We present an extended related work (Appendix A), glossary (Appendix B), additional algorithmic details (Appendix C), proofs (Appendix D), experimental details (Appendix E), and additional experimental results (Appendix F).

A EXTENDED RELATED WORK

Weak supervision is a broad set of techniques using weak sources of signal to supervise models, such as distant supervision [Takamatsu et al., 2012], co-training methods [Blum and Mitchell, 1998], pattern-based supervision [Gupta and Manning, 2014] and feature annotation [Mann and McCallum, 2010; Liang et al., 2009]. Weak supervision frameworks often train in two stages—first modeling source accuracies to generate weak labels, and then fine-tuning a powerful end model for generalization [Ratner et al., 2018; Bach et al., 2019; Khetan et al., 2018; Sheng et al., 2020; Fu et al., 2020; Zhan et al., 2019; Safranchik et al., 2020; Boecking and Dubrawski, 2019]. Our work removes the second stage from the equation and addresses two common challenges in weak supervision, coarse accuracy modeling and low coverage.

One weak supervision work that does not train in two stages, and models source qualities in a way that can be nonuniform over the points is WeaSuL [Cachay et al., 2021]. However, this capability is present in a different context: end-to-end training of a weak supervision label model with an end model. This prevents the use of the label model directly for prediction, as we seek to do in our work. It requires much heavier computational budget, for example, when training a deep model, which is not needed with our approach. In addition, WeaSuL relies on the use of an encoder for source qualities, rendering a theoretical analysis intractable. By contrast, our approach offers clean and easy-to-interpret theoretical guarantees.

Transfer learning uses large datasets to learn useful feature representations that can be fine-tuned for downstream tasks [Kolesnikov et al., 2020; Devlin et al., 2018]. Transfer learning techniques for text applications typically pre-train on large corpora of unlabeled data [Devlin et al., 2018; Brown et al., 2020; Radford et al., 2019], while common applications of transfer learning to computer vision pre-train on both large supervised datasets such as ImageNet [Russakovsky et al., 2015] and large unsupervised or weakly-supervised datasets [He et al., 2019; Chen et al., 2020; Radford et al., 2021]. Pre-trained embeddings have also been used as data point descriptors for kNN search algorithms to improve model performance, interpretability, and robustness [Papernot and McDaniel, 2018; Khandelwal et al., 2019]. We view our work as complementary to these approaches, presenting another mechanism for using pre-trained networks.

Foundation models offer a new interface for the transfer learning setting: when it is impossible to fine-tune the original models [Bommasani et al., 2021]. In this setting, the foundation models can still be used either by direct prompting [Lester et al., 2021; Brown et al., 2020], or by using embeddings [Neelakantan et al., 2022; Wang et al., 2021] prompts FMs to produce pseudolabels, providing a complementary way to use FMs in weak supervision. In contrast, our work focuses on using FM embeddings. Since we can only access the final embeddings of some foundation models, we focus on
adapters (Houlsby et al., 2019) over the final layer in this work—which are equivalent to linear probes (Alain and Bengio, 2016).

**Semi-supervised and few-shot learning** approaches aim to learn good models for downstream tasks given a few labeled examples. Semi-supervised approaches like label propagation (Iscen et al., 2019) start from a few labeled examples and iteratively fine-tune representations on progressively larger datasets, while few-shot learning approaches such as meta-learning and metric learning aim to build networks that can be directly trained with a few labels (Snell et al., 2017). Our work is inspired by these approaches for expanding signal from a subset of the data to the entire dataset using FM representations, but we do not assume that our labeling sources are perfect, and we do not tune the representation.

### B GLOSSARY

The glossary is given in Table 3 below.
Table 3: Glossary of variables and symbols used in this paper.
C ADDITIONAL ALGORITHMIC DETAILS

We describe some properties of the graphical model that justify our algorithm (Section C.1). Then, we formalize the triplet method algorithm for estimating local accuracy parameters, \( \hat{\alpha}_i(C_j) \) (Section C.2).

C.1 PROPERTIES OF THE GRAPHICAL MODEL

**Lemma 2.** For \( x, y, \lambda \) satisfying (1), it holds for any \( \lambda_i \) that

\[
\Pr(y, \lambda_i = 0|x) = \Pr(y|x) \Pr(\lambda_i = 0|x).
\]

That is, \( y \perp \perp \{\lambda_i = 0\} | x \).

**Proof.** Denote \( \lambda_{-i} = \lambda \setminus \lambda_i \), and equivalently let \( \theta_{-i} \) and \( \theta_{-i,0} \) denote vectors of canonical parameters corresponding to \( \lambda_{-i} \) in (1). We show independence by proving that \( \Pr(y = 1, \lambda_i = 0|x) = \Pr(y = 1|x) \Pr(\lambda_i = 0|x) \):

\[
\Pr(y = 1, \lambda_i = 0|x) = \frac{1}{Z} \sum_{\lambda_{-i}} \exp(\theta_y(x) + \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\})
\]

\[
= \frac{1}{Z} \exp(\theta_y(x) + \theta_i(x)) \sum_{\lambda_{-i}} \exp(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}).
\]

\( \Pr(y = 1|x) \) can be written as

\[
\Pr(y = 1) = \frac{1}{Z} \sum_{\lambda_{-i}} \left( \exp(\theta_y(x) + \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}) \right.
\]

\[
+ \exp(\theta_y(x) - \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\})
\]

\[
+ \exp(\theta_y(x) + \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\})
\]

\[
= \frac{1}{Z} \sum_{\lambda_{-i}} \exp(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}) \exp(\theta_y(x))(\exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x))).
\]

\( \Pr(\lambda_i = 0|x) \) can be written as

\[
\Pr(\lambda_i = 0|x) = \frac{1}{Z} \sum_{\lambda_{-i}} \left( \exp(\theta_y(x) + \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}) \right.
\]

\[
+ \exp(-\theta_y(x) + \theta_i(x) - \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\})
\]

\[
= \frac{1}{Z} \sum_{\lambda_{-i}} \exp(\theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}) \exp(\theta_{i,0}(x))(\exp(\theta_y(x) + \theta_{-i}(x)\lambda_{-i}(x)) + \exp(-\theta_y(x) - \theta_{-i}(x)\lambda_{-i}(x))).
\]

Setting (5) equal to (5) times (6), the term \( \sum_{\lambda_{-i}} \exp(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{I}\{\lambda_{-i} = 0\}) \) in the former two equations cancels out. We thus aim to prove the following equality:

\[
Z \exp(\theta_y(x) + \theta_{i,0}(x)) = \exp(\theta_y(x))(\exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x))) \times
\]

\[
\sum_{\lambda_{-i}} \exp(\theta_{-i,0}\mathbb{I}\{\lambda_{-i} = 0\}) \exp(\theta_{i,0}(x))(\exp(\theta_y(x) + \theta_{-i}(x)\lambda_{-i}(x)) + \exp(-\theta_y(x) - \theta_{-i}(x)\lambda_{-i}(x))).
\]

Canceling out \( \exp(\theta_y(x) + \theta_{i,0}(x)) \), (7) is equal to

\[
Z = \left( \exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x)) \right) \times
\]

\[
\sum_{\lambda_{-i}} \exp(\theta_{-i,0}\mathbb{I}\{\lambda_{-i} = 0\})(\exp(\theta_y(x) + \theta_{-i}(x)\lambda_{-i}(x)) + \exp(-\theta_y(x) - \theta_{-i}(x)\lambda_{-i}(x))).
\]
which is true since the RHS iterates over all values of \( \lambda_i, y, \) and \( \lambda_1. \) We have shown that \( \Pr(y = 1, \lambda_i = 0|x) = \Pr(y = 1|x) \Pr(\lambda_i = 0|x) \) and thus that \( y \perp \perp \{\lambda_i = 0\} | x \) for any \( \lambda_i. \)

Due to this independence property, we note that

\[
\Pr(\lambda_i = 0, \lambda_{-i}|x) = \Pr(\lambda_i = 0, \lambda_{-i}|y = 1, x) \Pr(y = 1|x) + \Pr(\lambda_i = 0, \lambda_{-i}|y = -1, x) \Pr(y = -1|x)
\]

\[
= \Pr(\lambda_i = 0|x) \left( \Pr(\lambda_{-i}|y = 1, x) \Pr(y = 1|x) + \Pr(\lambda_{-i}|y = -1, x) \Pr(y = -1|x) \right)
\]

\[
= \Pr(\lambda_i = 0|x) \Pr(\lambda_{-i}|x),
\]

and hence

\[
\Pr(y|\lambda_i = 0, \lambda_{-i}, x) = \frac{\Pr(\lambda_i = 0|y, x) \Pr(\lambda_{-i}|y, x) \Pr(y|x)}{\Pr(\lambda_i = 0, \lambda_{-i}|x)} = \frac{\Pr(\lambda_i = 0|y) \Pr(\lambda_{-i}|y, x) \Pr(y|x)}{\Pr(\lambda_i = 0|x) \Pr(\lambda_{-i}|x)}
\]

\[
= \Pr(y|\lambda_{-i}, x).
\]

\[\square\]

**Lemma 3.** For any \( i \neq j, \) if \( x, \lambda, y \) follows (1), then \( \lambda_i y \perp \perp \lambda_j y| \lambda_i \land \lambda_j \neq 0, x. \)

