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

*Equal Contribution. A preliminary version of the results in this paper can be found at https://arxiv.org/abs/2006.15168.
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.

| Symbol | Used for |
|--------|----------|
| $x$    | Input data point $x \in X$. |
| $y$    | True task label $y \in \mathcal{Y} = \{-1, +1\}$. |
| $\lambda$ | Weak sources $\lambda = \{\lambda_1, \ldots, \lambda_m\}$, where each $\lambda_j : X \to \mathcal{Y} \cup \{0\}$ is a probabilistic labeling function that votes on each $x$. |
| $m$    | Number of weak sources. |
| $f$    | A fixed mapping from input space $X$ to embedding space $Z$ that is made available by the off-the-shelf foundation model. |
| $\rho$ | A fixed metric on the embedding space, $\rho : Z \times Z \to \mathbb{R}^+$. |
| $D$    | A training dataset of $n$ i.i.d. unlabeled points, $D = \{x_i\}_{i=1}^n$. |
| $n$    | Number of points in the unlabeled training dataset $D$. |
| $G$    | The dependency graph $G = (V, E)$ used to model $\Pr(y, \lambda|x)$, where $V = \mathcal{Y} \cup \lambda$ and $E$ contains edges between $y$ and $\lambda$. |
| $\Theta(x)$ | The set of canonical parameters $\Theta(x) = \{\theta_y(x), \theta_i(x), \theta_{i,0}(x) \forall i \in [m]\}$ corresponding to class balance, source accuracy, and the abstain rate used to parametrize $\Pr(y, \lambda|x)$ in $[1]$. |
| $Z$    | Partition function used for normalizing the distribution of $\Pr(y, \lambda|x)$. |
| $a_\lambda(x)$ | Accuracy parameter of $\lambda_i$ on point $x$, $a_\lambda(x) = \mathbb{E}[\lambda_i y | \lambda_i \neq 0, x]$. |
| $C$    | Partition of the embedding space $Z$ into nonoverlapping subsets, $C = \{C_1, \ldots, C_s\}$. |
| $s$    | Size of the partition $C$. |
| $n'$   | The number of points from $D$ in each subset $C_j$, $n' = \frac{n}{s}$. |
| $C(x)$ | The subset that $x$ belongs to, i.e. $C(x) = C_j$ if $f(x) \in C_j$. |
| $a_i(C(x))$ | Local accuracy parameter of $\lambda_i$ on subset $C(x)$, $a_i(C(x)) = \mathbb{E}[\lambda_i y | \lambda_i \neq 0, C(x)]$. |
| $\hat{a}_i(C(x))$ | Our local accuracy estimate of $\lambda_i$ on subset $C(x)$ using the triplet method in Algorithm 2. |
| $\bar{\lambda}$ | Set of extended weak sources, where each $\bar{\lambda}_i$ is extended from $\lambda_i$ using threshold radius $r_i$ in $[3]$. |
| $r_i$  | Threshold radius for $\lambda_i$, which determines how much beyond the support of $\lambda_i$ to extend votes to. |
| $L(\lambda)$ | Generalization error (cross-entropy loss) of the label model, defined as $L(\lambda) = \mathbb{E}_{D,x,y,\lambda} \left[ -\log \Pr(y|\lambda, x) \right]$. |
| $K_y, K_\lambda, K_{\lambda,0}$ | Constants in Definition 1 corresponding to label, source, and coverage Lipschitzness, respectively. |
| $\alpha$ | The maximum average inverse source coverage over the subsets, $\alpha = \max_i \mathbb{E}_x \left[ \frac{1}{p_{ij}} | p_{ij} \neq 0 \right]$, where $p_{ij} = \Pr(\lambda_i \neq 0, f(x) \in C_j)$ is the coverage of $\lambda_i$ on $C_j$. |
| $a_{\max}$ | The maximum source accuracy over the subsets, $a_{\max} = \max_{i,j} a_i(C_j)$. |
| $b_{\min}$ | The minimum rate of agreement between sources over the subsets, $b_{\min} = \min_{i,j,k} \mathbb{E}[\lambda_i \lambda_j | \lambda_i \land \lambda_k \neq 0, C_j, \lambda_i \lambda_k | \lambda_i \land \lambda_k \neq 0, C_j]$. |
| $d_{\max}$ | The diameter of $C_j$, $d_{\max} = \max_{f(x), f(x') \in C_j} \rho(f(x), f(x'))$. |
| $d_C$  | The average subset diameter $d_C = \mathbb{E}_x [d_{C(x)}]$. |
| $H(y|\lambda, x)$ | Conditional entropy of $y$ given $\lambda, x$. |
| $a_i$  | The average accuracy of $\lambda_i$, $a_i = \mathbb{E}[\lambda_i y | \lambda_i \neq 0]$. |
| $\bar{a}_i(r_i)$ | The average accuracy of $\lambda_i$ on the extended region, $\bar{a}_i(r_i) = \mathbb{E} [\lambda_i y | \lambda_i \neq 0, \lambda_i = 0]$. |
| $\mathcal{P}_{\lambda_i}$ | The distribution of $(x, y)$ over the support of $\lambda_i$, $\mathcal{P}_{\lambda_i} = \Pr(\cdot | \lambda_i \neq 0)$. |
| $M$    | An increasing function $M : \mathbb{R}^+ \to [0, 1]$ used to describe probabilistic Lipschitzness. |
| $\beta_i$ | $\lambda_i$’s accuracy over an area close to where $\lambda_i$ is extended and $y$ changes value, $\beta_i = \mathbb{E}[\lambda_i y | \lambda_i \neq 0, \exists (x', y') : \lambda_i(x') = 0, \rho(f(x), f(x')) \leq r_i, y' = y]$. |
| $p_i$  | The proportion of the region where $\lambda_i$ is extended, $p_i = \Pr(\lambda_i \neq 0, \lambda_i = 0)$. |
| $p(\lambda_{-i})$ | The label model’s true probability of outputting the correct label in the extension region when only using $\lambda_{-i} = \lambda \setminus \lambda_i$, $p(\lambda_{-i}) = \mathbb{E}_{y', \lambda_{-i}, \lambda_i \neq 0, \lambda_i = 0} [\Pr(y = y'| \lambda_{-i}, x)]$. |
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, \( y \perp \mathbb{I}\{\lambda_i = 0\}| x \).

