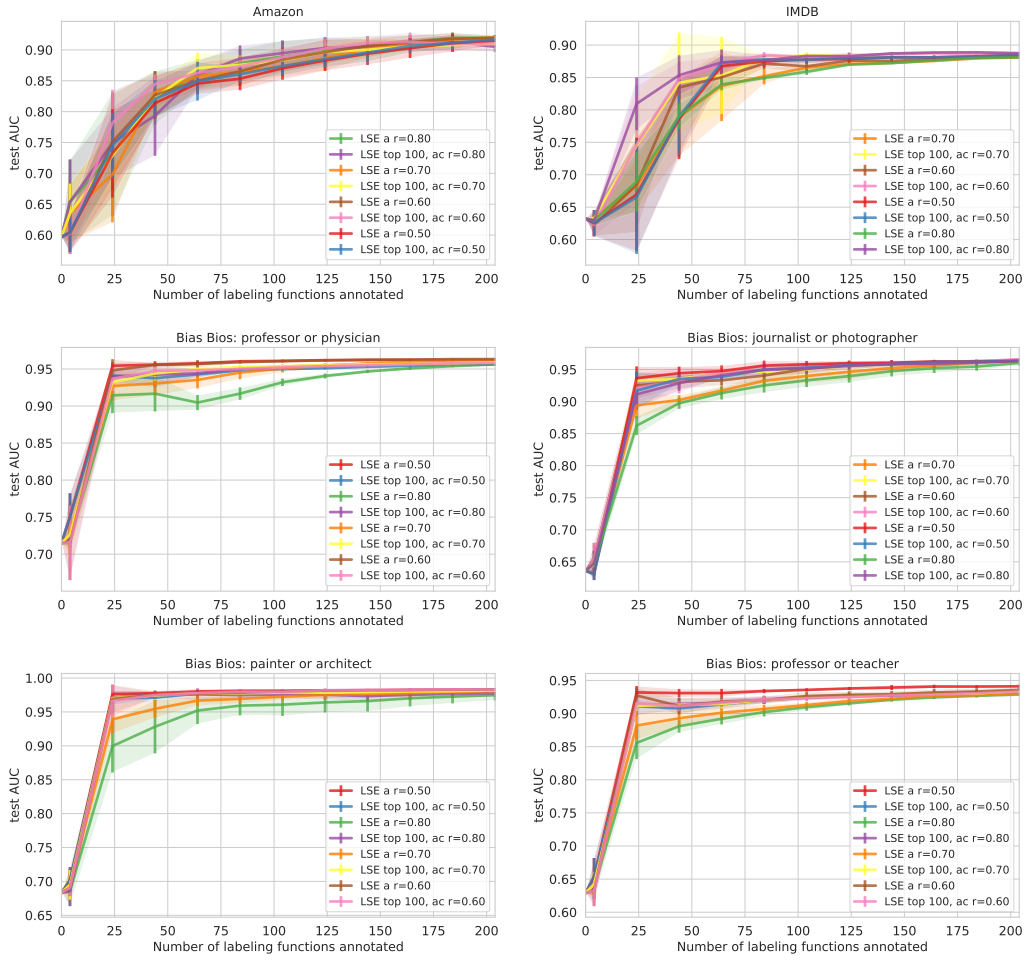

Figure 7: IWS-LSE annealing plots for varying thresholds r which we use to partitions our set of LFs. On all datasets test set performance is very similar after around 100 iterations, showing that a wide range of such thresholds leads to good test set performance.