**Proof.** Conditioning on the event that \( \lambda \neq 0, \) we have that

\[
\Pr(y, \lambda|\lambda \neq 0, x) = \frac{1}{Z_0} \exp \left( \theta_y(x) y + \sum_{i=1}^{m} \theta_i(x) \lambda_i y \right)
\]

for some partition function \( Z_0 \) different from \( Z \) in (1). This graphical model now follows the structure of the graphical model in [Fu et al. 2020](ref) (see their Equation 3). We can thus apply Proposition 1 of their work to get that \( \lambda_i y \perp \perp \lambda_j y| \lambda \neq 0, x. \) From Lemma 2 and conditional independence of sources, this independence property is equivalent to \( \lambda_i y \perp \perp \lambda_j y| \lambda_i \land \lambda_j \neq 0, x, \) as desired. \[\square\]

**Lemma 4.** If \( x, \lambda, y \) follows (1), then for any \( \lambda_i, \)

\[
\Pr(\lambda_i = 1|y = 1, \lambda_i \neq 0, x) = \Pr(\lambda_i = -1|y = -1, \lambda_i \neq 0, x) = \Pr(\lambda_i y = 1|\lambda_i \neq 0, x)
\]

\[
\Pr(\lambda_i = -1|y = -1, \lambda_i \neq 0, x) = \Pr(\lambda_i y = -1|\lambda_i \neq 0, x).
\]

Therefore,

\[
\Pr(\lambda_i|y, \lambda_i \neq 0, x) = \frac{1 + \text{sgn}(\lambda_i y) a_i(x)}{2}.
\]

**Proof.** Conditioning on the event that \( \lambda \neq 0, \) the graphical model is of the form in (8) above. This graphical model also follows the structure of that in [Chen et al. 2021](ref), and therefore we obtain our desired properties by Lemma 2 of their work. \[\square\]

### C.2 LOCAL ACCURACY PARAMETER ESTIMATION ALGORITHM

We formalize the triplet method used to recover latent source parameters \( \Pr(\lambda_i|y, x). \) First, when we want to evaluate \( \Pr(\lambda_i = 1|y, x) \) or \( \Pr(\lambda_i = -1|y, x), \) this probability can be written as \( \Pr(\lambda_i|y, x, \lambda_i \neq 0) \Pr(\lambda_i \neq 0|y, x) = \Pr(\lambda_i|y, x, \lambda_i \neq 0) \Pr(\lambda_i \neq 0|x) \) by Lemmas 2 and 4. We have that \( \mathbb{E}[\lambda_i y|x, \lambda_i \neq 0] = \Pr(\lambda_i y = 1|x, \lambda_i \neq 0) - \Pr(\lambda_i y = -1|x, \lambda_i \neq 0) = 2 \Pr(\lambda_i y = 1|x, \lambda_i \neq 0) - 1, \) so \( \Pr(\lambda_i|y, x) = \frac{1 + \text{sgn}(\lambda_i y) a_i(x)}{2} \cdot \Pr(\lambda_i \neq 0|x) \) when \( \lambda_i \in \{-1, 1\}. \) When \( \lambda_i \) is 0, the probability we want to estimate is \( \Pr(\lambda_i = 0|y, x) = \Pr(\lambda_i = 0|x) \) by Lemma 2.
We present the proofs for our results in Section 4.

We now explain how our algorithm estimates $a_i(x)$. From Lemma 3, we have that $\lambda_i \perp \lambda_j \mid \lambda_i \land \lambda_j \neq 0, x$ for any $i, j$. Then, given any set of $\lambda_i, \lambda_j, \lambda_k$, we have the set of equations

$$
\begin{align*}
\text{local accuracy} & = \mathbb{E}[\lambda_1 \lambda_2] \mid x = 0, \lambda = 3 \neq 0 \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x.
\end{align*}
$$

Solving, we get that

$$
|a_i(x)| = \sqrt{\frac{\mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x \mathbb{E}[\lambda_1 \lambda_2] \mid \lambda = 3 \neq 0, x}.
$$

This property allows us to recover $a_i(x)$ up to a sign. As discussed in Section 3, we use $C(x)$ to estimate the accuracy parameter over a region of the embedding space, such that in fact we are estimating $a_i(C(x)) = \mathbb{E}[\lambda_i \mid \lambda_i \neq 0, C(x)]$ (since Lemma 3 holds on any $x$, it holds conditioned over $C(x)$ too). We resolve the sign of the accuracy parameter by assuming that $a_i(C(x)) > 0$, meaning that the accuracy of a source over a subset is better than random. Finally, rather than estimating $a_i(C(x))$ using just one pair of $\lambda_j$ and $\lambda_k$, we compute the average $a_i(C(x))$ over all other pairs $(\lambda_j, \lambda_k \in \lambda \backslash \lambda_i)$ to make the estimate less noisy. Our approach for computing $a_i(C_j)$ for any $\lambda_i$ and $C_j$ (note that $\lambda_i$ and $\lambda_i$ are interchangeable in the above given that $(x, y, \lambda)$ and $(x, y, \lambda)$ both satisfy (1)) is described in Algorithm 2.

D PROOFS

We present the proofs for our results in Section 4.

D.1 PROOFS FOR SECTION 4.1

The proof of Theorem 1 involves decomposing the generalization error into the irreducible error, bias from using $C(x)$, and variance (sampling error).

**Theorem 1.** Suppose that data $x, y, \lambda$ follows the model in (1) and $\Pr(y \mid x)$ and $\Pr(\lambda_i \mid y, x)$ for each $\lambda_i$ are Lipschitz-smooth. The generalization error of the label model $\Pr(y \mid \lambda, x)$ in Algorithm 2 when $r_i = 0 \forall i$ can be decomposed into $L(\lambda) = \text{Bias} + \text{Variance} + \text{Irreducible Error} + o(1/n)$, where

$$
\begin{align*}
\text{Bias} & \leq 2dc(K_y + mK + mK_{\lambda,0}), \\
\text{Variance} & \leq \frac{ms}{n} \left( 3\alpha(1 - b_{\min}^2) \left( \frac{1}{b_{\min}^2} + \frac{2}{b_{\min}^2} \right) + 1 \right), \\
\text{Irreducible Error} & = H(y \mid \lambda, x),
\end{align*}
$$

where $H(y \mid \lambda, x)$ denotes conditional entropy.
Proof. We can write the generalization error as

\[ L(\lambda) = \mathbb{E}_{D,x,y,\lambda} \left[ - \log \hat{P}(y|\lambda, x) \right] = \mathbb{E} \left[ - \log \frac{\hat{P}(y|\lambda, x)}{P(y|\lambda, x)} \right] - \mathbb{E}_{x,y,\lambda} \left[ \log P(y|\lambda, x) \right]. \]

\(-\mathbb{E}_{x,y,\lambda} [P(y|\lambda, x)]\) is equal to the conditional entropy of \(y\) given \(\lambda, x\), expressed as \(H(y|\lambda, x)\). This describes the entropy of \(y\) after observing the weak labels and input and thus depends on how much signal we are getting from the labelers. Next, we decompose the expected log ratio using our construction of \(P(\lambda_i|y, C(x))\) to get

\[ L(\lambda) = \mathbb{E} \left[ - \log \left( \prod_{i=1}^{m} \frac{\hat{P}(\lambda_i|y, C(x)) P(y|C(x))}{\hat{P}(\lambda_i|y, x) P(y|x)} \right) \right] + H(y|\lambda, x) \]

\[ = -\mathbb{E} \left[ \sum_{i=1}^{m} \log \frac{\hat{P}(\lambda_i|y, C(x))}{\hat{P}(\lambda_i|y, x)} \right] - \mathbb{E} \left[ \log \frac{P(\lambda|y, C(x))}{P(\lambda|y, x)} \right] - \mathbb{E} \left[ \log \frac{P(y|C(x))}{P(y|x)} \right] + H(y|\lambda, x) \]

\[ \leq \sum_{i=1}^{m} \mathbb{E}_{x,y,\lambda_i} \left[ \log \frac{P(\lambda_i|y, x)}{P(\lambda_i|y, C(x))} \right] - \mathbb{E}_{x,y} \left[ \log \frac{P(y|C(x))}{P(y|x)} \right] + H(y|\lambda, x), \quad (11) \]

where we have used Lemma [2] and the fact that the Kullback-Leibler divergence is always nonnegative in the last line.

For notation, let \(\text{KL}_{C(x)}(y) = \mathbb{E}_x [D_{\text{KL}}(P(y|x)||P(y|C(x)))] = \mathbb{E}_{x,y} \left[ \log \frac{P(y|x)}{P(y|C(x))} \right]\), be the KL-divergence between distributions conditioned on \(C(x)\) versus \(x\), which describes the bias we incur from using a partition. Then, \(L(\lambda) \leq \sum_{i=1}^{m} \mathbb{E}_{x,y,\lambda_i} \left[ \log \frac{P(\lambda_i|y, x)}{P(\lambda_i|y, C(x))} \right] + \text{KL}_{C(x)}(y) + H(y|\lambda, x)\).