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 \( \Pr(y = 1, \lambda_i = 0|x) = \Pr(y = 1|x) \Pr(\lambda_i = 0|x) \):

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

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

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

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

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

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

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

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

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

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

Setting (1) equal to (3) 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 $\lambda_i$. 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

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

and hence

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

\[\blacklozenge\]

Lemma 3. For any $i \neq j$, if $x, \lambda, y$ follows \([\text{I}]\), 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 \([\text{I}]\). This graphical model now follows the structure of the graphical model in \cite{fu2020} (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. \[\blacklozenge\]

Lemma 4. If $x, \lambda, y$ follows \([\text{I}]\), then for any $\lambda_i$,

$$
\Pr(\lambda_i = 1|y = 1, \lambda_i \neq 0, x) = \Pr(\lambda_i = -1|y = -1, \lambda_i \neq 0, x) = \Pr(\lambda_i = 1|\lambda_i \neq 0, x) \tag{9}
$$

$$
\Pr(\lambda_i = -1|y = 1, \lambda_i \neq 0, x) = \Pr(\lambda_i = 1|y = -1, \lambda_i \neq 0, x) = \Pr(\lambda_i = -1|\lambda_i \neq 0, x). \tag{10}
$$

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 \cite{chen2021}, and therefore we obtain our desired properties by Lemma 2 of their work. \[\blacklozenge\]

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 $\mathbb{E} \left[ \lambda_i y|x, \lambda_i \neq 0 \right] = \Pr(\lambda_i y = 1|x, \lambda_i \neq 0) - \Pr(\lambda_i y = -1|x, \lambda_i \neq 0) = 2\Pr(\lambda_i y = 1|x, \lambda_i \neq 0) - 1$, 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 $a_i(C_j)$. From Lemma 3, we have that $\lambda_i, \lambda_j, \lambda_k$ satisfy (1) for any $i, j$. Then, given any set of $\lambda_i, \lambda_j, \lambda_k$, we have the set of equations

\[
\begin{align*}
a_i(x)a_j(x) &= \mathbb{E} [\lambda_i \lambda_j \lambda_i \land \lambda_j = 0, x] \\
a_i(x)a_k(x) &= \mathbb{E} [\lambda_i \lambda_k \lambda_i \land \lambda_k = 0, x] \\
a_j(x)a_k(x) &= \mathbb{E} [\lambda_j \lambda_k \lambda_j \land \lambda_k = 0, x].
\end{align*}
\]

Solving, we get that

\[
|a_i(x)| = \sqrt{\frac{\mathbb{E} [\lambda_i \lambda_j \lambda_i \land \lambda_j = 0, x] \mathbb{E} [\lambda_i \lambda_k \lambda_i \land \lambda_k = 0, x]}{\mathbb{E} [\lambda_j \lambda_k \lambda_j \land \lambda_k = 0, x]}}.
\]

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 $\lambda_i, \lambda_j, \lambda_k$ (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 $a_i(C_j)$ over all other pairs $(\lambda_j, \lambda_k \in \lambda \setminus \lambda_i)$ to make the estimate less noisy. Our approach for computing $\lambda_i(C_j)$ 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 $(y, x, \lambda)$ both satisfy (1)) is described in Algorithm 2.