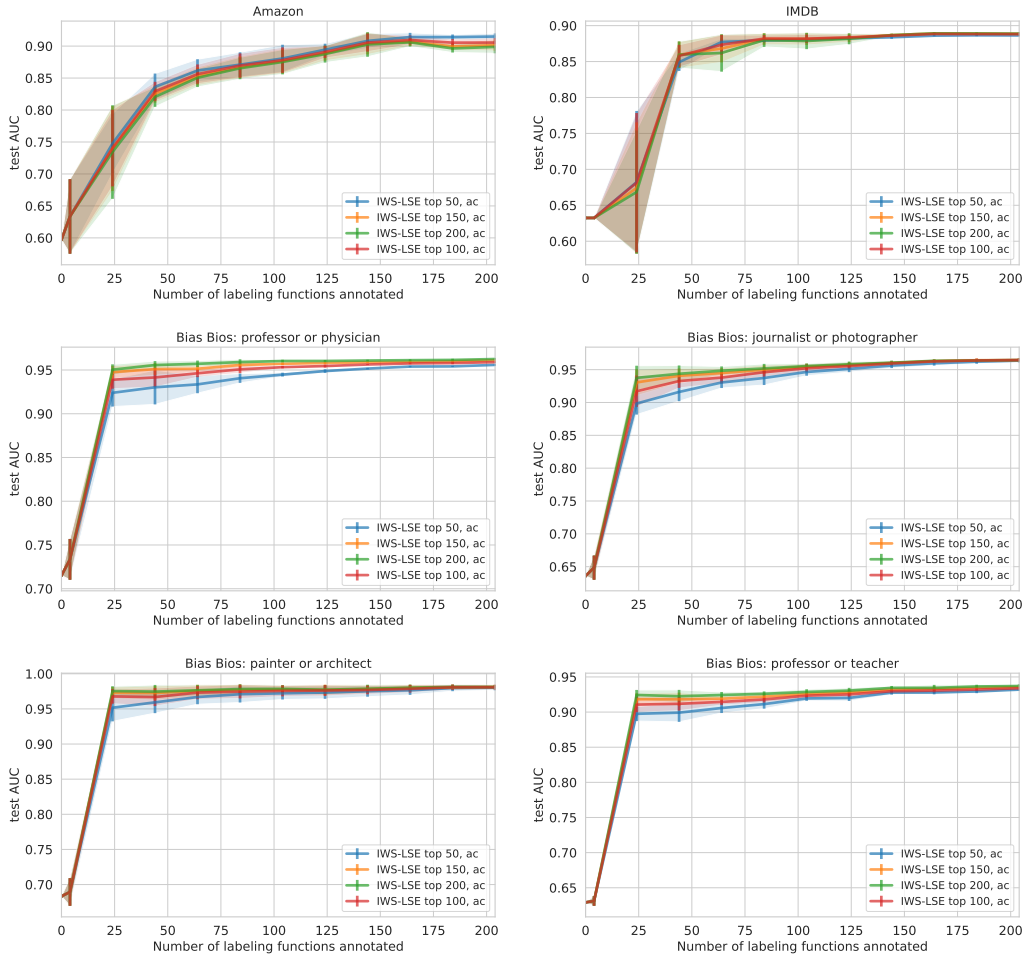

Figure 8: IWS-LSE ac annealing plots for varying set sizes. Note that this setting takes LF coverage into account to rank LFs according to $(2\alpha_j - 1) * \hat{l}_j$ where $\alpha_j, \hat{l}_j$ are the estimated LF accuracy and observed LF coverage.

B ON LABELING FUNCTION PROPERTIES

In this section we analyze how LF accuracy and LF propensity (i.e. non-abstain behavior) influence the estimate of the true latent class label Y^*. We focus on the binary classification case for simplicity. Assume each data point $x \in \mathcal{X}$ has a latent class label $y^* \in \mathcal{Y} = \{-1, 1\}$. Given n unlabeled, i.i.d. data points $X = \{x_i\}_{i=1}^n$, our goal is to train a classifier $f : \mathcal{X} \to \mathcal{Y}$ such that $f(x) = y^*$. As in Ratner et al. (2016) a user provides m LFs $\{\lambda_j\}_{j=1}^m$, where $\lambda_j : \mathcal{X} \to \mathcal{Y} \cup \{0\}$ noisily label the data with $\lambda_j(x) \in \{-1, 1\}$ or abstain with $\lambda_j(x) = 0$. The corresponding LF output matrix is $\Lambda \in \{-1, 0, 1\}^{n \times m}$, where $\Lambda_{i,j} = \lambda_j(x_i)$.