We now simplify the expression \(\mathbb{E} \left[ \log \frac{P(\lambda_i|y, x)}{P(\lambda_i|y, C(x))} \right]\) based on if \(\lambda_i = 0\) or \(\lambda_i \in \{-1, 1\}\):

\[ \mathbb{E} \left[ \log \frac{P(\lambda_i|y, x)}{P(\lambda_i|y, C(x))} \right] = \mathbb{E}_x \left[ \log P(\lambda_i = 0|x) \log \frac{P(\lambda_i = 0|x)}{P(\lambda_i = 0|C(x))} \right] + \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \frac{P(\lambda_i \neq 0|x)}{P(\lambda_i \neq 0|C(x))} \right] \]

\[ = \mathbb{E}_x \left[ \log \frac{P(\lambda_i = 0|x)}{P(\lambda_i = 0|C(x))} + P(\lambda_i \neq 0|x) \log \frac{P(\lambda_i \neq 0|x)}{P(\lambda_i \neq 0|C(x))} \right] + \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \frac{P(\lambda_i \neq 0|x)}{P(\lambda_i \neq 0|C(x))} \right] \]

\[ = \mathbb{E}_x \left[ D_{\text{KL}}(P(z_i|x)||P(z_i|C(x))) \right] + \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \frac{P(\lambda_i \neq 0|x)}{P(\lambda_i \neq 0|C(x))} \right], \quad (12) \]

where \(z_i \in \{\lambda_i = 0\}\) is an indicator variable pertaining to coverage. The first KL divergence pertains to estimating the coverage of \(\lambda_i\), while the second pertains to estimating the accuracy parameter of \(\lambda_i\). The first term in (12) can be written as

\[ \mathbb{E}_x \left[ D_{\text{KL}}(P(z_i|x)||P(z_i|C(x))) \right] = \text{KL}_{C(x)}(z_i) \]

\[ + \mathbb{E}_x \left[ \log \frac{P(\lambda_i = 0|C(x))}{P(\lambda_i = 0|C(x))} + P(\lambda_i \neq 0|C(x)) \log \frac{P(\lambda_i \neq 0|C(x))}{P(\lambda_i \neq 0|C(x))} \right] \]

\[ = \text{KL}_{C(x)}(z_i) + \mathbb{E}_{C(x)} \left[ \log \frac{P(\lambda_i = 0|C(x))}{P(\lambda_i = 0|C(x))} + P(\lambda_i \neq 0|C(x)) \log \frac{P(\lambda_i \neq 0|C(x))}{P(\lambda_i \neq 0|C(x))} \right] \]

\[ = \text{KL}_{C(x)}(z_i) + \mathbb{E}_{C(x),z_i} \left[ \log \frac{P(z_i|C(x))}{P(z_i|C(x))} \right], \]
The second term in \([12]\) can be written as
\[
\mathbb{E} \left[ \Pr(\lambda_i \neq 0|x) \log \frac{\Pr(\lambda_i|y, x, \lambda_i \neq 0)}{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)} \right] \leq \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \left( \frac{\Pr(\lambda_i|y, x, \lambda_i \neq 0)}{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)} \right) - \log \left( \frac{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)}{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)} \right) \right] = KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) + \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \frac{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)}{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)} \right].
\]

Putting everything together in \([11]\), the generalization error is at most
\[
L(\lambda) \leq \sum_{i=1}^{m} \left( \mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] + \mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log \frac{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)}{\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)} \right] + KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) \right) + KL_{C(x)}(y) + H(y|\lambda, x).
\]

We can interpret the generalization error as consisting of bias, variance (and irreducible error) coming from 1) estimating the coverage of a weak source over a part, and then, conditioned on the support of a source, 2) estimating the accuracy of the source over a part. The bias is from using \(C(x)\) instead of \(x\), and the variance is from estimating over the dataset over these two steps.

Using Lemmas 5, 6, and 7 we get our desired bound.

**Lemma 5.** The sampling error term coming from estimating \(\lambda_i\)’s coverage, \(\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right]\), where \(z_i = \mathbb{1} \{\lambda_i = 0\}\), is equal to
\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \frac{s}{n} + o(1/n).
\]

**Proof.** We can write this expectation across each \(C_j\). Denote \(p_{ij} = \Pr(\lambda_i \neq 0|C_j)\) as \(\lambda_i\)’s coverage on \(C_j\), and equivalently \(\hat{p}_{ij}\) as its estimate over \(D\).

Then,
\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \mathbb{E}_D \left[ \frac{p_{ij} \log \frac{p_{ij}}{\hat{p}_{ij}} + (1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - \hat{p}_{ij}}}{1} \right].
\]

Performing a Taylor approximation of \(g(x) = \log \frac{x}{2}\) at \(x = c\) gives us \(\log \frac{x}{2} \approx \log 1 + \frac{1}{x}(x - c) + \frac{1}{2x^2}(x - c)^2\). Setting \(x = \hat{p}_{ij}, 1 - \hat{p}_{ij}\) and \(c = \hat{p}_{ij}, 1 - \hat{p}_{ij}\) respectively in \([13]\) and using the fact that \(\hat{p}_{ij}\) is an unbiased estimate of \(p_{ij}\), this expression becomes
\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{p_{ij} (1 - p_{ij})} \mathbb{E} \left[ (p_{ij} - \hat{p}_{ij})^2 \right] + o(1/n) = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{p_{ij} (1 - p_{ij})} \text{Var} \left[ \hat{p}_{ij} \right] + o(1/n),
\]

where we use the fact that the Taylor remainder scales in \(\mathbb{E} \left[ (\hat{p}_{ij} - p_{ij})^2 \right] \sim O(1/n^2)\). We can simplify the variance \(\text{Var} \left[ \hat{p}_{ij} \right] = \frac{1}{n} \sum_{x:f(x) \in C_j} \mathbb{1} \{\lambda_i(x) \neq 0\} = \frac{1}{(n^2 - s)} \sum_{x:f(x) \in C_j} \text{Var} \left[ \mathbb{1} \{\lambda_i(x) \neq 0\} \right] = \frac{p_{ij}(1 - p_{ij})}{n^2} \). Putting this all together, we have
\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{n^2} + o(1/n) = \frac{s}{n} + o(1/n).
\]

\(\square\)
Lemma 6. Define $p_{ij} = \Pr(\lambda_i \neq 0 | C_j)$ as the coverage of the $\lambda_i$ on $C_j$. The sampling error term coming from estimating source accuracy of $\lambda_i$, $\Pr(\lambda_i | y, C(x), \lambda_i \neq 0)$, is at most

$$
E_{x,y,\lambda_i \neq 0} \left[ \log \frac{\Pr(\lambda_i | y, C(x), \lambda_i \neq 0)}{\Pr(\lambda_i | y, C(x), \lambda_i \neq 0)} \right] \leq E_{C_j} \left[ \frac{1}{p_{ij}} \mid p_{ij} \neq 0 \right] \cdot \frac{3s}{8n} \cdot \frac{1 - b_{min}^2}{b_{min}^2 (1 - a_{max}^2)} \left( \frac{1}{b_{min}^2} + \frac{2}{b_{min}^2} \right) + o(1/n).
$$

Proof. Define $C_i \subseteq C$ to be the subsets where $\lambda_i$ has non-zero coverage, $\{C \in C : \exists x : f(x) \in C, \lambda_i(x) \neq 0\}$. When there are subsets with no $\lambda_i$ coverage, we do not estimate the accuracy and can discard them from this bound. We can thus write the above expectation as $E \left[ \log \frac{1 + \text{sgn}(\lambda_i y, a_i(C(x))}{1 + \text{sgn}(\lambda_i y, \hat{a}_i(C(x)))} \right] = \sum_{C_j \in C} E(f(x) \in C_j) E \left[ \log \frac{1 + \text{sgn}(\lambda_i y, a_i(C_j))}{1 + \text{sgn}(\lambda_i y, \hat{a}_i(C_j)))} \right]$. We can decompose the expectation as

$$
E_{x,y,\lambda_i \neq 0} \left[ \log \frac{1 + \text{sgn}(\lambda_i y, a_i(C_j))}{1 + \text{sgn}(\lambda_i y, \hat{a}_i(C_j))} \right] = E \left[ \log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)} \right] \Pr(\lambda_i y = 1 | C_j, \lambda_i \neq 0) + E \left[ \log \frac{1 - a_i(C_j)}{1 - \hat{a}_i(C_j)} \right] \Pr(\lambda_i y = -1 | C_j, \lambda_i \neq 0).
$$

(14)

(15)

Pr($\lambda_i y = 1 | C_j, \lambda_i \neq 0$) is equal to $\frac{1 + a_i(C_j)}{2}$. (15) becomes

$$
E \left[ \log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)} \right] = \frac{1}{2} \left( 1 + a_i(C_j) \right) E \left[ \log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)} \right] + \frac{1}{2} \left( 1 - a_i(C_j) \right) E \left[ \log \frac{1 - a_i(C_j)}{1 - \hat{a}_i(C_j)} \right].
$$

(16)

Again, we can perform a Taylor expansion on $g(x) = \log \frac{1 + x}{1 + x}$ at $x = c$ to get that $\log \frac{1 + \epsilon}{1 + \epsilon} \approx -\frac{1 + \epsilon}{1 + \epsilon} (x-c) + o\left(\frac{1 + \epsilon}{1 + \epsilon} (x-c)^2\right)$, and therefore $E \left[ \log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)} \right] = E\left[\frac{a_i(C_j) - \hat{a}_i(C_j)}{1 + a_i(C_j)}\right] + E\left[\frac{(\hat{a}_i(C_j) - a_i(C_j))^2}{2(1 + a_i(C_j))^2}\right] + o(1/n)$, (see Lemma 4 of Chen et al. [2021]).