\section{Proofs}

We present the proofs for our results in Section 4.

\subsection{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).

\textbf{Theorem 1.} Suppose that data $x, y, \lambda$ follow the model in (1) and $\mathbb{P}(y|x)$ and $\mathbb{P}(\lambda_i|x)$ for each $\lambda_i$ are Lipschitz-smooth. The generalization error of the label model $\mathbb{P}(y|\lambda, x)$ 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

\[
\begin{align*}
\text{Bias} &\leq 2d_K(K_y + mK_\lambda + mK_{\lambda,0}), \\
\text{Variance} &\leq \frac{m}{n} \left( 3\alpha(1 - b_{\text{min}}^2) \left( \frac{1}{b_{\text{min}}^2} + \frac{2}{b_{\text{min}}^2} \right) + 1 \right), \\
\text{Irreducible Error} &\leq H(y|\lambda, x),
\end{align*}
\]

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

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

\(-\mathbb{E}_{x,y,\lambda} [\Pr(y|\lambda, x)]\) 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 \(\Pr(\lambda_i|y, C(x))\) to get

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

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

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

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

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

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

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

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

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

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

where \(z_i = 1 \{\lambda_i = 0\}\) 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

\[ \mathbb{E}_x \left[ D_{KL}(\Pr(z_i|x)||\hat{\Pr}(z_i|C(x))) \right] = KL_{C(x)}(z_i) \]

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

\[ = KL_{C(x)}(z_i) + \mathbb{E}_{C(x)} \left[ \Pr(\lambda_i = 0|C(x)) \log \frac{\Pr(\lambda_i = 0|C(x))}{\hat{\Pr}(\lambda_i = 0|C(x))} + \Pr(\lambda_i \neq 0|C(x)) \log \frac{\Pr(\lambda_i \neq 0|C(x))}{\hat{\Pr}(\lambda_i \neq 0|C(x))} \right] \]

\[ = KL_{C(x)}(z_i) + \mathbb{E}_{C(x),z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\hat{\Pr}(z_i|C(x))} \right], \]
We can interpret the generalization error as consisting of bias, variance (and irreducible error) coming from 1) estimating the bias, variance coming from 2) estimating the sampling error term coming from 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 \( \mathcal{D} \). Then,

\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \mathbb{E}_{\mathcal{D}} \left[ p_{ij} \log \frac{p_{ij}}{\hat{p}_{ij}} + (1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - \hat{p}_{ij}} \right]. \tag{13}
\]

Performing a Taylor approximation of \( q(x) = \log \frac{\hat{x}}{x} \) at \( x = c \) gives us \( \log \frac{\hat{x}}{x} \approx \log 1 + \frac{1}{c}(x - c) + \frac{1}{2c^2}(x - c)^2 \). Setting \( x = \hat{p}_{ij}, 1 - \hat{p}_{ij} \) and \( c = p_{ij}, 1 - p_{ij} \) respectively in (13) and using the fact that \( \hat{p}_{ij} \) is an unbiased estimate of \( p_{ij} \), this expression becomes

\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{p_{ij}(1 - p_{ij})} \mathbb{E} \left[ (p_{ij} - \hat{p}_{ij})^2 \right] + o(1/n)
\]

\[
= \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{p_{ij}(1 - p_{ij})} \text{Var} [\hat{p}_{ij}] + o(1/n),
\]

where we use the fact that the Taylor remainder scales in \( \mathbb{E} \left[ (\hat{p}_{ij} - p_{ij})^3 | C_j \right] \sim \mathcal{O}(1/n^2) \). We can simplify the variance

\[
\text{Var} [\hat{p}_{ij}] = \text{Var} \left[ \frac{1}{n} \sum_{x:f(x) \in C_j} \mathbb{1} \{ \lambda_i(x) \neq 0 \} \right] = \frac{1}{n^2} \sum_{x:f(x) \in C_j} \text{Var} \left[ \mathbb{1} \{ \lambda_i(x) \neq 0 \} \right] = \frac{p_{ij}(1 - p_{ij})}{n^2}.
\]