We define a factor graph as proposed in Ratner et al. (2016; 2020) to obtain probabilistic labels by modeling the LF accuracies via factor $\phi_{i,j}^{Lab}(\Lambda, Y) \triangleq \mathbb{1}\{\Lambda_{ij} \neq 0\}$ and labeling propensity by factor $\phi_{i,j}^{Acc}(\Lambda, Y) \triangleq \mathbb{1}\{\Lambda_{ij} = y_i\}$, and for simplicity assume LFs are independent conditional on Y. The label model is defined as

$$p_\theta(Y, \Lambda) \triangleq Z_\theta^{-1} \exp\left(\sum_{i=1}^n \theta^\top \phi_i(\Lambda_i, y_i)\right), \tag{8}$$

where Z_θ is a normalizing constant and $\phi_i(\Lambda_i, y_i)$ defined to be the concatenation of the factors for all LFs $j = 1, \ldots, m$ for sample i. Also, let $\theta = (\theta^{(1)}, \theta^{(2)})$ where $\theta^{(1)}, \theta^{(2)} \in \mathbb{R}^m$. Here, $\theta^{(1)}$ are the canonical parameters for the LF accuracies, and $\theta^{(2)}$ the canonical parameters for the LF propensities.

To estimate the label model parameters, we generally obtain the maximum marginal likelihood estimate via the (scaled) log likelihood

$$l(\theta) = 1/n \sum_{i=1}^n \log\left(\sum_{y \in \mathcal{Y}} p(\Lambda_i, y|\theta)\right).$$

Let finite $\hat{\theta} \in \mathbb{R}^{2m}$ be such an estimate. We use $p_{\hat{\theta}}(y|\Lambda_i)$ to obtain probabilistic labels:

$$p_{\hat{\theta}}(y_i = k|\Lambda_i) = \frac{p_{\hat{\theta}}(y_i = k, \Lambda_i)}{p_{\hat{\theta}}(\Lambda_i)} \tag{9}$$

$$= \frac{\exp(\sum_{j=1}^m \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), k))}{\sum_{\tilde{y} \in \mathcal{Y}} \exp(\sum_{j=1}^m \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), \tilde{y}))}. \tag{10}$$

Note that the label estimate does not directly depend on $\theta^{(2)}$. Further, note that the denominator is the same over different label possibilities. Finally, note that even in a case where we include correlation factors $\phi_{i,j,k}^{Corr}(\Lambda, Y) = \mathbb{1}\{\Lambda_{ij} = \Lambda_{ik}\}, (j, k) \in C$ in the model above with C as a set of potential dependencies, the probabilistic label will only directly depend on the estimated canonical accuracy parameters $\theta^{(1)}$. In the binary classification case, which we assume here, the expression simplifies further. For $k \in \{-1, 1\}$:

$$p_{\hat{\theta}}(y_i = k|\Lambda_i) = \frac{\exp(\sum_{j=1}^m \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), k))}{\sum_{\tilde{y} \in \{-1,1\}} \exp(\sum_{j=1}^m \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), \tilde{y}))} \tag{11}$$

$$= \frac{1}{1 + \exp(\sum_{j=1}^m \hat{\theta}_j^{(1)} (\phi^{Acc}(\lambda_j(x_i), -k) - \phi^{Acc}(\lambda_j(x_i), k)))} \tag{12}$$

$$= \sigma(\sum_{j=1}^m \hat{\theta}_j^{(1)} (\phi^{Acc}(\lambda_j(x_i), k) - \phi^{Acc}(\lambda_j(x_i), -k))), \tag{13}$$

where σ deontes the sigmoid function. The probabilistic labels are a softmax in the multi-class classification case and, as shown above, simplify to a sigmoid in the binary case. An absolute label prediction $\hat{y} \in \{-1, 1\}$ is therefore simply a function of

$$\hat{y}_i = \arg\max_{\tilde{y} \in \mathcal{Y}} p_{\hat{\theta}}(y_i = \tilde{y}|\Lambda_i) = \arg\max_{\tilde{y} \in \mathcal{Y}} \sum_{j=1}^m \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), \tilde{y}).$$

We now introduce some assumptions on the accuracy and error probabilities of labeling functions, similar to the Homogenous Dawid-Skene model (Dawid & Skene, 1979; Li et al., 2013) in crowd sourcing, where label source accuracy is the same across classes and errors are evenly divided with probability mass *independent* of the true class.