(16) for bounding the Taylor remainder. Similarly, we have that $E \left[ \log \frac{1 - a_i(C_j)}{1 - \hat{a}_i(C_j)} \right] = E\left[\frac{\hat{a}_i(C_j) - a_i(C_j)}{1 - a_i(C_j)}\right] + E\left[\frac{(\hat{a}_i(C_j) - a_i(C_j))^2}{2(1 - a_i(C_j))^2}\right] + o(1/n)$. Therefore, (16) becomes

$$
E \left[ \log \frac{1 + \text{sgn}(\lambda_i y, a_i(C_j))}{1 + \text{sgn}(\lambda_i y, \hat{a}_i(C_j))} \right] = \frac{1}{2} \left( E \left[ a_i(C_j) - \hat{a}_i(C_j) \right] + \frac{E\left[(\hat{a}_i(C_j) - a_i(C_j))^2\right]}{2(1 + a_i(C_j))} \right) + E \left[ \hat{a}_i(C_j) - a_i(C_j) \right] + \frac{E\left[(\hat{a}_i(C_j) - a_i(C_j))^2\right]}{2(1 - a_i(C_j))} + o(1/n).
$$

The value of $E\left[(\hat{a}_i(C_j) - a_i(C_j))^2\right]$ has been studied in previous works that use the triplet method of Fu et al. [2020]. In particular, we use Lemma 6 of Chen et al. [2021] to get that

$$
E\left[(\hat{a}_i(C_j) - a_i(C_j))^2\right] \leq \frac{3s}{4p_{i,j,n}} \cdot \frac{1 - b_{min}^2}{b_{min}^2} \left( \frac{1}{b_{min}^2} + \frac{2}{b_{min}^2} \right).
$$

Therefore, the overall expression can be bounded by

$$
E \left[ \log \frac{1 + \text{sgn}(\lambda_i y, a_i(C(x))}{1 + \text{sgn}(\lambda_i y, \hat{a}_i(C(x))} \right] \leq \sum_{C_j \in C} E(f(x) \in C_j) \cdot \frac{1}{2(1 - a_{max}^2)} \cdot \frac{3s}{4p_{i,j,n}} \cdot \frac{1 - b_{min}^2}{b_{min}^2} \left( \frac{1}{b_{min}^2} + \frac{2}{b_{min}^2} \right) + o(1/n)
$$

$$
\leq E_{C_j} \left[ \frac{1}{p_{ij}} \mid p_{ij} \neq 0 \right] \frac{3s}{8n} \cdot \frac{1 - b_{min}^2}{b_{min}^2 (1 - a_{max}^2)} \left( \frac{1}{b_{min}^2} + \frac{2}{b_{min}^2} \right) + o(1/n).
$$

□
Lemma 7. Denote \( z_i = \mathbb{1} \{ \lambda_i = 0 \} \) and \( KL_{C(x)}(\cdot) = \mathbb{E}_x [D_{KL}(\Pr(\cdot|x), \Pr(\cdot|C(x)))] \). The bias terms from conditioning on \( C(x) \) rather than \( x \) are at most

\[
KL_{C(x)}(y) \leq 2K_y d_C
\]

\[
KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) \leq 2mK_\lambda d_C
\]

\[
KL_{C(x)}(z_i) \leq 2mK_\lambda,0 d_C.
\]

Proof. We can write the expected KL-divergence between the distribution of the true label \( y \) conditioned on \( C(x) \) versus \( x \) as

\[
KL_{C(x)}(y) = \mathbb{E}_x [D_{KL}(\Pr(y|x)\|\Pr(y|C(x)))] = \sum_{j=1}^s \Pr(f(x) \in C_j) \int \Pr(x|C_j) D_{KL}(\Pr(y|x, C_j)\|\Pr(y|C_j)) dx.
\]

This inner KL-divergence is on two Bernoulli distributions. Define \( p_{y,j} = \Pr(y = 1|C_j) \), and denote \( p_{y,x,j} = \Pr(y = 1|x, C_j) \). Then, \( D_{KL}(\Pr(y|x, C_j)\|\Pr(y|C_j)) = p_{y,x,j} \log \frac{p_{y,x,j}}{p_{y,j}} + (1 - p_{y,x,j}) \log \frac{1 - p_{y,x,j}}{1 - p_{y,j}} \).

Next, recall that \( \Pr(y|x) \) is \( K_y \)-Lipschitz in the embedding space; that is, \( |\Pr(y = 1|x) - \Pr(y = 1|x')| \leq K_y |f(x) - f(x')| \). Since \( p_{y,j} \) is \( \Pr(y|x) \) averaged over \( C_j \), it holds that \( |p_{y,x,j} - p_{y,j}| \leq K_y d_j \), where \( d_j \) is the diameter of \( C_j \). We then have that \( p_{y,x,j} \leq K_y d_j + p_{y,j} \), and since \( |1 - p_{y,x,j} - (1 - p_{y,j})| \leq K_y d_j \), we also have that \( 1 - p_{y,x,j} \leq 1 - p_{y,j} + K_y d_j \). Therefore, the KL-divergence is bounded by

\[
D_{KL}(\Pr(y|x, C_j)\|\Pr(y|C_j)) \leq p_{y,x,j} \log \frac{K_y d_j + p_{y,j}}{p_{y,j}} + (1 - p_{y,x,j}) \log \frac{K_y d_j + (1 - p_{y,j})}{1 - p_{y,j}}
\]

\[
\leq p_{y,x,j} \cdot \frac{K_y d_j}{p_{y,j}} + (1 - p_{y,x,j}) \cdot \frac{K_y d_j}{1 - p_{y,j}},
\]

where we use the fact that \( \log(1 + x) \leq x \). Plugging this back into (17),

\[
\mathbb{E}_x [D_{KL}(\Pr(y|x)\|\Pr(y|C(x)))] \leq \sum_{j=1}^s \Pr(f(x) \in C_j) \int \Pr(x|C_j) \left( p_{y,x,j} \cdot \frac{K_y d_j}{p_{y,j}} + (1 - p_{y,x,j}) \cdot \frac{K_y d_j}{1 - p_{y,j}} \right) dx
\]

\[
= \sum_{j=1}^s \Pr(f(x) \in C_j) \int \Pr(x, y = 1|C_j) \cdot \frac{K_y d_j}{p_{y,j}} + \Pr(x, y = -1|C_j) \cdot \frac{K_y d_j}{1 - p_{y,j}} dx
\]

\[
= \sum_{j=1}^s \Pr(f(x) \in C_j) \left( \Pr(y = 1|C_j) \cdot \frac{K_y d_j}{p_{y,j}} + \Pr(y = -1|C_j) \cdot \frac{K_y d_j}{1 - p_{y,j}} \right)
\]

\[
= \sum_{j=1}^s \Pr(f(x) \in C_j) \cdot 2K_y d_j = 2K_y d_C.
\]

Next, we bound \( KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) \). Using the same approach, we have that \( KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) \leq 2K_\lambda d_C \). We also have that \( KL_{C(x)}(z_i) \leq 2K_\lambda,0 d_C \).

D.2 PROOFS FOR SECTION 4.2

Lemma 8. When we use \( \bar{\lambda} \) instead of \( \lambda \), the bias term in \( L(\bar{\lambda}) \) is at most

\[
Bias \leq 2d_C K_y + 2m(d_C + 2 \max_i r_i)(K_\lambda + K_\lambda,0).
\]
Proof. The term $\mathbb{E}_{x, y, \tilde{\lambda}_i \neq 0} [D_{KL}(Pr(\tilde{\lambda}_i | y, \tilde{\lambda}_i) \neq \tilde{\lambda}_i) || Pr(\tilde{\lambda}_i | y, C(x), \tilde{\lambda}_i \neq 0)]$ in the bias is unchanged since the distribution of $y$ given $x$ is not impacted by $\lambda$. We next look at $\mathbb{E}_{x, y, \tilde{\lambda}_i \neq 0} [D_{KL}(Pr(\tilde{\lambda}_i | y, \tilde{\lambda}_i, \tilde{\lambda}_i \neq 0) || Pr(\tilde{\lambda}_i | y, C(x), \tilde{\lambda}_i \neq 0))].$ Using the approach in Lemma [7] recall that $Pr(\tilde{\lambda}_i | y, \tilde{\lambda}_i) = \Pr(\lambda_i | y, x, \lambda_i(x) \neq 0)$ when $\lambda_i(x) \neq 0$, and $\Pr(\lambda_i(\text{NN}(x)) | y, \text{NN}(x), \lambda_i(\text{NN}(x)) \neq 0)$ when $\lambda_i(x) = 0$. Therefore, by Assumption [1] we have $\Pr(\lambda_i = 1 | y, \lambda_i \neq 0, x) - \Pr(\lambda_i = 1 | y, \lambda_i \neq 0, x') \leq K \max \{\rho(f(\text{NN}(x)), f(\text{NN}(x'))), \rho(f(x), f(\text{NN}(x))), \rho(f(\text{NN}(x)), f(x')), \rho(f(x), f(x'))\}$, which is the maximum possible distance in embedding space between $\text{NN}(x)$ and $\text{NN}(x')$ when $f(x), f(x') \in C_j$ under our method of source extension is $d_j + 2r_i$. We can thus view the extensions as changing the diameter of the subset in Lemma [7]. The rest of the approach remains unchanged, so we get that

$$\mathbb{E}_{x, y, \tilde{\lambda}_i \neq 0} [D_{KL}(Pr(\tilde{\lambda}_i | y, \tilde{\lambda}_i) \neq \tilde{\lambda}_i) || Pr(\tilde{\lambda}_i | y, C(x), \tilde{\lambda}_i \neq 0)] \leq 2K_\lambda(d_c + 2r_i).$$

