Shoring Up the Foundations: Fusing Model Embeddings and Weak Supervision

Abstract

Foundation models offer an exciting new paradigm for constructing models with out-of-the-box embeddings and a few labeled examples. However, it is not clear how to best apply foundation models without labeled data. A potential approach is to fuse foundation models with weak supervision frameworks, which use weak label sources—pretrained models, heuristics, crowd-workers—to construct pseudolabels. The challenge is building a combination that best exploits the signal available in both foundation models and weak sources. We propose LIGER, a combination that uses foundation model embeddings to improve two crucial elements of existing weak supervision techniques. First, we produce finer estimates of weak source quality by partitioning the embedding space and learning per-part source accuracies. Second, we improve source coverage by extending source votes in embedding space. Despite the black-box nature of foundation models, we prove results characterizing how our approach improves performance and show that lift scales with the smoothness of label distributions in embedding space. On six benchmark NLP and video tasks, LIGER outperforms vanilla weak supervision by 14.1 points, weakly-supervised kNN and adapters by 11.8 points, and kNN and adapters supervised by traditional hand labels by 7.2 points.

1 INTRODUCTION

Foundation models—large pretrained models such as GPT-3, BERT, CLIP, and DALL-E [Brown et al., 2020, Devlin et al., 2019, Radford et al., 2021, Ramesh et al., 2021]—offer powerful representations that can be used in a broad array of settings [Bommasani et al., 2021]. These models have achieved state-of-the-art performance in many tasks. However, it remains unclear how to best apply foundation models in situations where users lack access to any labeled data, but do have some weak signals. These are the cases where another class of techniques—weak supervision [Ratner et al., 2018, Fu et al., 2020]—shines.

The broad success of foundation models (FMs) suggests that fusing them with weak supervision may offer substantial benefits. Intuitively, the signals present in both can be used to replace large amounts of hand-labeled data in supervised learning. These signals are complementary. Foundation models are trained on huge amounts of data and thus offer powerful general-purpose embeddings. Weak supervision frameworks rely on multiple weak sources of signal that can be synthesized into pseudolabels for downstream training. These weak sources typically express specialized domain expertise. The fusion may enable each component to be improved: FM embeddings can be used without labeled data, while weak sources may be extended to be more general-purpose.

Our goal is to combine these complementary signals to address two challenges in existing approaches to weak supervision. The first challenge is performing fine-grained estimation of source quality. Current weak supervision approaches typically coarsely model source quality by assuming error distributions are uniform over unlabeled points [Ratner et al., 2019, Fu et al., 2020], but source quality may vary across points in actuality. The second challenge is producing votes on points where sources abstain. Weak sources often abstain, so that current approaches suffer from low coverage and have many points lacking any signal. We seek to exploit the powerful embeddings from FMs—and the geometry induced by them—to address these challenges.

We propose LIGER, a new weak supervision approach based on the notion of *local* quality of weak sources in the FM embedding space (named after a well-known fusion of powerful animals). We introduce an efficient algorithm that partitions the embedding space and learns per-part local

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

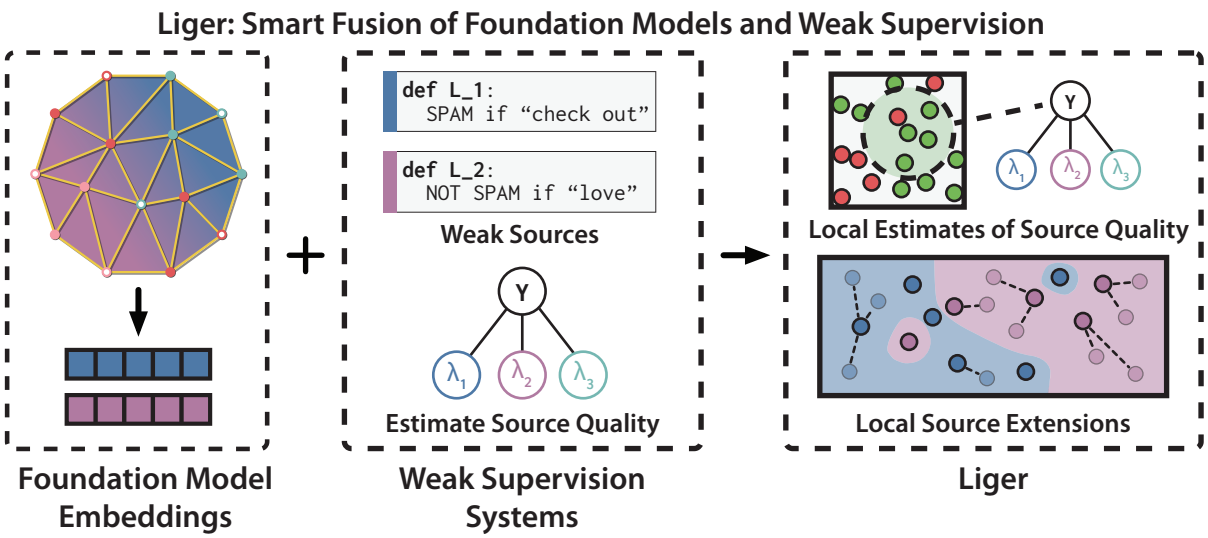


Figure 1: LIGER fuses representations from foundation models (left) with weak supervision (middle) by exploiting local smoothness of the representations (right). LIGER uses the representations to a) produce more accurate local estimates of weak source quality (right, top), and b) to locally extend weak sources, improving their coverage (right, bottom).

source accuracies. LIGER also extends weak sources into nearby regions of the embedding space that they previously abstained on. Despite the fact that FMs are typically black-box, our localized approach exploits a simple measurable notion of their signal: the smoothness of the label distribution in the embedding space. When label values do not vary significantly over an embedding region, local source accuracies can be estimated well, and local source extensions maintain their accuracy. We introduce generalization error bounds that individually characterize the impact of partitioning and extending. These error bounds scale in the embedding smoothness and involve a bias-variance tradeoff in the number of partitions and the radius for extensions, suggesting that careful insertion of the FM's signal into our approach is necessary.

We evaluate LIGER on six benchmark NLP and video weak supervision tasks, fusing weak sources with GPT-3 embeddings [Brown et al., 2020, Neelakantan et al., 2022] for the NLP tasks, and with image embeddings from CLIP [Radford et al., 2021] for the video tasks. LIGER outperforms simple kNN or adapter-based fusions of weak supervision and FMs by 11.8 points, and traditional weak supervision on its own by 14.1 points. We also compare LIGER against two strong baselines for traditional supervision of FM: kNN and Adapters [Houlsby et al., 2019]. LIGER outperforms these methods by 7.2 points. Lift scales with embedding smoothness. We measure the smoothness of CLIP embeddings against BiT-M [Kolesnikov et al., 2020] and ResNet-101 embeddings pretrained on ImageNet [Russakovsky et al., 2015]. We find that CLIP embeddings are smoothest and result in the best performance. Similarly, we find that using the right prompt for GPT-3 has a strong effect on smoothness and performance on a relation extraction task.

In summary, we contribute:

- LIGER, a new approach for fusing foundation models with weak supervision by exploiting local smoothness of labels and weak sources in embedding space.
- Finite-sample generalization error bounds of our algorithm that scale in this smoothness.
- Evaluation of LIGER on six benchmark NLP and video weak supervision tasks, where LIGER outperforms simple fusions of foundation models and weak supervison, as well as either on their own.

2 BACKGROUND

We describe the problem setting for weak supervision. We introduce two general challenges in weak supervision that our approach using foundation model embeddings can mitigate. We then propose a model and explain its two stages—source quality estimation and pseudolabel inference. While our approach is compatible with multiple basic weak supervision techniques, we focus on the approach from Fu et al. [2020]. We give a brief background on this estimation technique.

2.1 PROBLEM SETUP

Our goal is to predict label $y \in \mathcal{Y} = \{-1, +1\}$ from datapoints $x \in \mathcal{X}$. If we had access to pairs (x, y), we could train a supervised model. However, we do not have access to any samples of y; instead, we observe m *weak sources* $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_m\}$, each voting or abstaining on each point x via a probabilistic *labeling function* $\lambda_j : \mathcal{X} \to \mathcal{Y} \cup \{0\}$

for all $j \in [m]$. We refer to $\lambda_j(x) = 0$ as an abstain, which occurs when a source is uncertain on a point.

We also have access to FM embeddings. These embeddings are the outputs of a mapping $f : \mathcal{X} \to \mathcal{Z}$ from input space to an embedding space $\mathcal{Z}$ equipped with metric $\rho : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$. This mapping is fixed and obtained from an off-the-shelf model. Overall, we have an unlabeled dataset $\mathcal{D}\{x_i\}_{i=1}^n$ of n i.i.d. points, as well as access to m weak sources and the embedding map f.

Given an input x and $\boldsymbol{\lambda}(x)$, we aim to learn a *label model* that predicts y by estimating $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ (we drop the x in $\boldsymbol{\lambda}(x)$ when obvious). The goal of the label model is to combine sources based on their individual accuracies (i.e. λ_i's rate of agreement with y) by upweighting high-quality sources and downweighting low-quality ones. The resulting pseudolabels given by $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ can be used to train a downstream supervised model or just directly as predictions. The latter case is often ideal, since users need not train an additional model. We focus on this setting.