Putting this all together, we have

\[
\mathbb{E}_{C(x), z_i} \left[ \log \frac{\Pr(z_i|C(x))}{\Pr(z_i|C(x))} \right] = \sum_{j=1}^{s} \Pr(f(x) \in C_j) \frac{1}{n'} + o(1/n) = \frac{s}{n} + o(1/n).
\]
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

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

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 $\mathbb{E} \left[ \log 1 + \frac{1 + \sgn(\lambda_i y - a_i(C(x)))}{1 - \sgn(\lambda_i y - a_i(C(x)))} \right] = \sum_{C_j \in C_i} \Pr(f(x) \in C_j) \mathbb{E} \left[ \log 1 + \frac{1 + \sgn(\lambda_i y - a_i(C_j))}{1 - \sgn(\lambda_i y - a_i(C_j))} \right].$ We can decompose the expectation as

$$\mathbb{E}_{x,y,\lambda_i \neq 0} \left[ \log 1 + \frac{1 + \sgn(\lambda_i y - a_i(C_j))}{1 + \sgn(\lambda_i y - a_i(C_j))} \left| C_j \right. \right] = \mathbb{E} \left[ \log 1 + \frac{1 + a_i(C_j)}{1 + a_i(C_j)} \left| C_j \right. \right] \Pr(\lambda_i y = 1 | C_j, \lambda_i \neq 0) \quad (14)$$

$$+ \mathbb{E} \left[ \log 1 + \frac{1 - a_i(C_j)}{1 - a_i(C_j)} \left| C_j \right. \right] \Pr(\lambda_i y = -1 | C_j, \lambda_i \neq 0). \quad (15)$$

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

$$\mathbb{E} \left[ \log 1 + \frac{1 + a_i(C_j)}{1 + a_i(C_j)} \left| C_j \right. \right] = \frac{1}{2} (1 + a_i(C_j)) \mathbb{E} \left[ \log 1 + \frac{1 + a_i(C_j)}{1 + a_i(C_j)} \left| C_j \right. \right] + (1 - a_i(C_j)) \mathbb{E} \left[ \log 1 + \frac{1 - a_i(C_j)}{1 - a_i(C_j)} \left| C_j \right. \right]. \quad (16)$$

Again, we can perform a Taylor expansion on $g(x) = \log 1 + \frac{1 + x}{1 - x}$ at $x = c$ to get that $\log 1 + \frac{1 + c}{1 + c} \approx -\frac{1}{1 + c}(x - c) + \frac{1}{2(1 + c)^2}(x - c)^2,$ and therefore

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

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

$$\mathbb{E} \left[ \log 1 + \frac{1 + \sgn(\lambda_i y - a_i(C_j))}{1 + \sgn(\lambda_i y - a_i(C_j))} \left| C_j \right. \right] = \frac{1}{2} \mathbb{E} \left[ a_i(C_j) - \hat{a}_i(C_j) \right] + \mathbb{E} \left[ \frac{(\hat{a}_i(C_j) - a_i(C_j))^2}{2(1 + a_i(C_j)^2)} \right] \quad (17)$$

$$+ \mathbb{E} \left[ \hat{a}_i(C_j) - a_i(C_j) \right] + \mathbb{E} \left[ \frac{(\hat{a}_i(C_j) - a_i(C_j))^2}{2(1 - a_i(C_j)^2)} \right] + o(1/n).$$

The value of $\mathbb{E} \left[ (\hat{a}_i(C_j) - a_i(C_j))^2 \right]$ 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

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

Therefore, the overall expression can be bounded by

$$\mathbb{E} \left[ \log 1 + \frac{1 + \sgn(\lambda_i y - a_i(C(x)))}{1 + \sgn(\lambda_i y - a_i(C(x)))} \right] \leq \sum_{C_j \in C_i} \Pr(f(x) \in C_j) \cdot \frac{1}{2(1 - a_{\max}^2)} \cdot \frac{3s}{4p_{ij} n} \cdot \frac{1 - b_{\min}^2}{b_{\min}^2} \left( \frac{1}{b_{\min}^2} + \frac{2}{b_{\min}^2} \right) + o(\frac{1}{n})$$