We consider $\mathbb{E}_{x, y, \tilde{\lambda}_i \neq 0} [D_{KL}(Pr(\tilde{\lambda}_i | y, \tilde{\lambda}_i) \neq \tilde{\lambda}_i) || Pr(\tilde{\lambda}_i | y, \lambda(x), \tilde{\lambda}_i \neq 0)]$. Similarly, $\Pr(\tilde{\lambda}_i \neq 0 | x)$ is either $\Pr(\lambda_i(x) \neq 0 | x)$ or $\Pr(\lambda_i(\text{NN}(x)) \neq 0 | \text{NN}(x))$ depending on the region $x$ is in. Therefore,

$$\mathbb{E}_{x} [D_{KL}(Pr(\lambda_i \neq 0 | x) || Pr(\lambda_i \neq 0 | C(x)))] \leq 2K_\lambda,0(d_c + 2r_i),$$

and we obtain the desired bound. □

Lemma 1. Suppose $\mathcal{P}_{\lambda_i}$ is $M$-probabilistically Lipschitz. The average accuracy of $\hat{\lambda}_i$ on the extended region is at least $\bar{a}_i = \frac{1}{1 + \beta_i}$.

Proof. We first introduce some notation. Define $S = \{x \in \mathcal{X} : \lambda_i(x) \neq 0\}$ as the support of $\lambda_i$, and $\hat{S} = S \cap \mathcal{D}$ as the set of points in $\mathcal{D}$ that $\lambda_i$ has coverage on. In particular, $\hat{S}$ consists of points sampled from $\mathcal{P}_{\lambda_i}$, and suppose that $|S| = n_0$. Define the extended region as $S_{r_i} = \{x \in \mathcal{X} : \exists x' \in S \text{ s.t. } \rho(f(x), f(x')) \leq r_i\}$, and let the distribution of $x$ over this support be $\mathcal{P}_{\hat{S}, x} = Pr(x | x \in \hat{S}_{r_i})$. With slight abuse of notation, we also use $\mathcal{P}_{\hat{S}, r_i}$ to refer to the joint distribution over $x, y$ with $x$ from $\mathcal{P}_{\hat{S}, r_i}$. We also use $\hat{S}_{r_i}$ to refer to the support $\hat{S}_{r_i} \times \mathcal{Y}$.

Define the expected error $\varepsilon = \mathbb{E}_{S \sim \mathcal{P}_{\lambda_i}^0, x, y \sim \mathcal{P}_{\hat{S}, x_i}} \left[Pr_{x, y \sim \mathcal{P}_{\hat{S}, x_i}}(\hat{\lambda}_i \neq y | x, \hat{\lambda}_i(x) \neq 0)\right] = \mathbb{E}_{\hat{S} \sim \mathcal{P}_{\lambda_i}^0} \left[Pr_{x, y \sim \mathcal{P}_{\hat{S}, r_i}}(\hat{\lambda}_i \neq y | x)\right]$. Let $\hat{S}$ also be written as a set of $n_0$ random variables $\{x_1, \ldots, x_{n_0}\}$. Denote $\text{NN}_\hat{S}(x) = \arg\min_{x' \in \hat{S}} \rho(f(x), f(x'))$ to be $x'$'s nearest neighbor in $\hat{S}$ (in the body, this is just referred to as $\text{NN}(x)$), so $\hat{\lambda}_i(x) := \lambda_i(\text{NN}_\hat{S}(x))$ for $x \in \hat{S}_{r_i}$. Then, we decompose $\varepsilon$ based on which point in $\hat{S}$ is $x$'s nearest neighbor:

$$\varepsilon = \sum_{j=1}^{n_0} \mathbb{E}_{x \sim \mathcal{P}_{\lambda_i}^0, x, y \sim \mathcal{P}_{\hat{S}, r_i}} \left[Pr_{x, y \sim \mathcal{P}_{\hat{S}, x_i}}(\hat{\lambda}_i(\text{NN}_\hat{S}(x)) \neq y | x) \cdot Pr_{x \sim \mathcal{P}_{\lambda_i}^0}(\lambda_i(x_j) \neq y | \text{NN}_\hat{S}(x) = x_j)\right].$$

Let $y_j$ denote the label corresponding to $x_j$, drawn from $\mathcal{P}_{\lambda_i}(\{x_j\})$. The probability $Pr_{\hat{S} \sim \mathcal{P}_{\lambda_i}^0, x, y \sim \mathcal{P}_{\hat{S}, r_i}}(\lambda_i(x) \neq y | \text{NN}_\hat{S}(x) = x_j)$ can be further decomposed into two cases: when $\lambda(x_j) = y_j$, $y_j \neq y$ and when $\lambda(x_j) \neq y_j$, $y_j = y$. That is,

$$Pr_{\hat{S} \sim \mathcal{P}_{\lambda_i}^0, x, y \sim \mathcal{P}_{\hat{S}, r_i}}(\lambda_i(x) \neq y | \text{NN}_\hat{S}(x) = x_j) = \sum_{j=1}^{n_0} \mathbb{E}_{x \sim \mathcal{P}_{\lambda_i}^0, x, y \sim \mathcal{P}_{\hat{S}, r_i}} \left[Pr_{x, y \sim \mathcal{P}_{\hat{S}, r_i}}(\hat{\lambda}_i(x) \neq y) \cdot Pr_{x \sim \mathcal{P}_{\lambda_i}^0}(\lambda_i(x_j) \neq y | \text{NN}_\hat{S}(x) = x_j)\right].$$

Next, we recall the definition of $\hat{S}_{r_i}$ and observe that $\text{NN}_\hat{S}(x) = x_j$ implies that $\rho(f(x), f(x')) \leq r_i$. These allow us to
write the probability only over one \((x_j, y_j) \sim \mathcal{P}_{\lambda_i}\), rather than \(\hat{S}\), and so the expression in (19) satisfies

\[
\Pr_{\hat{S} \sim \mathcal{P}_{\lambda_i}} \Pr_{x,y \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) \neq y | NN_{\hat{S}}(x) = x_j) \leq \Pr_{x,y \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) = y_j, \exists (x,y) \in \mathcal{X}\mathcal{S} : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) + \Pr_{x,y \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) \neq y, \exists (x,y) \in \mathcal{X}\mathcal{S} : \rho(f(x_j), f(x)) \leq r_i, y_j = y).
\]

The first probability on the RHS can be written as \(\Pr_{x,y \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) = y_j, \exists (x,y) \in \mathcal{X}\mathcal{S} : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) \leq 1 + \frac{\beta_i}{2} M(r_i)\), and the second one is at most \(\Pr_{x,y \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) \neq y_j) = 1 - a_i\). Therefore, putting this back into (18), \(\varepsilon \leq 1 + \frac{\beta_i}{2} M(r_i) + 1 - a_i\). Since \(\bar{a}_i(r_i) = 2(1 - \varepsilon) - 1\), we now have our desired bound

\[
\bar{a}_i(r_i) \geq a_i - (1 + \beta_i) M(r_i).
\]

\(\square\)

**Theorem 2.** Suppose that data follows the model in Algorithm 1. The irreducible error decreases by at least the following amount when using \(\hat{\lambda}_i\) rather than \(\lambda_i\) in Algorithm 1

\[
H(y|\lambda, x) - H(y|\hat{\lambda}, x) \geq 2p_i(1 - p(\lambda_{-i}))^2 \cdot \bar{a}_i(r_i)^2.
\]

We aim to lower bound \(H(y|\lambda, x) - H(y|\hat{\lambda}, x)\) where only \(\lambda_i\) is extended to be \(\hat{\lambda}_i\) with threshold radius \(r_i\).

\[
H(y|\lambda, x) - H(y|\hat{\lambda}, x) = E_{x,y,\lambda} \left[ - \log Pr(y|\lambda, x) + E_{x,y,\lambda} \left[ \log Pr(y|\lambda, x) \right] \right] = E_{x,y,\lambda_{-i}} \left[ E_{\lambda_i} \left[ \log \frac{Pr(\lambda_i|x, y) Pr(\lambda_{-i}|x, y) Pr(y|x)}{Pr(\lambda_i, \lambda_{-i}|x)} \right] \right] - E_{\lambda_i} \left[ \log \frac{Pr(\lambda_i|x, y) Pr(\lambda_{-i}|x, y) Pr(y|x)}{Pr(\lambda_i, \lambda_{-i}|x)} \right].
\]

\[
(20)
\]