Two Challenges and Opportunities. Next, we describe two challenges common to weak supervision techniques. Fusing weak supervision with FM embeddings presents opportunities to mitigate these challenges.

- **Coarse Accuracy Modeling.** The most common assumption in weak supervision is to model $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ as $\hat{\Pr}(y|\boldsymbol{\lambda})$. That is, conditioned on the weak sources, the true label is viewed as independent of the features, so only one set of accuracies is learned over the data. Removing this assumption is desirable, since the feature space may have information about the task not captured fully by weak sources. However, naively attempting to model per-point accuracies leads to noisy estimation.

- **Low Coverage.** Weak sources frequently abstain, leading to low coverage—a situation where much of the dataset has no votes. A simple mitigation is to extend votes from nearby non-abstaining points, but this is risky if the notion of distance is not well-aligned with the label distribution.

An intuitive way to tackle these two challenges is to operate *locally*. Suppose the source votes and the true label satisfy some level of smoothness such that within some local region of the feature space, they have a low probability of changing values. We can then model accuracies specific to such local regions and can extend source votes to points they abstain on within the regions. However, raw image and text features may lack signal and not offer sufficient smoothness to permit operating locally. By acting on the embedding space, the desired smoothness property is improved (see Figure 2). We can thus obtain finer-grained accuracy estimation and improved coverage by using FM embeddings to model local accuracies and extend locally.

Next, we make these notions concrete by presenting the explicit model for $\Pr(y, \boldsymbol{\lambda}|x)$.

2.2 LABEL MODEL

We model $\Pr(y, \boldsymbol{\lambda}|x)$ as a probabilistic graphical model. Our use of this model has two steps. First, in training, we must estimate the accuracy parameters of $\Pr(y, \boldsymbol{\lambda}|x)$ without access to y. Then, at inference, we compute $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$.

Let the graphical model be based on $G = (V, E)$, where $V = y \cup \boldsymbol{\lambda}$ and E consists of edges from y to each λ_j. For simplicity, we assume there are no dependencies between the weak sources, although the dependencies can be learned and handled in the label model [Varma et al., 2019, Ratner et al., 2019]. Our choice of base estimator [Fu et al., 2020] can handle dependencies, so our approach applies to this case as well. We model the data distribution as

$$\Pr(y, \boldsymbol{\lambda}|x) = \frac{1}{Z} \exp\Big(\underbrace{\theta_y(x) y}_{\text{Class Balance}} + \sum_{i=1}^{m} \underbrace{\theta_i(x) \lambda_i y}_{\text{Source Accuracy}} + \sum_{i=1}^{m} \underbrace{\theta_{i,0}(x) \mathbb{1}\{\lambda_i = 0\}}_{\text{Abstain Rate}} \Big) \quad (1)$$

with partition function Z and a set of canonical parameters per x, $\Theta(x) = \{\theta_y(x), \theta_i(x), \theta_{i,0}(x) \ \forall i \in [m]\}$. An important property above is that $\lambda_i \perp\!\!\!\perp \lambda_j | y, x \ \forall i, j \in [m]$.

The model concretely portrays the two challenges in weak supervision. First, canonical parameters $\Theta(x)$ that are a function of the input can capture varying accuracy across the data. This is less strict than prior formulations that model the marginal $\Pr(y, \boldsymbol{\lambda})$ with one set of canonical parameters without considering input data. However, estimating $\Theta(x)$ is challenging; parametric approaches are often not realistic, and nonparametric approaches rely on regularity conditions of $\mathcal{X}$ that may not hold in practice. Second, when $\lambda_i(x) = 0$, the weak source provides no information on x at inference and is thus typically ignored on that point in previous approaches. This is reflected in the graphical model by Lemma 2, by which $\Pr(y|\lambda_i = 0, \boldsymbol{\lambda} \backslash \lambda_i, x) = \Pr(y|\boldsymbol{\lambda} \backslash \lambda_i, x)$. In fact, the weak sources provide no direct signal on x when $\boldsymbol{\lambda}(x) = \vec{0}$.

Pseudolabel Inference. To perform inference, we compute $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ for some $x \in \mathcal{X}$. This is done via Bayes' rule and the conditional independence of weak sources: $\Pr(y|\boldsymbol{\lambda}, x) = \prod_{i=1}^m \Pr(\lambda_i|y, x) \Pr(y|x) / \Pr(\boldsymbol{\lambda}|x)$. The latent parameter of interest in this decomposition is $\Pr(\lambda_i|y, x)$, which corresponds to the accuracy of λ_i.

Source Parameter Estimation. Previous approaches have considered how to estimate $\Pr(\lambda_i|y)$ in a model of $\Pr(\lambda, y)$ via the *triplet method* [Fu et al., 2020], using conditional independence properties. For our setting, (1) tells us that $\lambda_i y \perp\!\!\!\perp \lambda_j y | \lambda_i \wedge \lambda_j \neq 0, x$ for any $i \neq j$ (Lemma 3). As a result, $\mathbb{E}[\lambda_i y | \lambda_i \neq 0, x] \times \mathbb{E}[\lambda_j y | \lambda_j \neq 0, x] = \mathbb{E}\left[\lambda_i \lambda_j y^2 | \lambda_i \wedge \lambda_j \neq 0, x\right] = \mathbb{E}[\lambda_i \lambda_j | \lambda_i \wedge \lambda_j \neq 0, x]$, which consists of observable variables. Define $a_i(x) = \mathbb{E}[\lambda_i y | \lambda_i \neq 0, x]$ as the *accuracy* of λ_i on x. If we intro-

Algorithm 1 LIGER

Input: Dataset $\mathcal{D} = \{x_i\}_{i=1}^n$, weak sources $\boldsymbol{\lambda}$, embedding mapping f and metric ρ, threshold radii $r_1, \dots r_m$, partition $\mathcal{C}$ and class balances $\Pr(y|C_j)$ for $j \in [s]$.
Returns: Label model $\hat{\Pr}(y|\bar{\boldsymbol{\lambda}}, x)$.
for $\lambda_i \in \boldsymbol{\lambda}$ **do**
  Construct extended source $\bar{\lambda}_i$ using r_i, f, ρ as in (3).
end for
for $C_j \in \mathcal{C}$ **do**
  for $\bar{\lambda}_i \in \bar{\boldsymbol{\lambda}}$ **do**
    Compute accuracy $\hat{a}_i(C_j)$ using Algorithm 2 on $\bar{\lambda}_i$ over C_j and coverage $\hat{\Pr}(\bar{\lambda}_i \neq 0|C_j)$ on $\mathcal{D}$.
    Set $\hat{\Pr}(\bar{\lambda}_i|y, C_j) := \frac{1+\text{sgn}(\bar{\lambda}_i y)\hat{a}_i(C_j)}{2}\hat{\Pr}(\bar{\lambda}_i \neq 0|C_j)$.
  end for
  Compute $\hat{\Pr}(\bar{\boldsymbol{\lambda}}|C_j)$ on $\mathcal{D}$.
end for
return For test point $x \in \mathcal{X}$, compute $\hat{\Pr}(y|\bar{\boldsymbol{\lambda}}, x) = \hat{\Pr}(y|\bar{\boldsymbol{\lambda}}, C(x)) = \frac{\prod_{i=1}^m \hat{\Pr}(\bar{\lambda}_i|y, C(x)) \Pr(y|C(x))}{\hat{\Pr}(\bar{\boldsymbol{\lambda}}|C(x))}$.

duce a third λ_k, we can generate a system of equations over $a_i(x), a_j(x), a_k(x)$ in terms of the conditional expected products of pairs of $\lambda_i, \lambda_j, \lambda_k$. As a result,

$$|a_i(x)| := \sqrt{\left|\frac{\mathbb{E}\left[\lambda_i\lambda_j|\lambda_i \wedge \lambda_j \neq 0, x\right]\mathbb{E}\left[\lambda_i\lambda_k|\lambda_i \wedge \lambda_k \neq 0, x\right]}{\mathbb{E}\left[\lambda_j\lambda_k|\lambda_j \wedge \lambda_k \neq 0, x\right]}\right|}, \tag{2}$$

and likewise for $\hat{a}_j(x), \hat{a}_k(x)$. A full treatment is in Fu et al. [2020]. (1) allows us to write $\Pr(\lambda_i|y, x) = \frac{1+\text{sgn}(\lambda_i y)a_i(x)}{2} \times \Pr(\lambda_i \neq 0|x)$ (Lemmas 2 and 4), so the desired probability estimate is just a linear transformation of $a_i(x)$ scaled by λ_i's coverage.

3 FUSION ALGORITHM

We are ready to present LIGER, our approach to fusing foundation model embeddings and weak supervision. We explain the two components: first, how to compute conditional estimates of the label model parameters over local regions of the partitioned embedding space for finer-grained accuracy estimation; second, how to extend weak sources via a kNN-like augmentation in the embedding space, improving their coverage and hence the signal available at inference. The full approach is shown in Algorithm 1.