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

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

\[
KL_{C(x)}(y) \leq 2K_y d_C \\
KL_{C(x)}(\lambda_i|y, \lambda_i \neq 0) \leq 2m K_\lambda d_C \\
KL_{C(x)}(z_i) \leq 2m K_{\lambda,0} d_C.
\]

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

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

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),

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

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

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

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

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 \( \bar{X} \) instead of \( X \), the bias term in \( L(\bar{X}) \) is at most

\[
Bias \leq 2d_C K_y + 2m(d_C + 2 \max_i r_i)(K_\lambda + K_{\lambda,0}).
\]
We consider $\lambda$. Next, we recall the definition of $\hat{\lambda}$.

\[ \text{decompose $\hat{\lambda}$, also be written as a set of} \]
\[ \text{support be} \]
\[ \text{Define the extended region as} \]
\[ \text{Suppose} \]
\[ \text{Proof.} \]
\[ \text{The term $E_x [D_{KL}(Pr(y|x)||Pr(y|C(x)))]$ in the bias is unchanged since the distribution of $y$ given $x$ is not impacted by $\lambda$. We next look at $E_{x,y,\lambda_i \neq 0} [D_{KL}(Pr(\hat{\lambda}_i|y, \lambda_i|, \hat{\lambda}_i \neq 0)||Pr(\hat{\lambda}_i|y, C(x), \hat{\lambda}_i \neq 0))].$ Using the approach in Lemma 7 recall that $Pr(\hat{\lambda_i}|y, \lambda_i|, \hat{\lambda}_i \neq 0) = Pr(\lambda_i|y, x, \lambda_i(x) \neq 0) \text{ when } \lambda_i(x) \neq 0, \text{ and } Pr(\lambda_i(\mathbb{N}(x))|y, \mathbb{N}(x), \lambda_i(\mathbb{N}(x)) \neq 0) \text{ when } \lambda_i(x) = 0. \text{ Therefore, by Assumption 1,}$ \]
\[ \text{We first introduce some notation. Define $\lambda_i(\mathbb{N}(x))$ has coverage on. In particular, $\mathbb{N}(x)$ is the nearest neighbor in $S$ (in the body, this is just referred to as $\mathbb{N}(x)$), so $\lambda_i(x) := \lambda_i(\mathbb{N}(x))$ for $x \in \tilde{S}_{r_i}$. Then, we decompose $\varepsilon$ based on which point in $\tilde{S}$ is $x$’s nearest neighbor:} \]
\[ \varepsilon = \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}} (\lambda_i(\mathbb{N}_{\tilde{S}}(x)) \neq y|x) = \sum_{j=1}^{n_0} \text{Pr}_{\mathbb{P}_{\lambda_i}^{\mathbb{N}}} (\mathbb{N}_{\tilde{S}}(x) = x_j) \cdot \text{Pr}_{\mathbb{P}_{\lambda_i}^{\mathbb{N}}} (\lambda_i(x_j) \neq y|\mathbb{N}_{\tilde{S}}(x) = x_j). \]  
(18)

Let $y_j$ denote the label corresponding to $x_j$, drawn from $\mathbb{P}_{\lambda_i}(\cdot|x_j)$. The probability $\text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) \neq y|\mathbb{N}_{\tilde{S}}(x) = x_j)$ 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,
\[ \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) \neq y|\mathbb{N}_{\tilde{S}}(x) = x_j) \]
\[ \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) = y_j, y_j \neq y|\mathbb{N}_{\tilde{S}}(x) = x_j) + \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) \neq y_j, y_j = y|\mathbb{N}_{\tilde{S}}(x) = x_j) \]
\[ \leq \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) = y_j, \exists(x, y) \in \hat{S}_{r_i} : \mathbb{N}_{\tilde{S}}(x) = x_j, y_j \neq y) + \text{Pr}_{\tilde{S}_{r_i} \sim \mathbb{P}_{\lambda_i}^{\mathbb{N}}}(\lambda_i(x_j) \neq y_j, \exists(x, y) \in \hat{S}_{r_i} : \mathbb{N}_{\tilde{S}}(x) = x_j, y_j = y). \]  
(19)

Next, we recall the definition of $\hat{S}_{r_i}$ and observe that $\mathbb{N}_{\tilde{S}}(x) = x_j$ 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
\[
\Pr_{\tilde{S} \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) \neq y | \mathcal{N}_S(x) = x_j) \leq \Pr_{x_j, y_j \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) = y_j, \exists (x, y) \in \mathcal{X} \setminus S : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) \\
+ \Pr_{x_j, y_j \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) \neq y_j, \exists (x, y) \in \mathcal{X} \setminus S : \rho(f(x_j), f(x)) \leq r_i, y_j = y).
\]