\(Pr(\lambda_{-i}|x, y)\) and \(Pr(y|x)\) are the same when using \(\hat{\lambda}_i\) versus \(\lambda_i\), so (20) becomes

\[
H(y|\lambda, x) - H(y|\hat{\lambda}, x) = E_{x,y,\lambda_{-i}} \left[ E_{\lambda_i} \left[ \log \frac{Pr(\lambda_i|x, y)}{Pr(\lambda_i, \lambda_{-i}|x)} \right] \right] - E_{\lambda_i} \left[ \log \frac{Pr(\lambda_i|x, y)}{Pr(\lambda_i, \lambda_{-i}|x)} \right].
\]

\[
(21)
\]

When extending \(\lambda_i\), there are three regions of interest in input space: \(\lambda_i(x), \hat{\lambda}_i(x) \neq 0; \lambda_i(x) = 0, \hat{\lambda}_i(x) \neq 0;\) and \(\lambda_i(x) = \hat{\lambda}_i(x) = 0\). In the first region, \(\hat{\lambda}_i\) has the exact same behavior as \(\lambda_i\) since \(\lambda_i\) has coverage over this region. Therefore, conditioning on \(\lambda_i(x) \neq 0\), the expectation on the RHS of (21) is equal to 0. Similarly, in the third region where \(\lambda_i(x) = \hat{\lambda}_i(x) = 0\), the extended and original labeler vote exactly the same, so the expectation on the RHS of (21) is again equal to 0. The primary region of interest are the points that previously had no signal from \(\lambda_i\) but now have signal from \(\hat{\lambda}_i\). Then, (21) becomes

\[
H(y|\lambda, x) - H(y|\hat{\lambda}, x) = p_i E_{y,\lambda_{-i}, \hat{\lambda}_i(x) \neq 0, \lambda_i(x) = 0} \left[ \log \frac{Pr(\lambda_i|x, y)}{Pr(\lambda_i, \lambda_{-i}|x)} - \log \frac{Pr(\lambda_i = 0|x, y)}{Pr(\lambda_i = 0, \lambda_{-i}|x)} \right].
\]

\[
(22)
\]

We can write \(\frac{Pr(\lambda_i = 0|x, y)}{Pr(\lambda_i = 0, \lambda_{-i}|x)} = \frac{Pr(\lambda_i = 0|x)}{Pr(\lambda_i = 0|x) Pr(\lambda_{-i}|x)} = 1 \frac{Pr(\lambda_i = 0|x)}{Pr(\lambda_{-i}|x)}\) by decomposing the denominator conditional on \(y\) and using Lemma 2. Using the chain rule on \(Pr(\lambda_i, \lambda_{-i}|x) = Pr(\lambda_i|\lambda_{-i}, x) Pr(\lambda_{-i}|x)\), (22) is now

\[
H(y|\lambda, x) - H(y|\hat{\lambda}, x) = p_i E_{y,\lambda_{-i}, \hat{\lambda}_i(x) \neq 0, \lambda_i(x) = 0} \left[ \log \frac{Pr(\hat{\lambda}_i|x, y)}{Pr(\lambda_i|x, y)} \right].
\]

To analyze this expectation, we first look at the case where \(y = 1\). Then,

\[
E_{y=1,\lambda_{-i}, \hat{\lambda}_i(x) \neq 0, \lambda_i(x) = 0} \left[ \log \frac{Pr(\hat{\lambda}_i|x, y)}{Pr(\lambda_i|x, y)} \right] = \left[ Pr(\hat{\lambda}_i = 1|x, y = 1) \log \frac{Pr(\hat{\lambda}_i = 1|x, y = 1)}{Pr(\lambda_i = 1|x, y = 1)} + Pr(\hat{\lambda}_i = -1|x, y = 1, \lambda_i) \neq 0 \log \frac{Pr(\lambda_i = -1|x, y = 1)}{Pr(\lambda_i = -1|x, y = 1)} \right].
\]

\[
(23)
\]
Denote \( \alpha_i(x) = \Pr(\lambda_i = 1 | y = 1, x, \tilde{\lambda}_i \neq 0) \) as the probability corresponding to \( \tilde{\lambda}_i \)'s accuracy parameter. In addition, note that we can write

\[
\Pr(\lambda_i = 1 | \lambda_i, x) = \Pr(\tilde{\lambda}_i = 1 | \lambda_i, x, y = 1) \Pr(y = 1 | \lambda_i, x) + \Pr(\tilde{\lambda}_i = 1 | \lambda_i, x, y = -1) \Pr(y = -1 | \lambda_i, x) = \alpha_i(x)p(x, \lambda_i) + (1 - \alpha_i(x))(1 - p(x, \lambda_i)),
\]

where \( p(x, \lambda_i) \) is shorthand for \( \Pr(y = 1 | \lambda_i, x) \) (importantly, it does not depend on \( \tilde{\lambda}_i \)) and likewise for \( \Pr(\tilde{\lambda}_i = -1 | \lambda_i, x) = \alpha_i(x)(1 - p(x, \lambda_i)) + (1 - \alpha_i(x))p(x, \lambda_i) \). Our expression from (23) is now

\[
\mathbb{E}_{y=1, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} \left[ \alpha_i(x) \log \frac{\alpha_i(x)p(x, \lambda_i) + (1 - \alpha_i(x))(1 - p(x, \lambda_i))}{(1 - \alpha_i(x))(1 - p(x, \lambda_i)) + \alpha_i(x)p(x, \lambda_i)} \right].
\]

Note that the expression inside the expectation is convex in both \( \alpha_i(x) \) and \( p(x, \lambda_i) \).

Define \( \alpha_{i,1} = \mathbb{E}_{y=1, \lambda_i(x) \neq 0, \alpha_i(x) = 0} [\alpha_i(x)] \) to be the expected accuracy probability over the extended region when \( y = 1 \), and \( p_{\lambda_i,1} = \mathbb{E}_{y'=-1, \lambda_i(x) \neq 0, \alpha_i(x) = 0} [\Pr(y = y' | x, \lambda_i)] \) to be the expected label model performance using just \( \lambda_i \) over the extended region when \( y = 1 \). Then, this expression from (24) is at least

\[
\alpha_{i,1} \log \frac{\alpha_{i,1}p_{\lambda_i,1} + (1 - \alpha_{i,1})(1 - p_{\lambda_i,1})}{1 - \alpha_{i,1}} + (1 - \alpha_{i,1}) \log \frac{\alpha_{i,1}(1 - p_{\lambda_i,1}) + (1 - \alpha_{i,1})p_{\lambda_i,1}}{1 - \alpha_{i,1}}.
\]

We look at the case where \( y = -1 \). Similarly, we get

\[
\mathbb{E}_{y=-1, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} \left[ \alpha_i(x) \log \frac{\alpha_i(x)(1 - p(x, \lambda_i)) + (1 - \alpha_i(x))p(x, \lambda_i)}{(1 - \alpha_i(x))(1 - p(x, \lambda_i)) + \alpha_i(x)p(x, \lambda_i)} \right].
\]

Again, define \( \alpha_{i,-1} = \mathbb{E}_{y=-1, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} [\alpha_i(x)] \) and \( p_{\lambda_i,-1} = \mathbb{E}_{y'=-1, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} [\Pr(y = y' | x, \lambda_i)] \), and by Jensen’s inequality we have that (26) is at least

\[
\alpha_{i,-1} \log \frac{\alpha_{i,-1}p_{\lambda_i,-1} + (1 - \alpha_{i,-1})(1 - p_{\lambda_i,-1})}{1 - \alpha_{i,-1}} + (1 - \alpha_{i,-1}) \log \frac{\alpha_{i,-1}(1 - p_{\lambda_i,-1}) + (1 - \alpha_{i,-1})p_{\lambda_i,-1}}{1 - \alpha_{i,-1}}.
\]

Therefore, \( \mathbb{E}_{y, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} \left[ \log \frac{\Pr(\lambda_i | y, x) \mathbb{P}(\tilde{\lambda}_i | \lambda_i, x)}{\Pr(\tilde{\lambda}_i | \lambda_i, x)} \right] \) is lower bounded by the weighted sum of \( \Pr(y = 1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \) times (25) and \( \Pr(y = -1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \) times (27). Since (25) and (27) are convex in \( \alpha_{i,1}, p_{\lambda_i,1} \) and \( \alpha_{i,-1}, p_{\lambda_i,-1} \) respectively, we can define \( \alpha_i = \Pr(y = 1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \cdot \alpha_{i,1} + \Pr(y = -1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \cdot \alpha_{i,-1} = \mathbb{E}_{\lambda_i(x) \neq 0, \alpha_i(x) = 0} [\alpha_i(x)] \) as a notion of \( \hat{\lambda}_i \)'s accuracy in the region where we extend \( \lambda_i \). We also define \( p_{\lambda_i} = \Pr(y = 1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \cdot p_{\lambda_i,1} + \Pr(y = -1 | \lambda_i, \tilde{\lambda}_i(x) \neq 0, \lambda_i(x) = 0) \cdot p_{\lambda_i,-1} = \mathbb{E}_{y, \lambda_i, \tilde{\lambda}_i(x) \neq 0, \alpha_i(x) = 0} [\Pr(y = y' | \lambda_i, x)] \) as the label model’s probability of outputting the correct label in our region of interest when relying on only \( \lambda_i \). Then, we have that

\[
H(y | \lambda, x) - H(y | \tilde{\lambda}, x) \geq p_i \left( \alpha_i \log \frac{\alpha_i}{\alpha_ip_{\lambda_i} + (1 - \alpha_i)(1 - p_{\lambda_i})} + (1 - \alpha_i) \log \frac{1 - \alpha_i}{(1 - \alpha_i)p_{\lambda_i} + \alpha_i(1 - p_{\lambda_i})} \right).
\]