Local Parameter Estimation Our first task is to compute the label model's local parameters. Based on (2), the quantities to estimate are of the form $\mathbb{E}\left[\lambda_i\lambda_j|\lambda_i \wedge \lambda_j \neq 0, x\right]$, $\Pr(\lambda_i \neq 0|x)$, $\Pr(\boldsymbol{\lambda}|x)$, $\Pr(y|x)$. These conditional statistics can be estimated using nonparametric approaches such as the Nadaraya-Watson estimator, but they require $\mathcal{O}(n)$ computations per point at inference.

To reduce the complexity, instead of estimating parameters per point, we partition the embedding space and compute *per-part* statistics. Intuitively, this choice exploits smoothness. If label distributions are smooth, i.e., they do not vary greatly within a local region, it is sufficient to estimate per-point statistics using a part given that parts are not too large. Controlling the size of the partition is thus important in determining how well we can approximate per-point statistics.

Concretely, partition $\mathcal{Z}$ into s subsets $\mathcal{C} = \{C_1, \dots, C_s\}$ of equal size $n' = \frac{n}{s}$ (we use K-means clustering with $K = s$ in practice). Denote $C(x)$ as the subset $f(x)$ belongs to. Instead of estimating statistics and performing inference conditioned on x, we condition on $C(x)$, producing s sets of parameters overall. We estimate $\mathbb{E}\left[\lambda_i\lambda_j|\lambda_i \wedge \lambda_j \neq 0, C(x)\right]$, yielding a local accuracy estimate $\hat{a}_i(C(x))$ formalized in Algorithm 2, as well as $\Pr(\lambda_i \neq 0|C(x))$, $\Pr(\boldsymbol{\lambda}|C(x))$, $\Pr(y|C(x))$. Then, we use $\hat{\Pr}(y|\boldsymbol{\lambda}, x) = \hat{\Pr}(y|\boldsymbol{\lambda}, C(x))$ as our label model prediction on x. These estimates are done over the subsets; for instance, $\Pr(\boldsymbol{\lambda}|C(x)) \approx \frac{1}{n'}\sum_{x' \in C(x)} \mathbb{1}\left\{\boldsymbol{\lambda}(x')=\boldsymbol{\lambda}\right\}$. We assume that class balance on subsets, $\Pr(y|C(x))$, are known although there are several techniques that can be used to estimate these [Ratner et al., 2019], or they can be treated as hyperparameters.

Weak Source Extension Next, we improve the model of $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ by increasing source coverage. Let $\bar{\lambda}_i$ be an extended labeling function with corresponding threshold radius $r_i > 0$ for $i \in [m]$. The extension works as follows. For any x, let $\text{NN}(x) = \text{argmin}_{x' \in \mathcal{D}:\lambda_i(x) \neq 0} \rho(f(x), f(x'))$ be the nearest neighbor of x in embedding space from $\mathcal{D}$ such that λ_i has coverage on it. $\bar{\lambda}_i$ weakly labels points within r_i of λ_i's support on $\mathcal{D}$ using x'. Formally,

$$\bar{\lambda}_i(x) := \begin{cases} \lambda_i(x) & \lambda_i(x) \neq 0 \\ \lambda_i(\text{NN}(x)) & \rho(f(x), f(\text{NN}(x))) \leq r_i \\ 0 & \text{o.w.} \end{cases}. \tag{3}$$

We can view $\bar{\lambda}_i$ as an augmentation on λ_i using $\mathcal{D}$ and f. We thus perform parameter estimation and inference using $\bar{\boldsymbol{\lambda}}$ instead of $\boldsymbol{\lambda}$, namely learning $\Pr(y|\bar{\boldsymbol{\lambda}}, C(x))$.

There are two advantages to using extended sources. First, extended sources improve sampling error, since expressions like $\mathbb{E}\left[\lambda_i\lambda_j|\lambda_i \wedge \lambda_j \neq 0, x\right]$ are estimated over more data in $\mathcal{D}$. Second, $\bar{\lambda}_i$ provides signal at inference on points that λ_i previously abstains on. However, the quality of this signal greatly depends on r_i. If λ_i is overextended and the embedding space is not sufficiently smooth, points far away from λ_i's support may receive incorrect extended source votes, suggesting that careful choice of r_i is needed

Our approach combines the two components discussed—partitioning the embedding space and extending sources—to output predictions $\hat{\Pr}(y|\bar{\boldsymbol{\lambda}}, C(x))$ as in Algorithm 1.

4 THEORETICAL ANALYSIS

Now we turn to analyzing Algorithm 1. Our goal is to understand how performance depends on the key parameters: fineness of the partition, radii of the extensions used to improve coverage, and smoothness of the embedding space.

We begin with a result on the generalization error of the label model $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$, which relies on s to control the granularity of the estimated parameters. Then, we discuss the improvement from using $\bar{\boldsymbol{\lambda}}$ instead of $\boldsymbol{\lambda}$. We first bound the local accuracy of an extended source in a region it previously abstains, and then we show that as long as this local accuracy is better than random, we can further reduce the generalization error. The former result presents a bias-variance tradeoff depending on s, while the latter has a tradeoff dependent on the threshold radius r_i. In both cases, s and r_i must be carefully set based on the signal in the FM embeddings, namely the smoothness of label distributions in the FM embedding space, in order to optimize performance. We provide proofs in Appendix D, synthetic experiments supporting our findings in Appendix F.3, and smoothness measurements on real data in Section 5.2.

Define the generalization error of the label model using weak sources $\boldsymbol{\lambda}$ as the expected cross-entropy loss, $L(\boldsymbol{\lambda}) = \mathbb{E}_{\mathcal{D},x,y,\boldsymbol{\lambda}}[-\log \hat{\Pr}(y|\boldsymbol{\lambda}, x)]$.

4.1 LABEL MODEL GENERALIZATION ERROR

We bound the generalization error $L(\boldsymbol{\lambda})$ of the label model using the unextended, initial weak sources. The key quantity in this analysis is embedding smoothness:

Definition 1. *The distributions* $\Pr(y|x)$ *and* $\Pr(\lambda_i|y, x)$ *are Lipschitz-smooth on the metric space* $(\mathcal{Z}, \rho)$ *with constants* $K_y, K_\lambda, K_{\lambda,0} > 0$ *if for all* $i \in [m]$,

$$
\begin{aligned}
&|\Pr(y = 1|x) - \Pr(y = 1|x')| \leq K_y \rho(f(x), f(x')),\\
&|\Pr(\lambda_i = 1|y, \lambda_i \neq 0, x) - \Pr(\lambda_i = 1|y, \lambda_i \neq 0, x')|\\
&\quad \leq K_\lambda \rho(f(x), f(x')),\\
&|\Pr(\lambda_i \neq 0|x) - \Pr(\lambda_i \neq 0|x')| \leq K_{\lambda,0} \rho(f(x), f(x')),
\end{aligned}
$$

We refer to these three properties as label, source, and coverage Lipschitzness, respectively.

In words, if the constants $K_y, K_\lambda, K_{\lambda,0}$ are small, the value of y and the way each source votes (or doesn't) do not vary significantly over a local region of the embedding space.

We define some additional quantities. Set $p_{\min} = \min_i \mathbb{E}_x [\Pr(\lambda_i \neq 0|C(x))]$ to be the minimum average source coverage over the subsets. Assume that $a_i(C_j) > 0$ for all λ_i and C_j, meaning that the average source accuracy on a subset is better than random. Then, define $a_{\max} = \max_{i,j} a_i(C_j)$, and $b_{\min} = \min_{i,j,k}\{\mathbb{E}[\lambda_i\lambda_k|\lambda_i \wedge \lambda_k \neq 0, C_j], \hat{\mathbb{E}}[\lambda_i\lambda_k|\lambda_i \wedge \lambda_k \neq 0, C_j]\}$ as the minimum rate of agreement between sources over subsets, where $\hat{\mathbb{E}}$ denotes the empirical estimate. Define $d_{C_j} = \max_{f(x),f(x') \in C_j} \rho(f(x), f(x'))$ as the diameter of C_j and $d_{\mathcal{C}} = \mathbb{E}_x \left[d_{C(x)}\right]$ as its average.

Theorem 1. *Suppose that data* $x, y, \boldsymbol{\lambda}$ *follows the model in* (1). *The generalization error of the label model* $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ *in Algorithm 1 when* $r_i = 0\ \forall i$ *can be decomposed into* $L(\boldsymbol{\lambda})$*=Bias + Variance + Irreducible Error +* $o(1/n)$*, where*

$$
\begin{aligned}
&\textit{Bias} \leq 2d_{\mathcal{C}}(K_y + mK_\lambda + mK_{\lambda,0}),\\
&\textit{Variance} \leq \frac{ms}{2n}\left(\frac{3(1-b_{\min}^2)}{4b_{\min}^2(1-a_{\max})^2}\left(\frac{1}{b_{\min}^4} + \frac{2}{b_{\min}^2}\right)\right.\\
&\qquad\qquad \left. + \frac{1-p_{\min}}{p_{\min}}\right),\\
&\textit{Irreducible Error} = H(y|\boldsymbol{\lambda}, x),
\end{aligned}
$$

where $H(y|\boldsymbol{\lambda}, x)$ *denotes conditional entropy.*

We discuss each term of this bound.