The first probability on the RHS can be written as \(\Pr_{x_j, y_j \sim \mathcal{P}_{\lambda_i}} (\lambda_i(x_j) = y_j | \exists (x, y) \in \mathcal{X} \setminus S : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) \leq 1 + \beta_i \frac{\lambda_i}{2} \). Therefore, putting this back into \((18)\), \(\varepsilon \leq 1 + \beta_i \frac{\lambda_i}{2} + \frac{1 - a_i}{2}\). Since \(\tilde{a}_i(r_i) = 2(1 - \varepsilon)^2 - 1\), we have our desired bound
\[
\tilde{a}_i(r_i) \geq a_i - (1 + \beta_i) \lambda_i(r_i).
\]

\[\square\]

**Theorem 2.** Suppose that data follows the model in \((1)\). The irreducible error decreases by at least the following amount when using \(\tilde{\lambda}_i\) rather than \(\lambda_i\) in Algorithm \((7)\)
\[
H(y | \lambda_i, x) - H(y | \tilde{\lambda}_i, x) \geq 2p_i(1 - p(\lambda_{-i}))^2 \cdot \tilde{a}_i(r_i)^2.
\]

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

\[
H(y | \lambda_i, x) - H(y | \tilde{\lambda}_i, x) = \mathbb{E}_{x, y, \lambda_i} \left[ - \log \Pr(y | \lambda_i, x) + \mathbb{E}_{x, y, \lambda_i} \left[ \log \Pr(y | \tilde{\lambda}_i, x) \right] \right] = \mathbb{E}_{x, y, \lambda_i} \left[ \log \frac{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)}{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)} \right].
\]

\((20)\)

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

\[
H(y | \lambda_i, x) - H(y | \tilde{\lambda}_i, x) = \mathbb{E}_{x, y, \lambda_i} \left[ \log \frac{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)}{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)} \right] = \mathbb{E}_{x, y, \lambda_i} \left[ \log \frac{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)}{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x) \Pr(y | x)} \right].
\]

\((21)\)

When extending \(\lambda_i\), there are three regions of interest in input space: \(\lambda_i(x), \tilde{\lambda}_i(x) \neq 0; \lambda_i(x) = 0, \tilde{\lambda}_i(x) \neq 0; \text{ and } \lambda_i(x) = \tilde{\lambda}_i(x) = 0\). In the first region, \(\tilde{\lambda}_i\) 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 \(\lambda_i(x) = \tilde{\lambda}_i(x) = 0\), 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 \(\tilde{\lambda}_i\). Then, \((21)\) becomes

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

\((22)\)

We can write \(\Pr(\lambda_i = 0 | x, y) \Pr(\lambda_i = 0 | x, \lambda_{-i} = x) = \frac{1}{\Pr(\lambda_i = 0 | x)}\) 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

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

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

\[
\mathbb{E}_{y=1, \lambda_{-i}, \lambda_i(x) \neq 0, \lambda_i(x) = 0} \left[ \log \frac{\Pr(\lambda_i | x, y) \Pr(\lambda_i | \lambda_{-i}, x)}{\Pr(\lambda_i | \lambda_{-i}, x)} \right] = \mathbb{E} \left[ \Pr(\lambda_i = 1 | y = 1, x, \lambda_i \neq 0) \log \frac{\Pr(\lambda_i = 1 | x, y = 1) \Pr(\lambda_i = 1 | \lambda_{-i}, x)}{\Pr(\lambda_i = 1 | \lambda_{-i}, x)} + \Pr(\lambda_i = -1 | y = 1, x, \lambda_i \neq 0) \log \frac{\Pr(\lambda_i = -1 | x, y = 1)}{\Pr(\lambda_i = -1 | \lambda_{-i}, x)} \right].
\]

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

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

where \( p(x, \lambda \neq i) \) is shorthand for \( \Pr(y = 1 | \lambda \neq i, x) \) (importantly, it does not depend on \( \tilde{\lambda}_i \)) and likewise for \( \Pr(\tilde{\lambda}_i = -1 | \lambda \neq i, x) = \alpha_i(x)(1 - p(x, \lambda \neq i)) + (1 - \alpha_i(x))p(x, \lambda \neq i) \). Our expression from (25) is now