Under these assumptions, we denote by $\alpha_j = P(\lambda_j(x) = y^* | \lambda_j(x) \neq 0)$ the accuracy of LF j. Further, we denote by $l_j = P(\lambda_j(x) \neq 0)$ the labeling propensity of j, i.e. how frequently LF j does not abstain. The observed LF propensity is also referred to as LF coverage in the related literature. We recall theorem 3.1:

Theorem 3.1. *Assume a binary classification setting, m independent labeling functions with accuracy $\alpha_j \in [0, 1]$ and labeling propensity $l_j \in [0, 1]$. For a label model as in Eq. (1) with given label model parameters $\hat{\theta} \in \mathbb{R}^{2m}$, and for any $i \in \{1, \ldots, n\}$,*

$$P(\hat{y}_i = y_i^*) \geq 1 - \exp\left(-\frac{(\sum_{j=1}^{m} \hat{\theta}_j^{(1)}(2\alpha_j - 1)l_j)^2}{2||\hat{\theta}^{(1)}||^2}\right)$$

where $\hat{\theta}^{(1)}$ are the m weights of ϕ^{Acc}, and $\hat{y}_i \in \{-1, 1\}$ is the label model estimate for y_i^*.

Proof. Assume that we use the label model to obtain a label estimate $\hat{y}_i \in \{-1, 1\}$. As shown in Eq. (10), the prediction rule in that case is

$$\hat{y}_i = \underset{\tilde{y} \in \{-1,1\}}{\arg\max} \sum_{j=1}^{m} \hat{\theta}_j^{(1)} \phi^{Acc}(\lambda_j(x_i), \tilde{y}).$$

Define by $\lambda(x) = (\lambda_1(x), \ldots, \lambda_m(x))$ the vector of the $j = 1, \ldots, m$ LF outputs on x. Further, we define for $k \in \{-1, 1\}$:

$$\begin{aligned} V_{\hat{\theta}}(\lambda(x), k) &= \sum_{j=1}^{m} \hat{\theta}_j^{(1)} (\phi^{Acc}(\lambda_j(x), k) - \phi^{Acc}(\lambda_j(x), -k)) \\ &= \sum_{j=1}^{m} \hat{\theta}_j^{(1)} \left(\mathbb{1}\{\lambda_j(x) = k\} - \mathbb{1}\{\lambda_j(x) = -k\}\right). \end{aligned}$$

For the two label options $k \in \{-1, 1\}$, we have

$$V_{\hat{\theta}}(\lambda(x), 1) = \sum_{j=1}^{m} \hat{\theta}_j^{(1)} \left(\mathbb{1}\{\lambda_j(x) = 1\} - \mathbb{1}\{\lambda_j(x) = -1\}\right) = \sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x)$$

and

$$V_{\hat{\theta}}(\lambda(x), -1) = \sum_{j=1}^{m} \hat{\theta}_j^{(1)} \left(\mathbb{1}\{\lambda_j(x) = -1\} - \mathbb{1}\{\lambda_j(x) = 1\}\right) = -\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x).$$

Now, we want to obtain a bound on the probability that the label estimate $\hat{y}_i$ is equal to the true label. We have

$$\begin{aligned} P(\hat{y}_i = y_i^*) &= P(y_i^* = 1)P(\hat{y}_i = 1 | y_i^* = 1) + P(y_i^* = -1)P(\hat{y}_i = -1 | y_i^* = -1) \\ &= P(y_i^* = 1)P(\hat{y}_i = 1 | y_i^* = 1) + (1 - P(y_i^* = 1))P(\hat{y}_i = -1 | y_i^* = -1). \end{aligned}$$

Note that

$$P(\hat{y}_i = 1 | y_i^* = 1) = P(V_{\hat{\theta}}(\lambda(x_i), 1) > 0 | y_i^* = 1) = P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x) > 0 | y_i^* = 1),$$

and that

$$P(\hat{y}_i = -1|y_i^* = -1) = P(V_{\hat{\theta}}(\lambda(x_i), -1) > 0|y_i^* = -1) = P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x_i) < 0|y_i^* = -1).$$