- The bias comes from the partition $\mathcal{C}$, since conditional statistics on $C(x)$ are not equivalent to those on x. When the embedding space is smooth with small $K_y, K_\lambda, K_{\lambda,0}$, the bias is low. Note that making the subset diameter $d_C \to 0$ makes the bias go to zero.
- The variance comes from sampling error in Algorithm 2 and $\hat{\Pr}(\lambda_i \neq 0|C_j)$. This quantity scales in $\mathcal{O}(1/n)$ and depends on coverage, accuracy, and agreement among weak sources.
- The irreducible error depends on quality of $\boldsymbol{\lambda}$. If knowledge of $\boldsymbol{\lambda}$ significantly reduces uncertainty in y, i.e., the sources contain lots of signal, this quantity is low. On the other hand, $H(y|\boldsymbol{\lambda}, x)$ is maximized when $\boldsymbol{\lambda} \perp\!\!\!\perp y|x$, i.e. no signal about y in $\boldsymbol{\lambda}$.

Our result reveals a bias-variance tradeoff dependent on the number of parts s. As s increases, subset diameter $d_{\mathcal{C}}$ decreases, resulting in lower bias because the subset parameters estimated will be closer in true value to those conditional on x. The variance increases in s because there are fewer points per subset for estimation. The $s = 1$ case, which incurs a large bias, is algorithmically equivalent to the approach in Fu et al. [2020]. Such approaches thus suffer from model misspecification in our setting—and likely in most practical cases—as they assume uniform quality per source.

4.2 IMPROVEMENT FROM EXTENSIONS

Suppose that $x, y, \bar{\boldsymbol{\lambda}}$ follows (1). When we use $\bar{\boldsymbol{\lambda}}$ rather than $\boldsymbol{\lambda}$ (i.e. $r_i \neq 0$), there are several changes to the decomposition in Theorem 1:

- The bias is now bounded by $2d_{\mathcal{C}}K_y + 2m(d_{\mathcal{C}} + 2\max_i r_i)(K_\lambda + K_{\lambda,0})$ (see Lemma 8). We must consider when $\text{NN}(x)$ is not in $C(x)$, essentially resulting in a wider subset diameter.
- The variance is still $\mathcal{O}(1/n)$, but multiplicative factors change. For instance, $p_{\min}$ increases due to improved coverage, decreasing the variance.
- The irreducible error is now $H(y|\bar{\boldsymbol{\lambda}}, x)$.

We analyze $H(y|\bar{\boldsymbol{\lambda}}, x)$ in this section. $\bar{\lambda}_i$ provides more signal than λ_i at inference on points where $\lambda_i(x) = 0$, but the signal about y's value may be incorrect. Extending λ_i using too large of r_i could yield incorrect source votes, resulting in lower accuracy of the extended weak source.

We first present a result on how to control the extended source's accuracy. Define $a_i = \mathbb{E}\left[\lambda_i y | \lambda_i \neq 0\right]$ as the average accuracy of λ_i, and $\bar{a}_i(r_i) = \mathbb{E}\left[\bar{\lambda}_i y | \bar{\lambda}_i \neq 0, \lambda_i = 0\right]$ as $\bar{\lambda}_i$'s average accuracy on the extended region. We also need a notion of smoothness of y between the original support and the extended region. We define a local notion of *probabilistic Lipschitzness* (PL), originally introduced in Urner and Ben-David [2013].

Definition 2 (Probabilistic Lipschitzness)**.** *Define $\mathcal{P}_{\lambda_i} = \Pr_{x,y}(\cdot|\lambda_i \neq 0)$ to be the distribution of (x, y) over the support of λ_i, and let $\mathcal{P}_{\lambda_i,x}$ be its marginal distribution on x. Then $\mathcal{P}_{\lambda_i}$ is M-probabilistically Lipschitz for an increasing function $M : \mathbb{R}^+ \to [0, 1]$ if for any $r > 0$,*

$$\Pr_{x,y\sim\mathcal{P}_{\lambda_i}}(\exists(x', y') \in \mathcal{X}\backslash supp(\mathcal{P}_{\lambda_i,x}) \times \mathcal{Y} : \rho(f(x), f(x')) \leq r, y' \neq y) \leq M(r).$$

We refer to this property as local label PL.

In words, the probability that there is a point outside of the support of λ_i but within r of $(x, y) \sim \mathcal{P}_{\lambda_i}$ with a different label from y is bounded by an increasing function of r. We also define $\beta_i = \mathbb{E}[\lambda_i y | \lambda_i \neq 0, \exists(x', y') : \lambda_i(x') = 0, \rho(f(x), f(x')) \leq r_i, y' = y]$ as λ_i's accuracy over a region close to where λ_i is extended and y changes value.

With this definition, we show that:

Lemma 1. *The average accuracy of $\bar{\lambda}_i$ on the extended region is at least $\bar{a}_i(r_i) \geq a_i - (1 + \beta_i)M(r_i)$.*

Our result provides local accuracy guarantees on $\bar{\lambda}_i$ as a function of the original λ_i's accuracy, the probabilistic Lipschitzness of the embedding space, and the r_i the user sets. Extending a source with higher original accuracy will yield stronger accuracy guarantees in the extended region. On the other hand, if $M(r_i)$ is too large due to improper r_i or lack of smoothness, the true label is more likely to change value, and hence accuracy in the extended region worsens.

Next, to analyze the improvement in irreducible error, we suppose we extend just one weak source λ_i by r_i and keep $\lambda_{-i} := \boldsymbol{\lambda}\backslash\lambda_i$ unextended. Define $p_i = \Pr(\bar{\lambda}_i \neq 0, \lambda_i = 0)$ as the proportion of the region where $\bar{\lambda}_i$ is extended and $p(\lambda_{-i}) = \mathbb{E}_{y',\lambda_{-i},\bar{\lambda}_i\neq 0,\lambda_i=0}\left[\Pr(y = y'|\lambda_{-i}, x)\right]$ as the label model's probability of outputting the correct label in the extension region when only using λ_{-i}.

Theorem 2. *Suppose that data follows the model in* (1)*. The irreducible error decreases by the following amount when using $\bar{\lambda}_i$ rather than λ_i in Algorithm 1:*

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

Lift increases with probability mass p_i on the extended region since more of the data is impacted by $\bar{\lambda}_i$. Lift is not as significant if $p(\lambda_{-i})$ is large because the other weak sources already are providing sufficient signal for y. Most importantly, lift scales with how far $\bar{a}_i(r_i)$ is from 0 (random voting). This highlights a tradeoff in r_i: as r_i increases, p_i increases but the lower bound on $\bar{a}_i(r_i)$ from Lemma 1 decreases. This shows that threshold radii must be selected carefully; if the embedding space has strong probabilistic Lipschitzness (i.e. small M) or the original weak source has high accuracy, then the source can be extended further while providing lift. However, overextension of the source can yield low local accuracy and thus less lift.

Our results demonstrate that s and r_i control the label model's performance, and setting these terms depends on how smooth label distributions are in the embedding space.

5 EXPERIMENTS

This section evaluates the following claims about LIGER:

- **Performance (Section 5.1):** LIGER outperforms vanilla weak supervision, as well as baseline approaches for using foundation models directly, either with traditional weak supervision or hand supervision.
- **Smoothness (Section 5.2):** Lift is correlated with the smoothness of the label distribution in the representation space. We measure smoothness and performance of CLIP against three other embedding methods on a video task, and measure three prompting strategies for an NLP task.
- **Ablations (Section 5.3):** Both components of LIGER—partitioning the representation space and extending labeling function votes—are important for performance.

Datasets We evaluate LIGER on six benchmark NLP and video tasks used to evaluate previous weak supervision methods [Fu et al., 2020, Zhang et al., 2021]. In NLP, **Spam** identifies spam YouTube comments [Alberto et al., 2015]; **Weather** identifies the sentiment of weather-related tweets [Cro]; **Spouse** identifies spouse relationships in newspaper articles [Corney et al., 2016]. In video, **Commercial**

	Task	Weak Sources Only					Dev Labels Available		
		WS-kNN	WS-Adapter	WS-LM	LIGER	ΔCoverage	kNN	Adapter	LIGER-Adapter
NLP	Spam	72.8	92.3	83.6	**95.0**	+45.5	91.2	94.4	**95.4**
NLP	Weather	62.0	86.0	78.0	**98.0**	+90.2	92.0	90.0	**96.8**
NLP	Spouse	16.9	17.1	47.0	**52.2**	+12.1	21.6	15.7	**49.6**
Video	Basketball	33.3	48.9	27.9	**69.6**	+8.3	64.4	79.3	**79.5**
Video	Commercial	84.7	92.8	88.4	**93.5**	+18.8	92.0	93.0	**93.2**
Video	Tennis	83.0	**83.8**	82.0	83.3	+32.5	73.2	83.1	**84.0**

Table 1: Left: LIGER performance compared to baselines that only have access to weak labels, as well as the change in coverage from traditional weak supervision. Right: LIGER-Adapter compared to baselines that have access to dev labels available. Scores are F1 except for Spam and Weather (accuracy); best score in bold in each setting.

identifies commercial segments in TV news [Hong et al., 2021, Fu et al., 2019]; **Tennis** identifies rallies in tennis segments; **Basketball** identifies basketball videos in a subset of ActivityNet [Caba Heilbron et al., 2015]. Each dataset consists of a large unlabeled training set, a smaller hand-labeled *development set* (train/dev split sizes from 187/50 points to 64,130/9,479 points), and a held-out test set. We use the unlabeled training set to train label models and use the development set for a) training of traditional supervision baselines, and b) hyperparameter tuning of the label models, including s and r_i.