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

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

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

\[
\frac{\alpha_i \log \alpha_i}{\alpha_i - \alpha_i p_{\lambda \neq i}} + \frac{(1 - \alpha_i) \log (1 - \alpha_i)}{1 - \alpha_i}.
\]

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

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

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

\[
\frac{\alpha_i \log \alpha_i}{\alpha_i - \alpha_i p_{\lambda \neq i}} + \frac{(1 - \alpha_i) \log (1 - \alpha_i)}{1 - \alpha_i}.
\]

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

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

We can lower bound the expression in the parentheses. Define \( g(x) = x \log \frac{x}{x + (1 - x)p} + (1 - x) \log \frac{1 - x}{(1 - x)p + (1 - p)} \) 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 \( x < 0.5 \). \( g'(x) = \frac{1-p}{x \beta + (1-x) \gamma} + \log \frac{1-z}{x \beta + (1-x) \gamma} \), 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 \( x \in [0, 1] \). \( g''(x) = \frac{(p-1)(2p-1)}{(x(1-p)+(1-x)p)^2} + \frac{1-p}{x(1-x)+(1-p)} + \frac{1-p}{(1-x)(1-p)+(1-x)p} \), 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 \( g(x) \geq 8/(1-p)^2(x-0.5)^2 \). We thus get

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

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

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

### E.1 DATASET 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).

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

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., 2018] 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 [0.6, 0.35, 0.35, 0.65] and \( s \) is set to 3. For Tennis, thresholds are set to [0.11, 0.110, 0.11, 0.85, 0.11, 0.11] and \( s \) is set to 2.

In our experiments, class balance \( \text{Pr}(y|C_j) \) 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 \( \in \{1e-3, 1e-2, 1e-1\} \), weight decay \( \in \{5e-4, 0\} \), 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 1e-1 learning rate, 5e-4 weight decay, and 0.9 momentum. For Weather, we use 1e-1 learning rate, 5e-4 weight decay, and 0.9 momentum. For Spouse, we use 1e-2 learning rate, 5e-4 weight decay, and 0 momentum. For Basketball, we use 1e-3 learning rate, 0 weight decay, and 0.9 momentum. For Commercial, we use 1e-1 learning rate, 5e-4 weight decay, and 0 momentum. For Tennis, we use 1e-1 learning rate, 5e-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

| Task     | LIGER (s) |
|----------|-----------|
| Spam     | 96.8 (2)  |
| Weather  | 95.3 (3)  |
| Spouse   | 17.0 (6)  |
| Basketball | 81.7 (2)  |
| Commercial | 93.4 (3)  |
| Tennis   | 83.4 (1)  |

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 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

$$
| \Pr(y = 1 | x) - \Pr(y = 1 | x') | = |\mathbb{E} [ 1 \{ y = 1 | x \} ] - \mathbb{E} [ 1 \{ y = 1 | x' \} ] | \\
\leq |\mathbb{E} [ 1 \{ y = 1 | x \} - 1 \{ y = 1 | x' \} ] | \\
= \mathbb{E} [ 1 \{ y \neq y' \} ] = \Pr(y \neq y')
$$
| Embedding      | F1-score |
|---------------|----------|
| Raw pixel     | 19.3     |
| RN-101        | 31.1     |
| BiT-M         | 42.5     |
| CLIP          | 69.6     |

| Prompting     | F1-score |
|---------------|----------|
| No Prompt     | 48.5     |
| Prompt at Beginning | 50.2 |
| Prompt at End | 52.2     |

![Graphs showing performance and smoothness measurements for different embeddings and conditions.](image)

Figure 3: Top: LIGER performance and smoothness measurements of CLIP, BiT-M, ResNet-101, and raw pixels as embeddings for Basketball. Bottom: LIGER performance and smoothness measurements of no prompting, prompting at beginning, and prompting at end in GPT-3 for Spouse.

by Jensen’s inequality. Therefore, we estimate $\Pr(y = y')$ on data as an upper bound on label Lipschitzness. We do this by computing the average percentage of points in some local region (defined either by a radius or by nearest neighbors) around a given point where the label is different from that of the given point.

For source Lipschitzness, the sources in practice are unimodal and hence $K_\lambda = 0$.

For coverage Lipschitzness, we note that $|\Pr(\lambda_i \neq 0|x) - \Pr(\lambda_i \neq 0|x')| \leq \Pr(\mathbb{I} \{\lambda_i(x) \neq 0\} = 0) = \mathbb{I} \{\lambda_i(x) \neq 0\}$, so we estimate this probability on data as an upper bound. This is done by computing the average percentage of points that abstain in some local region around a point that has coverage, and vice versa. We average over all sources.