We therefore have

$$P(\hat{y}_i = y_i^*) = P(y_i^* = 1)P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x_i) > 0|y_i^* = 1) + (1 - P(y_i^* = 1))P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x_i) < 0|y_i^* = -1).$$

Now we define $\xi_{ij} = \hat{\theta}_j^{(1)} \lambda_j(x_i)$ and we know that $\xi_j \in [-|\hat{\theta}_j^{(1)}|, |\hat{\theta}_j^{(1)}|]$. Given the Dawid-Skene model assumptions stated previously, we have

$$\mathbb{E}[\sum_{j=1}^{m} \xi_{ij}|y_i^* = 1] = \sum_{j=1}^{m} \mathbb{E}[\xi_{ij}|y_i^* = 1] = \sum_{j=1}^{m} \hat{\theta}_j^{(1)} l_j(2 * \alpha_j - 1),$$

and

$$\mathbb{E}[\sum_{j=1}^{m} \xi_{ij}|y_i^* = -1] = \sum_{j=1}^{m} \mathbb{E}[\xi_{ij}|y_i^* = -1] = -\sum_{j=1}^{m} \hat{\theta}_j^{(1)} l_j(2 * \alpha_j - 1).$$

Now, using Hoeffding's inequality and assuming independent labeling functions, we can bound $P(\hat{y}_i = 1|y_i^* = 1)$ and $P(\hat{y}_i = -1|y_i^* = -1)$ from below:

$$\begin{aligned}
P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x_i) > 0|y_i^* = 1) &= P(\sum_{j=1}^{m} \xi_{ij} > 0|y_i^* = 1) \\
&= P(\sum_{j=1}^{m} \xi_{ij} - \mathbb{E}[\sum_{j=1}^{m} \xi_{ij}|y_i^* = 1] > -\sum_{j=1}^{m} \hat{\theta}_j^{(1)} l_j(2 * \alpha_j - 1)\ |y_i^* = 1) \\
&\geq 1 - \exp\left(-\frac{(\sum_{j=1}^{m} \hat{\theta}_j^{(1)}(2\alpha_j - 1)l_j)^2}{2||\hat{\theta}^{(1)}||^2}\right),
\end{aligned}$$

and

$$\begin{aligned}
P(\sum_{j=1}^{m} \hat{\theta}_j^{(1)} \lambda_j(x_i) < 0|y_i^* = -1) &= P(\sum_{j=1}^{m} \xi_{ij} < 0|y_i^* = -1) \\
&= P(\sum_{j=1}^{m} \xi_{ij} - \mathbb{E}[\sum_{j=1}^{m} \xi_{ij}|y_i^* = -1] < \sum_{j=1}^{m} \hat{\theta}_j^{(1)} l_j(2 * \alpha_j - 1)\ |y_i^* = -1) \\
&\geq 1 - \exp\left(-\frac{(\sum_{j=1}^{m} \hat{\theta}_j^{(1)}(2\alpha_j - 1)l_j)^2}{2||\hat{\theta}^{(1)}||^2}\right).
\end{aligned}$$

Finally we have

$$\begin{aligned}
P(\hat{y}_i = y_i^*) &= P(y_i^* = 1)P(\hat{y}_i = 1|y_i^* = 1) + (1 - P(y_i^* = 1))P(\hat{y}_i = -1|y_i^* = -1) \\
&\geq 1 - \exp\left(-\frac{(\sum_{j=1}^{m} \hat{\theta}_j^{(1)}(2\alpha_j - 1)l_j)^2}{2||\hat{\theta}^{(1)}||^2}\right).
\end{aligned}$$

□

What do the theorem and the quantities analyzed in this section indicate?

- The trade-off between LF accuracy and LF propensity (also referred to as LF coverage) is captured by $(2\alpha_j - 1)l_j$ which allows us to rank LFs if we know the accuracy α_j or can estimate it and use the observed, empirical coverage as an estimate of l_j.