Pre-trained embeddings For the NLP datasets, we use pre-trained GPT-3 [Brown et al., 2020] embeddings from OpenAI's Ada model for label extension. For **Spam** and **Weather**, we simply embed the text directly. For **Spouse**, we add a prompt "Are [person 1] and [person 2] spouses?" after the end of the sentence. We discuss further prompting strategies in Section 5.2. For video datasets, we use image embeddings from CLIP [Radford et al., 2021] over individual frames of the videos.

5.1 PERFORMANCE

We compare LIGER against baseline approaches for fusing foundation models with weak supervision, as well as against using either on their own. We split our evaluation into two parts: methods that only have access to weak sources, and methods that additionally have access to the dev set.

Weak Sources Only We compare the performance of LIGER against vanilla weak supervision (WS-LM) [Fu et al., 2020], weakly-supervised kNN (WS-kNN), and weakly-supervised adapters (WS-Adapter). In the latter two methods, we use the predictions from WS-LM to generate pseudolabels for the train set. We report results on a single layer for our adapters (MLP in Appendix F), and we only use adapters on the final embeddings since we do not have access to full foundation model.

Table 1 (left) shows the results, as well as statistics on the change in coverage between LIGER and WS-LM. LIGER outperforms WS-LM due to better coverage (33.2 points on average). As a result, LIGER also outperforms both of the baseline approaches for fusing foundation models with weak supervision, WS-kNN and WS-Adapter.

Weak Sources and Dev Labels Next, we compare performance against methods that have access to a small hand-labeled dev set. We compare against two baselines: kNN and Adapter, both trained over the dev set labels. For our method, we train an adapter over LIGER labels on the train set, as well as the dev labels. In some cases, LIGER labels are too noisy to provide good signal on the train set; in this case, our solution automatically downsamples the trained pseudolabels. We also provide the original LIGER prediction as input to the Adapter. See Appendix E for the details.

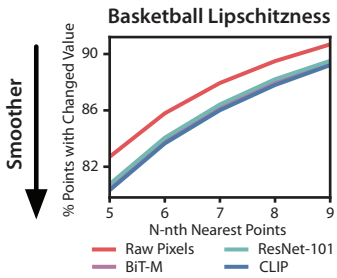
Table 1 (right) shows the results. LIGER-Adapter outperforms Adapters and kNN. On the datasets where LIGER labels are very accurate, we see additional lift from the Adapters because we have more points to train on. When the labels are not very accurate, our downsampling prevents the noisy labels from harming Adapter performance. In one case, learning an Adapter over the embeddings is very hard (**Spouse**). Here, providing the LIGER prediction as input is critical for performance.

5.2 EMBEDDING SMOOTHNESS

We measure how smoothness of the embedding space affects the performance of LIGER. First, we compare embeddings from CLIP against BiT-M embeddings [Kolesnikov et al., 2020], a ResNet-101 pretrained on ImageNet [Russakovsky et al., 2015], and raw pixels. Second, we vary the GPT-3 prompting strategy for **Spouse** and compare against two alternative methods that result in a less smooth representation. We report label Lipschitzness—the smoothness of embeddings with respect to ground-truth labels—in this section. See Appendix F for additional measures of Lipschitzness.

Comparing Image Embeddings Figure 2 (left) shows the performance of CLIP, BiT-M, ResNet-101, and raw pixels as embeddings for LIGER, as well as measures of Lipschitzness for each method (lower is smoother). CLIP embeddings are smoother than the other methods—which matches their

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 Beginning	50.2
Prompt End	**52.2**

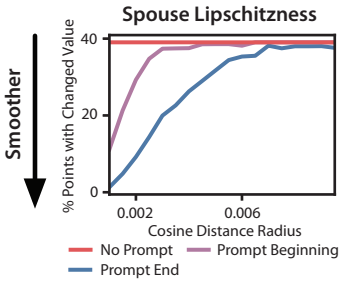


Figure 2: Left: LIGER performance and smoothness measurements of CLIP, BiT-M, ResNet-101, and raw pixels as embeddings for **Basketball**. Right: LIGER performance and smoothness measurements of no prompting, prompting at beginning, and prompting at end in GPT-3 for **Spouse**.

performance when used in LIGER.

Comparing Prompting Strategies Next, we examine the impact of prompting strategies for **Spouse**. **Spouse** is a relation extraction dataset, where the task is to predict whether two entities in a sentence are married. Since there may be multiple entities in a sentence, **Spouse** contains multiple duplicate sentences in the dataset, with different labels. To alleviate this problem, we introduce a prompt "Are [person 1] and [person 2] spouses?" after the end of the sentence, where "[person 1/2]" are replaced by the names of the first/second entity in the sentence. We compare this prompting strategy against two others: appending the same prompt to the beginning of the sentence, and leaving the original sentence as-is, without any prompting.

Figure 2 (right) shows the performance and smoothness of each of these prompting methods. Adding the prompt to the end of the sentence results in the best performance and smoothest embeddings. Both methods perform better than leaving the sentence alone (the flat line is a result of multiple sentences with different labels having the same embedding).

5.3 ABLATIONS

Task	LIGER (s)	-Part	-Ext (s)	-Part, -Ext
Spam	**95.0** (2)	94.0	92.0 (7)	83.6
Weather	**98.0** (3)	96.0	94.0 (5)	78.0
Spouse	**52.2** (6)	50.0	49.3 (5)	47.0
Basketball	**69.6** (2)	**69.6**	21.9 (2)	27.9
Commercial	**93.5** (3)	92.3	91.4 (5)	88.4
Tennis	**83.3** (1)	**83.3**	81.3 (2)	82.0

Table 2: Ablations of LIGER, removing partitions (-Part), extensions (-Ext), and both. Best s values inside parentheses.

We report ablations on each component of LIGER. Table 2 removes the partioning component and the extensions component. Partitioning improves performance in four tasks, and extensions improves performance on their own in all tasks (13.1 points of lift on average from partitioning, 3.8 points from extensions). Combining both additionally offers the best performance in four tasks.

6 RELATED WORK

We present an abbreviated related work here. See Appendix A for an extended treatment.

Weak supervision frameworks typically model source accuracies to generate weak labels and then fine-tunes an 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]. One framework models the end-to-end process all at once [Cachay et al., 2021], but requires training the end model at the same time—which is computationally expensive with large foundation models. Our work removes the fine-tuning step completely.

Our work is similar to transfer learning techniques, which adapt pretrained models for downstream tasks [Kolesnikov et al., 2020, Devlin et al., 2018]. Foundation models offer new requirements for transfer learning setting: when it is impossible to fine-tune the original models [Bommasani et al., 2021]. We build on approaches such as prompting [Lester et al., 2021, Brown et al., 2020], embedding search [Neelakantan et al., 2022], and adapters [Houlsby et al., 2019, Alain and Bengio, 2016].

7 CONCLUSION

We present LIGER, a system for fusing foundation models and weak supervision. We use embeddings to produce finer-grained estimates of weak source accuracies and improve weak source coverage. We prove a series of results on how the performance of this approach scales with the smoothness of the embeddings, and demonstrate LIGER on six benchmark NLP and video weak supervision datasets. We hope our work will encourage further work in combining foundation models and weak supervision.

References

Weather sentiment: Dataset in crowdflower. https://data.world/crowdflower/weather-sentiment.

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

Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, 2015.

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.

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

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.

David Corney, Dyaa Albakour, Miguel Martinez-Alvarez, and Samir Moussa. What do a million news articles look like? In *NewsIR@ECIR*, 2016.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the Association for Computational Linguistics*, 2019.

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.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. 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.

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

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, Shiying 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.

Ruth Urner and Shai Ben-David. Probabilistic lipschitzness a niceness assumption for deterministic labels. In *Learning Faster from Easy Data-Workshop@NIPS*, 2013.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Re. Learning dependency structures for weak supervision models. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

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.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. Wrench: A comprehensive benchmark for weak supervision. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

APPENDIX

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]. Adapters are a promising technique for using these foundation models [Houlsby et al., 2019]. Since we can only access the final embeddings of some foundation models, we focus on adapters 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.

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

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, $\hat{a}_i(C_j)$ (Section C.2).

C.1 PROPERTIES OF THE GRAPHICAL MODEL

Lemma 2. *For $x, y, \boldsymbol{\lambda}$ satisfying* (1)*, it holds for any λ_i that*

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

That is, $y \perp\!\!\!\perp \mathbb{1}\{\lambda_i = 0\}|x$.