We can lower bound the expression in the parentheses. Define \( g(x) = x \log \frac{x}{xp + (1 - x)p(1 - p)} + (1 - x) \log \frac{1 - x}{(1 - x)p + x(1 - p)} \) for some constant \( p \). We claim that \( g(x) \geq h(x) = 8(1 - p)^2(x - 0.5)^2 \) for \( x \in [0, 1] \). Note that \( g(0.5) = h(0.5) = 0 \).
To show that \( g(x) \geq h(x) \), it suffices to show that \( g'(x) > h'(x) \) for \( x > 0.5 \), and \( g'(x) < h'(x) \) for \( x < 0.5 \).\n
\[ g'(x) = \frac{1-p}{xp^+(1-x)(1-p)} + \frac{p-1}{x(1-p)(1-x)} + \log \frac{xp^+(1-x)(1-p)}{1-x} - \log \frac{1-x}{(1-x)p+x(1-p)}, \]
and \( h'(x) = 16(1-p)^2(x-0.5) \). Again, note that \( g'(0.5) = h'(0.5) = 0 \), so we want to show that \( g''(x) > h''(x) \) for all \( x \in [0, 1] \). \( g''(x) = -\frac{(1-p)(2p-1)}{(xp^+(1-x)(1-p))^2} - \frac{(p-1)(1-2p)}{(x(1-p)+(1-x)p)^2} + \frac{1-p}{x(xp^+(1-x)(1-p))} + \frac{1-p}{(1-x)(x(1-p)+(1-x)p)}, \)
and \( h''(x) = 16(1-p)^2 \). It is easy to check that \( g''(x) \) obtains a minimum at \( x = 0.5 \). We compute that \( g''(0.5) = 16(1-p)^2 \), which demonstrates that \( g(x) > 8(1-p)^2(x-0.5)^2 \). We thus get

\[ H(y|x, x) - H(y|\bar{x}, x) \geq 8p_i(1 - p_{\lambda_{-i}})^2 \left( \alpha_i - \frac{1}{2} \right)^2. \]

We know that \( \alpha_i = \frac{1 + \bar{a}_i(r_i)}{2} \), so our final bound is

\[ H(y|x, x) - H(y|\bar{x}, x) \geq 8p_i(1 - p_{\lambda_{-i}})^2 \cdot \frac{\bar{a}_i(r_i)^2}{4} = 2p_i(1 - p_{\lambda_{-i}})^2 \cdot \bar{a}_i(r_i)^2. \]

E EXPERIMENTAL DETAILS

We describe additional details about each task, including details about data sources (Section E.1), supervision sources (Section E.2), and setting extension thresholds (Section E.3).

E.1 DATASET DETAILS

<table>
<thead>
<tr>
<th>Task (Embedding)</th>
<th>T</th>
<th>( m/T )</th>
<th>Prop</th>
<th>( N_{\text{train}} )</th>
<th>( N_{\text{dev}} )</th>
<th>( N_{\text{test}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spam</td>
<td>1</td>
<td>10</td>
<td>0.49</td>
<td>1,586</td>
<td>120</td>
<td>250</td>
</tr>
<tr>
<td>Weather</td>
<td>1</td>
<td>103</td>
<td>0.53</td>
<td>187</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Spouse</td>
<td>1</td>
<td>9</td>
<td>0.07</td>
<td>22,254</td>
<td>2,811</td>
<td>2,701</td>
</tr>
<tr>
<td>Basketball</td>
<td>8</td>
<td>4</td>
<td>0.12</td>
<td>3,594</td>
<td>212</td>
<td>244</td>
</tr>
<tr>
<td>Commercial</td>
<td>3</td>
<td>4</td>
<td>0.32</td>
<td>64,130</td>
<td>9,479</td>
<td>7,496</td>
</tr>
<tr>
<td>Tennis</td>
<td>9</td>
<td>6</td>
<td>0.34</td>
<td>6,959</td>
<td>746</td>
<td>1,098</td>
</tr>
</tbody>
</table>

Table 4: Details for each dataset. \( T \): the number of related elements modeled by the weak supervision label model. \( m/T \): the number of supervision sources per element. Prop: The proportion of positive examples in each dataset. \( N_{\text{train}} \): The size of the unlabeled training set. \( N_{\text{dev}} \): The size of the labeled dev set. \( N_{\text{test}} \): The size of the held-out test set.

Table 4 provides details on train/dev/test splits for each dataset, as well as statistics about the positive class proportion and the number of labeling functions. Additional details about each dataset are provided below.

Spam We use the dataset as provided by Snorkel\(^1\) and those train/dev/test splits.

Weather, Spouse These datasets are used in Ratner et al. [2019] and Fu et al. [2020] for evaluation, and we use the train/dev/test splits from those works (Weather is called Crowd in that work).

Basketball This dataset is a subset of ActivityNet and was used for evaluation in Sala et al. [2019] and Fu et al. [2020]. We use the train/dev/test splits from those works.

Commercial We use the dataset from Fu et al. [2019], Hong et al. [2021] and Fu et al. [2020] and the train/dev/test splits from those works.

Tennis We use the dataset from Fu et al. [2020] and the train/dev/test splits from those works.

\(^1\)https://www.snorkel.org/use-cases/01-spam-tutorial
E.2 SUPERVISION SOURCES

Supervision sources are expressed as short Python functions. Each source relied on different information to assign noisy labels:

**Spam, Weather, Spouse** For these tasks, we used the same supervision sources as used in previous work [Ratner et al., 2018][Fu et al., 2020]. These are all text classification tasks, so they rely on text-based heuristics such as the presence or absence of certain words, or particular regex patterns.

**Basketball, Commercial, Tennis** Again, we use sources from previous work [Sala et al., 2019][Fu et al., 2020]. For Basketball, these sources rely on an off-the-shelf object detector to detect balls or people, and use heuristics based on the average pixel of the detected ball or distance between the ball and person to determine whether the sport being played is basketball or not. For Commercial, there is a strong signal for the presence or absence of commercials in pixel histograms and the text; in particular, commercials are book-ended on either side by sequences of black frames, and commercial segments tend to have mixed-case or missing transcripts (whereas news segments are in all caps). For Tennis, we use an off-the-shelf pose detector to provide primitives for the weak supervision sources. The supervision sources are heuristics based on the number of people on court and their positions. Additional supervision sources use color histograms of the frames (i.e., how green the frame is, or whether there are enough white pixels for the court markings to be shown).

E.3 SETTING $r_i$ AND $s$

We tune $r_i$ using the dev set in two steps. First, we set all the $r_i$ to the same value $r$ and use grid search over $r$. Then, we perform a series of small per-coordinate searches for a subset of the labeling functions to optimize individual $r_i$ values. For labeling functions with full coverage, we set the threshold to have no extensions.

Tuning $s$ is done independently from $r_i$. Once we have the best performing $r_i$ values, we search for the best possible $s$ from one to ten. We obtain the partition by performing K-means clustering with $K = s$.

Now we report thresholds in terms of cosine similarities (note that this is a different presentation than in terms of distances). For Spam, all thresholds are set to 0.844, except for weak sources 1, 2, and 7, which have thresholds 0.864, 0.854 and 0.804 respectively. The best $s$ is 2. For Weather, all thresholds are set to 0.2, and the best $s$ is 3. For Spouse, all thresholds are set to 0.9275, except for weak sources 2 and 3, which have thresholds 0.8385 and 0.9. The best $s$ is 8. For Basketball, thresholds are set to $[0.42, 0.97, 0.52, 0.42]$ and $s$ is set to 2. For Commercial, thresholds are set to $[.6, .35, .35, .65]$ and $s$ is set to 3. For Tennis, thresholds are set to $[0.11, 0.110.11, 0.85, 0.11, 0.11]$ and $s$ is set to 2.

In our experiments, class balance $Pr(y|C_j)$ is estimated from the dev set.

E.4 ADAPTERS

We describe adapter experimental details in the main results. For each dataset, we train single-layer adapters with gradient descent. Because this requires training labels, we consider two training setups: (1) splitting the validation set into a new 80% training set and 20% held-out validation set, and (2) using weak-supervision methods (WS-LM) combined with labeling functions to generate pseudolabels for the training data.

For both, we train adapters using the OpenAI GPT-3 Ada embeddings for NLP tasks and OpenAI CLIP embeddings for video tasks. We train with 50 epochs and early stopping, and sweep over the following hyperparameters: learning rate $\in \{1.0 - 3, 1.0 - 2, 1.0 - 1\}$, weight decay $\in \{5.0 - 4, 0\}$, momentum $\in \{0, 0.9\}$.

The best performing model (based on held-out validation set accuracy for Spam and Weather datasets, held-out validation F1-score for all other datasets), was then evaluated on the test set.