- Not surprising, the relation between sign(θ_j) and α_j is important. A better than random LF j should have a positive θ_j. This indicates that a gap to randomness is important if we cannot guarantee that we learn θ_j well, to reduce the chance of obtaining a negative θ_j for better than random LF j, or vice versa.
- Note how the label estimates are obtained in Eq. (10). Increasing the θ_j of am LF also effectively means reducing the impact other LFs have on a prediction. In particular when θ estimates are imperfect, a gap to random accuracy of α_j is important to obtain good label estimates. Intuitively, we do not want to add excessive noise by including LFs close to random unless we can guarantee that their parameter estimate is appropriately low and has the correct sign.

B.1 What are the optimal θ parameters?

Here we discuss the optimal θ parameters for two cases: first for when we assume that the label model factor graph consists only of accuracy factors and second for when this label model also takes LF propensity into account.

B.1.1 Assuming accuracy factors only

In the previous section we assumed that we are given estimated θ parameters. Naturally, we may next ask ourselves what the optimal theta parameters are. Let us start with a simple case. Let Y, H be random variables of the class variable and vector of LFs, respectively. Assume that we only model LF accuracy and define $\mathbb{1}_{\{\lambda(x)=y\}}$ to be the element-wise indicator function. Also, assume that the true distribution can be expressed by the following model:

$$P(Y = y, H = \lambda(x); \theta) = \frac{1}{Z} \exp\left(\theta^\top \mathbb{1}_{\{\lambda(x)=y\}}\right) \tag{14}$$

$$Z = \sum_{y,\lambda(x)} \exp\left(\theta^\top \mathbb{1}_{\{\lambda(x)=y\}}\right). \tag{15}$$

Note that Z can be written as

$$\begin{aligned} Z &= \sum_{y,\lambda(x)|\lambda(x)_j=y} \exp\left(\theta^\top \mathbb{1}_{\{\lambda(x)=y\}}\right) + \sum_{y,\lambda(x)|\lambda(x)_j\neq y} \exp\left(\theta^\top \mathbb{1}_{\{\lambda(x)=y\}}\right) \\ &= \sum_{y,\lambda(x)|\lambda(x)_j=y} \exp\left(\theta_j + \theta_{-i}^\top \mathbb{1}_{\{\lambda(x)_{-i}=y\}}\right) + \sum_{y,\lambda(x)|\lambda(x)_j\neq y} \exp\left(\theta_{-i}^\top \mathbb{1}_{\{\lambda(x)_{-i}=y\}}\right) \\ &= (e^{\theta_j} + L) \sum_{y,\lambda(x)_{-i}} \exp\left(\theta_{-i}^\top \mathbb{1}_{\{\lambda(x)_{-i}=y\}}\right) \end{aligned}$$

where L is the number of classes. Now, note that

$$P(H_j = Y) = \sum_{y,\lambda(x)|\lambda(x)_j=y} P(Y = y, H = \lambda(x)) \tag{16}$$

$$= \sum_{y,\lambda(x)|\lambda(x)_j=y} \frac{1}{Z} \exp\left(\theta^\top \mathbb{1}_{\{\lambda(x)=y\}}\right) \tag{17}$$

$$= \frac{e^{\theta_j}}{Z} \sum_{y,\lambda(x)_{-i}} \exp\left(\theta_{-i}^\top \mathbb{1}_{\{\lambda(x)_{-i}=y\}}\right) \tag{18}$$

Thus, $P(H_j = Y) = \frac{e^{\theta_j}}{e^{\theta_j}+L}$. Which implies

$$\theta_j = \ln\left(\frac{P(H_j = Y)L}{1 - P(H_j = Y)}\right). \tag{19}$$

Further, note that $P(H_j = Y) = P(H_j = Y, H_j \neq 0) = P(H_j = Y | H_j \neq 0)P(H_j \neq 0)$ is a combination of an LF's accuracy and label propensity.