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

$$\begin{aligned}\Pr(y = 1, \lambda_i = 0|x) &= \frac{1}{Z}\sum_{\lambda_{-i}} \exp\big(\theta_y(x) + \theta_{i,0}(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\}\big) \\ &= \frac{1}{Z}\exp(\theta_y(x) + \theta_{i,0}(x))\sum_{\lambda_{-i}} \exp\big(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\}\big).\end{aligned} \tag{4}$$

$\Pr(y = 1|x)$ can be written as $\Pr(y = 1, \lambda_i = 1|x) + \Pr(y = 1, \lambda_i = -1|x) + \Pr(y = 1, \lambda_i = 0|x)$:

$$\begin{aligned}\Pr(y = 1|x) = \frac{1}{Z}\sum_{\lambda_{-i}} \Big(&\exp(\theta_y(x) + \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\}) \\ &+ \exp(\theta_y(x) - \theta_i(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\}) \\ &+ \exp(\theta_y(x) + \theta_{i,0}(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\})\Big) \\ = \frac{1}{Z}\sum_{\lambda_{-i}} &\exp(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\})\exp(\theta_y(x))\big(\exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x))\big).\end{aligned} \tag{5}$$

$\Pr(\lambda_i = 0|x)$ can be written as $\Pr(y = 1, \lambda_i = 0|x) + \Pr(y = -1, \lambda_i = 0|x)$:

$$\begin{aligned}\Pr(\lambda_i = 0|x) = \frac{1}{Z}\sum_{\lambda_{-i}} \Big(&\exp(\theta_y(x) + \theta_{i,0}(x) + \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\}) \\ &+ \exp(-\theta_y(x) + \theta_{i,0}(x) - \theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\})\Big).\end{aligned} \tag{6}$$

Setting (4) equal to (5) times (6), the term $\sum_{\lambda_{-i}} \exp(\theta_{-i}(x)\lambda_{-i} + \theta_{-i,0}(x)\mathbb{1}\{\lambda_{-i} = 0\})$ in the former two equations cancels out. We thus aim to prove the following equality:

$$\begin{aligned}Z\exp(\theta_y(x) + \theta_{i,0}(x)) = \exp(\theta_y(x))\Big(&\exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x))\Big)\times \\ \sum_{\lambda_{-i}} &\exp(\theta_{-i,0}\mathbb{1}\{\lambda_{-i} = 0\})\exp(\theta_{i,0}(x))\Big(\exp(\theta_y(x) + \theta_{-i}(x)\lambda_{-i}(x)) + \exp(-\theta_y(x) - \theta_{-i}(x)\lambda_{-i}(x))\Big).\end{aligned} \tag{7}$$

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

$$\begin{aligned}Z = &\Big(\exp(\theta_i(x)) + \exp(-\theta_i(x)) + \exp(\theta_{i,0}(x))\Big)\times \\ &\sum_{\lambda_{-i}} \exp(\theta_{-i,0}\mathbb{1}\{\lambda_{-i} = 0\})\Big(\exp(\theta_y(x) + \theta_{-i}(x)\lambda_{-i}(x)) + \exp(-\theta_y(x) - \theta_{-i}(x)\lambda_{-i}(x))\Big),\end{aligned}$$

which is true since the RHS iterates over all values of λ_{-i}, y, and λ_i. We have shown that $\Pr(y = 1, \lambda_i = 0|x) = \Pr(y = 1|x)\Pr(\lambda_i = 0|x)$ and thus that $y \perp\!\!\!\perp \mathbb{1}\{\lambda_i = 0\}|x$ for any λ_i.

Due to this independence property, we note that

$$\begin{aligned}\Pr(\lambda_i = 0, \lambda_{-i}|x) &= \Pr(\lambda_i = 0, \lambda_{-i}|y = 1, x)\Pr(y = 1|x) + \Pr(\lambda_i = 0, \lambda_{-i}|y = -1, x)\Pr(y = -1|x) \\ &= \Pr(\lambda_i = 0|x)\Big(\Pr(\lambda_{-i}|y = 1, x)\Pr(y = 1|x) + \Pr(\lambda_{-i}|y = -1, x)\Pr(y = -1|x)\Big) \\ &= \Pr(\lambda_i = 0|x)\Pr(\lambda_{-i}|x),\end{aligned}$$

and hence

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

□

Lemma 3. *For any $i \neq j$, if $x, \boldsymbol{\lambda}, y$ follows* (1)*, then $\lambda_i y \perp\!\!\!\perp \lambda_j y|\lambda_i \wedge \lambda_j \neq 0, x$.*

Proof. Conditioning on the event that $\boldsymbol{\lambda} \neq 0$, we have that

$$\Pr(y, \boldsymbol{\lambda}|\boldsymbol{\lambda} \neq 0, x) = \frac{1}{Z_0}\exp\Big(\theta_y(x)y + \sum_{i=1}^{m}\theta_i(x)\lambda_i y\Big) \tag{8}$$

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] (see their Equation 3). We can thus apply Proposition 1 of their work to get that $\lambda_i y \perp\!\!\!\perp \lambda_j y|\boldsymbol{\lambda} \neq 0, x$. From Lemma 2 and conditional independence of sources, this independence property is equivalent to $\lambda_i y \perp\!\!\!\perp \lambda_j y|\lambda_i \wedge \lambda_j \neq 0, x$, as desired. □

Lemma 4. *If $x, \boldsymbol{\lambda}, y$ follows* (1)*, then for any λ_i,*

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

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

Therefore,

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

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

C.2 LOCAL ACCURACY PARAMETER ESTIMATION ALGORITHM

We formalize the triplet method used to recover latent source parameters $\Pr(\lambda_i|y, x)$. First, 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 $\Pr(\lambda_i|y, x) = \frac{1+\text{sgn}(\lambda_i y)a_i(x)}{2} \cdot \Pr(\lambda_i \neq 0|x)$, and we explain how our algorithm estimates $a_i(x)$.

Algorithm 2 Local Accuracy Estimation (Triplet Method)

Input: Dataset $\mathcal{D}$, weak sources $\bar{\boldsymbol{\lambda}}$, partition C_j.
Returns: Estimate of local accuracy $\hat{a}_i(C_j)$.
for $k, l \in [m] \backslash i$ **do**
  Estimate $\hat{\mathbb{E}}\left[\bar{\lambda}_i \bar{\lambda}_k | \bar{\lambda}_i \wedge \bar{\lambda}_k \neq 0, C_j\right]$ over the set of points $\{x \in \mathcal{D} : \bar{\lambda}_i(x), \bar{\lambda}_k(x) \neq 0, f(x) \in C_j\}$, and similarly estimate $\hat{\mathbb{E}}\left[\bar{\lambda}_i \bar{\lambda}_l | \bar{\lambda}_i \wedge \bar{\lambda}_l \neq 0, C_j\right]$ and $\hat{\mathbb{E}}\left[\bar{\lambda}_k \bar{\lambda}_l | \bar{\lambda}_k \wedge \bar{\lambda}_l \neq 0, C_j\right]$.
  Compute $\hat{a}_i^{k,l}(C_j) = \sqrt{\left|\frac{\hat{\mathbb{E}}\left[\bar{\lambda}_i \bar{\lambda}_k | \bar{\lambda}_i \wedge \bar{\lambda}_k \neq 0, C_j\right] \hat{\mathbb{E}}\left[\bar{\lambda}_i \bar{\lambda}_l | \bar{\lambda}_i \wedge \bar{\lambda}_l \neq 0, C_j\right]}{\hat{\mathbb{E}}\left[\bar{\lambda}_k \bar{\lambda}_l | \bar{\lambda}_k \wedge \bar{\lambda}_l \neq 0, C_j\right]}\right|}$.
end for
return $\hat{a}_i(C_j)$ as the average over all $\hat{a}_i^{k,l}(C_j)$.

From Lemma 3, we have that $\lambda_i y \perp\!\!\!\perp \lambda_j y | \lambda_i \wedge \lambda_j \neq 0, x$ for any i, j. Then, given any set of $\lambda_i, \lambda_j, \lambda_k$, we have the set of equations

$$\begin{aligned} a_i(x) a_j(x) &= \mathbb{E}\left[\lambda_i \lambda_j | \lambda_i \wedge \lambda_j \neq 0, x\right] \\ a_i(x) a_k(x) &= \mathbb{E}\left[\lambda_i \lambda_k | \lambda_i \wedge \lambda_k \neq 0, x\right] \\ a_j(x) a_k(x) &= \mathbb{E}\left[\lambda_j \lambda_k | \lambda_j \wedge \lambda_k \neq 0, x\right]. \end{aligned}$$

Solving, we get that

$$|a_i(x)| = \sqrt{\left|\frac{\mathbb{E}\left[\lambda_i \lambda_j | \lambda_i \wedge \lambda_j \neq 0, x\right] \mathbb{E}\left[\lambda_i \lambda_k | \lambda_i \wedge \lambda_k \neq 0, x\right]}{\mathbb{E}\left[\lambda_j \lambda_k | \lambda_j \wedge \lambda_k \neq 0, x\right]}\right|}.$$

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 $a_i(C(x)) = \mathbb{E}\left[\lambda_i y | \lambda_i \neq 0, C(x)\right]$ (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 λ_j and λ_k, we compute the average $a_i(C(x))$ over all other pairs ($\lambda_j, \lambda_k \in \boldsymbol{\lambda} \backslash \lambda_i$) to make the estimate less noisy. Our approach for computing $\hat{a}_i(C_j)$ for any $\bar{\lambda}_i$ and C_j (note that $\bar{\lambda}_i$ and λ_i are interchangeable in the above given that $(x, y, \boldsymbol{\lambda})$ and $(x, y, \bar{\boldsymbol{\lambda}})$ 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 $x, y, \boldsymbol{\lambda}$ follows the model in* (1)*. The generalization error of the label model $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ in Algorithm 1 when $r_i = 0\ \forall i$ can be decomposed into $L(\boldsymbol{\lambda})$=Bias + Variance + Irreducible Error + $o(1/n)$, where*

$$\begin{aligned} &\textit{Bias} \leq 2 d_{\mathcal{C}}(K_y + m K_\lambda + m K_{\lambda,0}), \\ &\textit{Variance} \leq \frac{ms}{2n}\left(\frac{3(1-b_{\min}^2)}{4 b_{\min}^2 (1-a_{\max})^2}\left(\frac{1}{b_{\min}^4} + \frac{2}{b_{\min}^2}\right) \right. \\ &\qquad\qquad \left. + \frac{1-p_{\min}}{p_{\min}}\right), \\ &\textit{Irreducible Error} = H(y|\boldsymbol{\lambda}, x), \end{aligned}$$

where $H(y|\boldsymbol{\lambda}, x)$ denotes conditional entropy.