Finally, for local label probabilistic Lipschitzness, we follow Definition 2. For each point in the support of $\lambda_i$, we search if there exists a nearby point within radius $r$ (or $k$-th nearest neighbor) such that this nearby point is not in the support and has a label different from that of the given point. We compute the percentage of points in the support that satisfy this property. We average over all sources.

To read $K_y, K_{\lambda, 0}, M$ from Figure 3 they can each be viewed as the slope of the linear function that upper bounds the smoothness curve. Note that for the curves that appear flat (i.e. no prompt), these constants are very large, as there is an initial sharp increase in the percentage of points with changed value.

### F.3 SYNTHETIC EXPERIMENTS

We evaluate LIGER on synthetic data to confirm our insights about 1) how generalization error for $\hat{P}_y(y|\lambda, x)$ 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 $\hat{P}_y(y|\lambda, x)$ (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 $\Pr(y, \lambda|\mathbf{x})$, 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 $\Pr(y, \mathbf{x})$ 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 $\bar{\lambda}_1$ over $\lambda_1$.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644, 2016.

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, 2019.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In Proceedings of the eleventh annual conference on Computational learning theory. ACM, 1998.

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

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, et al. On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 2020.

Salva Rühling Cachay, Benedikt Boecking, and Artur Dubrawski. End-to-end weak supervision. In Advances in Neural Information Processing Systems, 2021.

Mayee Chen, Benjamin Cohen-Wang, Stephen Mussmann, Frederic Sala, and Christopher Re. Comparing the value of labeled and unlabeled data in method-of-moments latent variable estimation. In Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning. PMLR, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

Daniel Y. Fu, Will Crichton, James Hong, Xinwei Yao, Haotian Zhang, Anh Truong, Avanika Narayan, Maneesh Agrawala, Christopher Ré, and Kayvon Fatahalian. Rekall: Specifying video events using compositions of spatiotemporal labels. arXiv preprint arXiv:1910.02993, 2019.

Daniel Y. Fu, Mayee F. Chen, Frederic Sala, Sarah M. Hooper, Kayvon Fatahalian, and Christopher Ré. Fast and three-rious: Speeding up weak supervision with triplet methods. In Proceedings of the 37th International Conference on Machine Learning, 2020.

Sonal Gupta and Christopher Manning. Improved pattern learning for bootstrapped entity extraction. In Proceedings of the Eighteenth Conference on Computational Natural Language Learning, 2014.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. arXiv preprint arXiv:1911.05722, 2019.

James Hong, Will Crichton, Haotian Zhang, Daniel Y Fu, Jacob Ritchie, Jeremy Barenholtz, Ben Hannel, Xinwei Yao, Michaela Murray, Geraldine Moriba, et al. Analysis of faces in a decade of us cable tv news. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In International Conference on Machine Learning. PMLR, 2019.
Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5070–5079, 2019.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2019.

Ashish Khetan, Zachary C. Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. In *International Conference on Learning Representations*, 2018.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Percy Liang, Michael I Jordan, and Dan Klein. Learning from measurements in exponential families. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009.

Gideon S Mann and Andrew McCallum. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of machine learning research*, 2010.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*, 2022.

Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the 44th International Conference on Very Large Data Bases (VLDB)*, Rio de Janeiro, Brazil, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

Esteban Safranchik, Shiyi Luo, and Stephen H Bach. Weakly supervised sequence tagging from noisy rules. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Frederic Sala, Paroma Varma, Jason Fries, Daniel Y. Fu, Shiori Sagawa, Saelig Khattar, Ashwini Ramamoorthy, Ke Xiao, Kayvon Fatahalian, James Priest, and Christopher Ré. Multi-resolution weak supervision for sequential data. In *Advances in Neural Information Processing Systems 32*, 2019.

Ying Sheng, Nguyen Ha Vo, James B. Wendt, Sandeep Tata, and Marc Najork. Migrating a privacy-safe information extraction system to a software 2.0 design. In *Proceedings of the 10th Annual Conference on Innovative Data Systems Research*, 2020.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012.
Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. Want to reduce labeling cost? GPT-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4195–4205, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.354. URL https://aclanthology.org/2021.findings-emnlp.354

Eric Zhan, Stephan Zheng, Yisong Yue, Long Sha, and Patrick Lucey. Generating multi-agent trajectories using programmatic weak supervision. In *7th International Conference on Learning Representations, ICLR*, 2019.