B.1.2 ASSUMING ACCURACY FACTORS AND LF PROPENSITY FACTORS

We now analyze a slightly more elaborate label model which also models LF propensity. Again, Let Y, H denote the random variables for class labels and LFs, respectively.

$$P(Y = y, H = \lambda(x); \theta,) = \frac{1}{Z} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) \tag{20}$$

$$Z = \sum_{y,\lambda(x)} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) \tag{21}$$

Z can be written as

$$\begin{aligned}
Z &= \sum_{y,\lambda(x)|\lambda(x)_j=y} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) + \sum_{y,\lambda(x)|\lambda(x)_j\neq y} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) \\
&= \sum_{y,\lambda(x)|\lambda(x)_j=y} \exp\left(\theta_j^{(1)} + \theta_j^{(2)} + \theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq 0\}}\right) \\
&+ \sum_{y,\lambda(x)|\lambda(x)_j\neq y, \lambda(x)_j\neq 0} \exp\left(\theta_j^{(2)} + \theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq 0\}}\right) \\
&+ \sum_{y,\lambda(x)|\lambda(x)_j=0} \exp\left(\theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq 0\}}\right) \\
&= (e^{\theta_j^{(1)}+\theta_j^{(2)}} + (L-1)e^{\theta_j^{(2)}} + 1) \sum_{y,\lambda(x)_{-i}} \exp\left(\theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq 0\}}\right)
\end{aligned}$$

The likelihood of a correct vote is given by

$$\begin{aligned}
P(H_j = Y) &= \sum_{y,\lambda(x)|\lambda(x)_j=y} P(Y = y, H = \lambda(x)) && (22) \\
&= \sum_{y,\lambda(x)|\lambda(x)_j=y} \frac{1}{Z} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) && (23) \\
&= \frac{e^{\theta_j^{(1)}+\theta_j^{(2)}}}{Z} \sum_{y,\lambda(x)_{-i}} \exp\left(\theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq y\}}\right) && (24)
\end{aligned}$$

Further note that

$$\begin{aligned}
P(H_j = 0) &= \sum_{y,\lambda(x)|\lambda(x)_j=0} P(Y = y, H = \lambda(x)) \\
&= \sum_{y,\lambda(x)|\lambda(x)_j=0} \frac{1}{Z} \exp\left(\theta^{(1)\top} \mathbb{1}_{\{\lambda(x)=y\}} + \theta^{(2)\top} \mathbb{1}_{\{\lambda(x)\neq 0\}}\right) \\
&= \frac{1}{Z} \sum_{y,\lambda(x)_{-i}} \exp\left(\theta_{-i}^{(1)\top} \mathbb{1}_{\{\lambda(x)_{-i}=y\}} + \theta_{-i}^{(2)\top} \mathbb{1}_{\{\lambda(x)_{-i}\neq y\}}\right) \\
&= \frac{1}{e^{\theta_j^{(1)}+\theta_j^{(2)}} + (L-1)e^{\theta_j^{(2)}} + 1}
\end{aligned}$$

So we can express

$$\begin{aligned}
P(H_j = Y | H_j \neq 0) &= \frac{P(H_j = Y, H_j \neq 0)}{P(H_j \neq 0)} = \frac{P(H_j = Y)}{P(H_j \neq 0)} = \frac{P(H_j = Y)}{1 - P(H_j = 0)} && (25) \\
&= \frac{e^{\theta_j^{(1)}+\theta_j^{(2)}}}{e^{\theta_j^{(1)}+\theta_j^{(2)}} + (L-1)e^{\theta_j^{(2)}}} && (26) \\
P(H_j \neq 0) &= 1 - \frac{1}{e^{\theta_j^{(1)}+\theta_j^{(2)}} + (L-1)e^{\theta_j^{(2)}} + 1} && (27)
\end{aligned}$$

Solving for $\theta_j^{(1)}$ and $\theta_j^{(2)}$, we find

$$\theta_j^{(1)*} = \ln\left(\frac{(L-1)\alpha_j}{1-\alpha_j}\right) \tag{28}$$

$$\theta_j^{(2)*} = \ln\left(\frac{(1-\alpha_j)l_j}{(L-1)(1-l_j)}\right), \tag{29}$$

where $\alpha_j = P(H_j = Y|H_j \neq 0)$ and $l_j = P(H_j \neq 0)$. Note that in the binary case $\theta_j^{(1)*}$ is positive only when $\alpha_j > 0.5$.