Proof. We can write the generalization error as

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

Note that $-\mathbb{E}_{x,\boldsymbol{\lambda}}\left[\Pr(y|\boldsymbol{\lambda},x)\right]$ is equal to the conditional entropy of y given $\boldsymbol{\lambda}, x$, expressed as $H(y|\boldsymbol{\lambda},x)$. 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 to get

$$\begin{aligned}
L(\boldsymbol{\lambda}) &= \mathbb{E}\left[-\log\left(\frac{\prod_{i=1}^m \hat{\Pr}(\lambda_i|y,C(x))\Pr(y|C(x))}{\hat{\Pr}(\boldsymbol{\lambda}|C(x))}\cdot\frac{\Pr(\boldsymbol{\lambda}|x)}{\prod_{i=1}^m \Pr(\lambda_i|y,x)\Pr(y|x)}\right)\right] + H(y|\boldsymbol{\lambda},x)\\
&= -\mathbb{E}\left[\sum_{i=1}^m \log\frac{\hat{\Pr}(\lambda_i|y,C(x))}{\Pr(\lambda_i|y,x)}\right] - \mathbb{E}\left[\log\frac{\Pr(\boldsymbol{\lambda}|x)}{\hat{\Pr}(\boldsymbol{\lambda}|C(x))}\right] - \mathbb{E}\left[\log\frac{\Pr(y|C(x))}{\Pr(y|x)}\right] + H(y|\boldsymbol{\lambda},x)\\
&= -\mathbb{E}\left[\sum_{i=1}^m \log\frac{\hat{\Pr}(\lambda_i|y,C(x))}{\Pr(\lambda_i|y,x)}\right] - \mathbb{E}_x\left[D_{\mathrm{KL}}(\Pr(\boldsymbol{\lambda}|x)||\hat{\Pr}(\boldsymbol{\lambda}|C(x)))\right] - \mathbb{E}\left[\log\frac{\Pr(y|C(x))}{\Pr(y|x)}\right] + H(y|\boldsymbol{\lambda},x)\\
&\le \mathbb{E}\left[\sum_{i=1}^m \log\frac{\Pr(\lambda_i|y,x,\lambda_i\neq 0)\Pr(\lambda_i\neq 0|x)}{\hat{\Pr}(\lambda_i|y,C(x),\lambda_i\neq 0)\hat{\Pr}(\lambda_i\neq 0|C(x))}\right] - \mathbb{E}\left[\log\frac{\Pr(y|C(x))}{\Pr(y|x)}\right] + H(y|\boldsymbol{\lambda},x),
\end{aligned}$$

where we have used Lemma 2 and the fact that the Kullback-Leibler divergence is always nonnegative in the last line. Then, we further decompose the first expectation into bias and variance terms to get

$$\begin{aligned}
L(\boldsymbol{\lambda}) \le& \sum_{i=1}^m \mathbb{E}\left[\log\left(\frac{\Pr(\lambda_i|y,C(x),\lambda_i\neq 0)}{\hat{\Pr}(\lambda_i|y,C(x),\lambda_i\neq 0)}\cdot\frac{\Pr(\lambda_i|y,x,\lambda_i\neq 0)}{\Pr(\lambda_i|y,C(x),\lambda_i\neq 0)}\right)\right]\\
&+\sum_{i=1}^m \mathbb{E}\left[\log\left(\frac{\Pr(\lambda_i\neq 0|C(x))}{\hat{\Pr}(\lambda_i\neq 0|C(x))}\cdot\frac{\Pr(\lambda_i\neq 0|x)}{\Pr(\lambda_i\neq 0|C(x))}\right)\right] + \mathbb{E}_x\left[D_{\mathrm{KL}}(\Pr(y|x)||\Pr(y|C(x)))\right] + H(y|\boldsymbol{\lambda},x)\\
=& \sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i|y,C(x),\lambda_i\neq 0)}{\hat{\Pr}(\lambda_i|y,C(x),\lambda_i\neq 0)}\right] + \sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i\neq 0|C(x))}{\hat{\Pr}(\lambda_i\neq 0|C(x))}\right]\\
&+\sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i|y,x,\lambda_i\neq 0)}{\Pr(\lambda_i|y,C(x),\lambda_i\neq 0)}\right] + \sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i\neq 0|x)}{\Pr(\lambda_i\neq 0|C(x))}\right]\\
&+\mathbb{E}_x\left[D_{\mathrm{KL}}(\Pr(y|x)||\Pr(y|C(x)))\right] + H(y|\boldsymbol{\lambda},x)
\end{aligned}$$

We can write $\Pr(\lambda_i|y,C(x),\lambda_i\neq 0)$ as $\frac{1+\mathrm{sgn}(\lambda_i y)a_i(C(x))}{2}$ by Lemma 4 , where $a_i(C(x)) = \mathbb{E}\left[\lambda_i y|\lambda_i\neq 0, C(x)\right]$ and similarly $\hat{\Pr}(\lambda_i|y,C(x),\lambda_i\neq 0)$ with $\hat{a}_i(C(x))$. We convert the other log ratios between distributions conditioned on x and distributions conditioned on $C(x)$ into KL-divergence terms. Our error bound is now

$$\begin{aligned}
L(\boldsymbol{\lambda}) \le& \sum_{i=1}^m \mathbb{E}\left[\log\frac{1+\mathrm{sgn}(\lambda_i y)\cdot a_i(C(x))}{1+\mathrm{sgn}(\lambda_i y)\cdot \hat{a}_i(C(x))}\right] + \sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i\neq 0|C(x))}{\hat{\Pr}(\lambda_i\neq 0|C(x))}\right]\\
&+\sum_{i=1}^m \mathop{\mathbb{E}}_{x,y,\lambda_i\neq 0}[D_{\mathrm{KL}}(\Pr(\lambda_i|y,x,\lambda_i\neq 0)||\Pr(\lambda_i|y,C(x),\lambda_i\neq 0))] + \sum_{i=1}^m \mathbb{E}_x\left[D_{\mathrm{KL}}(\Pr(\lambda_i\neq 0|x)||\Pr(\lambda_i\neq 0|C(x)))\right]\\
&+\mathbb{E}_x\left[D_{\mathrm{KL}}(\Pr(y|x)||\Pr(y|C(x)))\right] + H(y|\boldsymbol{\lambda},x).
\end{aligned}$$

The first two summations correspond to the variance from the sampling error of accuracy parameters and coverage. The terms with the KL-divergences correspond to the bias of conditioning on $C(x)$ instead of x, and the last term, the conditional entropy, denotes the irreducible error. Combining the results of Lemmas 5, 6, and 7, we get our desired bound. □

Lemma 5. *The sampling error term coming from estimating* $\Pr(\lambda_i\neq 0|C(x))$ *is overall at most*

$$\sum_{i=1}^m \mathbb{E}\left[\log\frac{\Pr(\lambda_i\neq 0|C(x))}{\hat{\Pr}(\lambda_i\neq 0|C(x))}\right] \le \frac{ms(1-p_{\min})}{2p_{\min}n} + o(1/n).$$

Proof. We can write $\mathbb{E}\left[\log \frac{\Pr(\lambda_i \neq 0|C(x))}{\hat{\Pr}(\lambda_i \neq 0|C(x))}\right]$ as $\sum_{j=1}^{s} \Pr(f(x) \in C_j)\mathbb{E}_{\mathcal{D}}\left[\log \frac{\Pr(\lambda_i \neq 0|C_j)}{\hat{\Pr}(\lambda_i \neq 0|C_j)}\middle| C_j\right]$. For ease of notation, define $p_{i,j} = \Pr(\lambda_i \neq 0|C_j)$ as λ_i's coverage on C_j, and equivalently $\hat{p}_{i,j}$ as the empirical estimate on $\mathcal{D}$. Performing a Taylor approximation of $f(x) = \log \frac{c}{x}$ at $x = c$ gives us $\log \frac{c}{x} \approx \log 1 + -\frac{1}{c}(x-c) + \frac{1}{c^2}(x-c)^2$, so