For the linear models, the best hyperparameters are as follows: for Spam, we use $1.0 - 1$ learning rate, $5.0 - 4$ weight decay, and 0.9 momentum. For Weather, we use $1.0 - 1$ learning rate, $5.0 - 4$ weight decay, and 0.9 momentum. For Spouse, we use $1.0 - 2$ learning rate, $5.0 - 4$ weight decay, and 0 momentum. For Basketball, we use $1.0 - 3$ learning rate, 0 weight decay, and 0.9 momentum. For Commercial, we use $1.0 - 1$ learning rate, $5.0 - 4$ weight decay, and 0 momentum. For Tennis, we use $1.0 - 1$ learning rate, $5.0 - 4$ weight decay, and 0 momentum.

For the MLPs, the best hyperparameters are as follows: for Spam, we use 0.1 learning rate, 0 weight decay, 0.9 momentum,
and 512 hidden layer dimension. For **Weather**, we use $1e-1$ learning rate, $5e-4$ weight decay, 0.9 momentum, and 256 hidden layer dimension. For **Spouse**, we use $1e-3$ learning rate, 0 weight decay, 0 momentum, and 256 hidden layer dimension. For **Basketball**, we use $1e-2$ learning rate, 0 weight decay, 0.9 momentum, and 512 hidden layer dimension. For **Commercial**, we use $1e-1$ learning rate, $5e-4$ weight decay, 0.9 momentum, and 512 hidden layer dimension. For **Tennis**, we use $1e-2$ learning rate, $5e-4$ weight decay, 0.9 momentum, and 256 hidden layer dimension.

**LIGER-Adapter**  In addition to evaluating LIGER on its own against linear adapters, we also demonstrate further boosts when combining the LIGER predictions with Adapters. For this approach, we first create training sets by combining the 80% split of the original validation set and the original training set. To get labels, we use the ground-truth labels for the former, and the LIGER predictions on the training set for the latter. To get data inputs, we tune between using the same data embeddings as in the original datasets, and optionally concatenating the LIGER predictions as an additional input dimension to the embeddings. In the setup, for validation and test sets, we also concatenate the LIGER predictions to the embeddings. For the **Spouse** dataset, we do this concatenation, as we found it to improve the validation set F1-score. For all others, we use the original embeddings. When the weak labels are not very accurate (<75% accuracy on dev), we downsample the train points (otherwise they would degrade performance from ground-truth dev labels). This allows performance on **Basketball** to be strong even though LIGER accuracy is relatively low.

We tune hyperparameters in the same way as the other adapters. The best hyperparameters are as follows: for **Spam**, we use 0.1 learning rate, 0 weight decay, 0.9 momentum. For **Weather**, we use 0.1 learning rate, $5e-4$ weight decay, 0.9 momentum. For **Spouse**, we use $1e-3$ learning rate, $5e-4$ weight decay, 0.9 momentum. For **Basketball**, we use 10.1 learning rate, $5e-4$ weight decay, 0 momentum. For **Commercial**, we use 0.1 learning rate, 0 weight decay, 0.9. For **Tennis**, we use $1e-1$ learning rate, $5e-4$ weight decay, 0 momentum.

### F ADDITIONAL EXPERIMENTAL RESULTS

#### F.1 MLP ADAPTERS

<table>
<thead>
<tr>
<th>Task</th>
<th>LIGER (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spam</td>
<td>96.8 (2)</td>
</tr>
<tr>
<td>Weather</td>
<td>95.3 (3)</td>
</tr>
<tr>
<td>Spouse</td>
<td>17.0 (6)</td>
</tr>
<tr>
<td>Basketball</td>
<td>81.7 (2)</td>
</tr>
<tr>
<td>Commercial</td>
<td>93.4 (3)</td>
</tr>
<tr>
<td>Tennis</td>
<td>83.4 (1)</td>
</tr>
</tbody>
</table>

Table 5: MLP Adapter performance. Scores are in F1, except for Spam and Weather (accuracy).

We also evaluated adapters using 3-layer MLPs as alternatives to the linear adapters. We considered MLPs with 512 or 256 dimensional hidden-layers with the ReLU nonlinear activation function. We report the results in Table 5. Performance is similar to the linear adapters, but the MLP adapters are slightly more expensive to train. We focus on a simple linear probe for LIGER-Adapter and the main experiments for simplicity.

#### F.2 ADDITIONAL MEASURES OF SMOOTHNESS

Figure 3 reports two additional measurements of smoothness on **Basketball** and **Spouse**—coverage Lipschitzness and local label probabilistic Lipschitzness (see Section 4 for the formal definitions). Trends match label Lipschitzness.

To measure label Lipschitzness, the property that $|\Pr(y = 1|x) - \Pr(y = 1|x')| \leq K_y\rho(f(x), f(x'))$, we observe that

$$|\Pr(y = 1|x) - \Pr(y = 1|x')| = |\mathbb{E} [\mathbb{1} \{ y = 1|x \} - \mathbb{E} [\mathbb{1} \{ y = 1|x' \}]]|$$

$$\leq \mathbb{E} [|\mathbb{1} \{ y = 1|x \} - \mathbb{1} \{ y = 1|x' \}]|$$

$$= \mathbb{E} [\mathbb{1} \{ y \neq y' \}] = \Pr(y \neq y')$$
We evaluate LIGER on synthetic data to confirm our insights about 1) how generalization error for $\hat{P}_{\lambda}(y|x)$ demonstrates a bias-variance tradeoff depending on the number of partitions, and 2) how additional lift depends on setting the threshold radius based on the original weak source’s accuracy and the embedding’s probabilistic Lipschitzness.

First, we conduct a synthetic experiment to understand how the number of partitions $s$ controls the bias-variance tradeoff in generalization error of $\hat{P}_{\lambda}(y|x)$ (Theorem 1). We generate two sets of canonical parameters and use them in (1) to generate $(y, \lambda)$ from two different distributions, $P_1$ and $P_2$ over an embedding space. We generate 1000 points each for $P_1$ and $P_2$ to form datasets $D_1$ and $D_2$, which are then concatenated to form a dataset $D$ of 2000 points. We first run Algorithm 1 with $s = 1$, which means that we estimate only one set of parameters over $D$ despite the dataset consisting of two different conditional distributions. We then set $s = 2$ and estimate the parameters of $P_1$ and $P_2$ separately over 1000 points each. Finally, we set $s = 4$ and $s = 8$ by dividing each of $D_1$ and $D_2$ into 2 subsets of 500 points and 4 subsets of 250 points, respectively. For each of these, we compute the average cross-entropy loss (over $s$) of our label model. Figure 4 plots how the generalization error changes with the number of partitions $s$. We plot the mean and 95% confidence interval over ten random
Figure 4: The bias-variance tradeoff in the generalization error based on $s$, the number of partitions used in our approach. When too few partitions are used, the accuracy estimates are not fine-grained and do not sufficiently approximate the true conditional distribution $\Pr(y, \lambda|x)$, resulting in large bias. When too many partitions are used, the variance increases due to sampling error on individual partitions.

Figure 5: Reduction in generalization error from extending labeling functions of varying accuracies (left), and on embedding spaces of varying smoothness (right). LF refers to a weak source’s labeling function.

initializations of canonical parameters and datasets drawn according to them. It demonstrates a bias-variance tradeoff: when $s = 1$, we estimate one set of parameters over the entire dataset rather than the two true sets of parameters, and this approach hence does not capture the distinctions in input space among the source accuracies. As a result, a low $s$ results in high bias, contributing to large generalization error. On the other hand, when $s = 4$ or $8$, our approach is correctly estimating $D_1$ and $D_2$ separately but is using much less data to do so. This approach has higher sampling error, which worsens variance and contributes to large generalization error.

Next, we conduct a synthetic experiment to understand how setting the threshold radius of the extended weak source controls improvement in generalization error as a function of the original average source accuracy and the probabilistic Lipschitzness of the FM embedding (Theorem 3). Suppose for simplicity that $s = 1$ and that $\Pr(y, \lambda)$ is modeled the same way as $\Pr(y, \lambda|x)$ in (1). This assumption reduces to previous weak supervision settings but allows us to isolate the effect of extending a source. We create an embedding space over 10000 uniformly sampled points in $[0, 1]^2$ with a fixed class balance $\Pr(y)$ and $m = 3$ labeling functions, where only $\lambda_1$ is extended. To understand the impact of a labeling function’s accuracy, we fix a task distribution by assigning $Y$ labels in a $10 \times 10$ “checkerboard” pattern and run our algorithm on four versions of $\lambda_1$ with varying average accuracies, keeping $\lambda_1$’s support consistent. In Figure 5 (left), we extend $\lambda_1$ based on $r$ for each of the four versions of the labeling function. This confirms that extending a highly accurate labeling function results in greater generalization lift. To understand the impact of Lipschitzness of the task distribution, we produce four distributions of $Y$ over the embedding space, three of which follow a checkerboard clustering pattern (such that more divisions mean less smoothness), and one that spatially distributes the values of $Y$ at random. For both experiments, we run our approach with threshold radius varying from 0 to 0.1 in increments of 0.005. In Figure 5 (right), each curve represents performance of the same high average accuracy labeling function ($a_1 = 0.89$) over embeddings of varying Lipschitzness. This confirms that the greatest improvement due to an extension occurs for the smoothest embedding. Lastly, both of these graphs illustrate the
tradeoff in setting a threshold radius, confirming the theoretical insight that this quantity must be chosen carefully to ensure lift from using $\lambda_1$ over $\lambda$.

References