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[\log \frac{p_{i,j}}{\hat{p}_{i,j}}\middle| C_j\right] &= -\frac{1}{p_{i,j}}\mathbb{E}_{\mathcal{D}}\left[\hat{p}_{i,j} - p_{i,j}\middle| C_j\right] + \frac{1}{2p_{i,j}^2}\mathbb{E}_{\mathcal{D}}\left[(\hat{p}_{i,j} - p_{i,j})^2\middle| C_j\right] + o(1/n) \\ &= \frac{1}{2p_{i,j}^2}\text{Var}\left[\hat{p}_{i,j}\middle| C_j\right] + o(1/n),\end{aligned}$$

where we use the fact that $\hat{p}_{i,j}$ is an unbiased estimate since λ_i is observable, and that the Taylor remainder scales in $\mathbb{E}\left[(\hat{p}_{i,j} - p_{i,j})^3\middle| C_j\right] \sim \mathcal{O}(1/n^2)$. We can write $\text{Var}\left[\hat{p}_{i,j}\middle| C_j\right] = \text{Var}\left[\frac{1}{n'}\sum_{x:f(x)\in C_j} \mathbb{1}\left\{\lambda_i(x) \neq 0\right\}\middle| C_j\right] = \frac{1}{(n')^2}\sum_{x:f(x)\in C_j} \text{Var}\left[\mathbb{1}\left\{\lambda_i(x) \neq 0\right\}\middle| C_j\right] = \frac{p_{i,j}(1-p_{i,j})}{n'}$. Putting this all together, we have

$$\begin{aligned}\sum_{i=1}^{m}\mathbb{E}\left[\log \frac{\Pr(\lambda_i \neq 0|C(x))}{\hat{\Pr}(\lambda_i \neq 0|C(x))}\right] &= \sum_{i=1}^{m}\sum_{j=1}^{s}\Pr(f(x) \in C_j) \cdot \frac{1}{2p_{i,j}^2} \cdot \frac{p_{i,j}(1-p_{i,j})}{n'} + o(1/n) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{s}\Pr(f(x) \in C_j) \cdot \frac{s(1-p_{i,j})}{2p_{i,j}n} + o(1/n) \\ &= \sum_{i=1}^{m}\frac{s(1-\mathbb{E}_x\left[\Pr(\lambda_i \neq 0|C(x))\right]}{2\mathbb{E}_x\left[\Pr(\lambda_i \neq 0|C(x))\right]n} + o(1/n) \\ &\leq \frac{ms(1-p_{\min})}{2p_{\min}n} + o(1/n).\end{aligned}$$

□

Lemma 6. *The sampling error term coming from estimating* $\Pr(\lambda_i|y, C(x), \lambda_i \neq 0)$ *is overall at most*

$$\sum_{i=1}^{m}\mathbb{E}\left[\log \frac{1 + sgn(\lambda_i y) \cdot a_i(C(x))}{1 + sgn(\lambda_i y) \cdot \hat{a}_i(C(x))}\right] \leq \frac{3ms}{8n} \cdot \frac{1 - b_{min}^2}{b_{min}^2(1-a_{\max})^2}\left(\frac{1}{b_{min}^4} + \frac{2}{b_{min}^2}\right) + o(1/n).$$

Proof. We can write $\mathbb{E}\left[\log \frac{1+\text{sgn}(\lambda_i y)\cdot a_i(C(x))}{1+\text{sgn}(\lambda_i y)\cdot \hat{a}_i(C(x))}\right]$ as $\sum_{j=1}^{s}\Pr(f(x) \in C_j)\mathbb{E}\left[\log \frac{1+\text{sgn}(\lambda_i y)\cdot a_i(C_j)}{1+\text{sgn}(\lambda_i y)\cdot \hat{a}_i(C_j)}\middle| C_j\right]$. We can decompose each expectation as

$$\mathbb{E}\left[\log \frac{1 + \text{sgn}(\lambda_i y) \cdot a_i(C_j)}{1 + \text{sgn}(\lambda_i y) \cdot \hat{a}_i(C_j)}\middle| C_j\right] = \mathbb{E}\left[\log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)}\middle| C_j\right]\Pr(\lambda_i y = 1|C_j) + \mathbb{E}\left[\log \frac{1 - a_i(C_j)}{1 - \hat{a}_i(C_j)}\middle| C_j\right]\Pr(\lambda_i y = -1|C_j). \tag{11}$$

$\Pr(\lambda_i y = 1|C_j) = \Pr(\lambda_i y = 1|C_j, \lambda_i \neq 0)\Pr(\lambda_i \neq 0|C_j) = \frac{1+a_i(C_j)}{2} \cdot p_{i,j}$, where we recall that $p_{i,j} = \Pr(\lambda_i \neq 0|C_j)$ is λ_i's coverage on C_j.(11) becomes

$$\mathbb{E}\left[\log \frac{1 + \text{sgn}(\lambda_i y) \cdot a_i(C_j)}{1 + \text{sgn}(\lambda_i y) \cdot \hat{a}_i(C_j)}\middle| C_j\right] = \frac{p_{i,j}}{2}\left((1 + a_i(C_j))\mathbb{E}\left[\log \frac{1 + a_i(C_j)}{1 + \hat{a}_i(C_j)}\middle| C_j\right] + (1 - a_i(C_j))\mathbb{E}\left[\log \frac{1 - a_i(C_j)}{1 - \hat{a}_i(C_j)}\middle| C_j\right]\right). \tag{12}$$

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

Therefore, (12) becomes

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

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

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

Therefore, the overall expression can be bounded by

$$\sum_{i=1}^{m}\mathbb{E}\left[\log\frac{1+\text{sgn}(\lambda_i y)\cdot a_i(C(x))}{1+\text{sgn}(\lambda_i y)\cdot \hat{a}_i(C(x))}\right] \leq \sum_{i=1}^{m}\sum_{j=1}^{s}\Pr(f(x)\in C_j)\frac{p_{i,j}}{2(1-a_{\max})^2}\cdot\frac{3s}{4p_{i,j}n}\cdot\frac{1-b_{\text{min}}^2}{b_{\text{min}}^2}\left(\frac{1}{b_{\text{min}}^4}+\frac{2}{b_{\text{min}}^2}\right) + o(1/n)$$
$$= \frac{3ms}{8n}\cdot\frac{1-b_{\text{min}}^2}{b_{\text{min}}^2(1-a_{\max})^2}\left(\frac{1}{b_{\text{min}}^4}+\frac{2}{b_{\text{min}}^2}\right) + o(1/n).$$

□

Lemma 7. *The bias terms from conditioning on $C(x)$ rather than x are at most*

$$\mathbb{E}_x\left[D_{KL}(\Pr(y|x)||\Pr(y|C(x)))\right] \leq 2K_y d_{\mathcal{C}}$$
$$\sum_{i=1}^{m}\mathop{\mathbb{E}}_{x,y,\lambda_i\neq 0}[D_{KL}(\Pr(\lambda_i|y,x,\lambda_i\neq 0)||\Pr(\lambda_i|y,C(x),\lambda_i\neq 0))] \leq 2mK_\lambda d_{\mathcal{C}}$$
$$\sum_{i=1}^{m}\mathbb{E}_x\left[D_{KL}(\Pr(\lambda_i\neq 0|x)||\Pr(\lambda_i\neq 0|C(x)))\right] \leq 2mK_{\lambda,0}d_{\mathcal{C}}.$$

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

$$\mathbb{E}_x\left[D_{\text{KL}}(\Pr(y|x)||\Pr(y|C(x)))\right] = \sum_{j=1}^{s}\Pr(f(x)\in C_j)\int\Pr(x|C_j)D_{\text{KL}}(\Pr(y|x,C_j)||\Pr(y|C_j))dx. \tag{13}$$

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, $D_{\text{KL}}(\Pr(y|x,C_j)||\Pr(y|C_j)) = p_{y,x,j}\log\frac{p_{y,x,j}}{p_{y,j}} + (1-p_{y,x,j})\log\frac{1-p_{y,x,j}}{1-p_{y,j}}$.

Next, recall that $\Pr(y|x)$ is K_y-Lipschitz in the embedding space; that is, $|\Pr(y=1|x)-\Pr(y=1|x')| \leq K_y\rho(f(x),f(x'))$. 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

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

where we use the fact that $\log(1+x) \leq x$. Plugging this back into (13),

$$\begin{aligned}
\mathbb{E}_x\left[D_{\text{KL}}(\Pr(y|x)||\Pr(y|C(x)))\right] &\leq \sum_{j=1}^{s} \Pr(f(x)\in C_j)\int \Pr(x|C_j))\left(p_{y,x,j}\cdot\frac{K_y d_j}{p_{y,j}} + (1-p_{y,x,j})\cdot\frac{K_y d_j}{1-p_{y,j}}\right)dx \\
&= \sum_{j=1}^{s} \Pr(f(x)\in C_j)\int \Pr(x, y=1|C_j)\cdot\frac{K_y d_j}{p_{y,j}} + \Pr(x, y=-1|C_j)\cdot\frac{K_y d_j}{1-p_{y,j}}dx \\
&= \sum_{j=1}^{s} \Pr(f(x)\in C_j)\left(\Pr(y=1|C_j)\cdot\frac{K_y d_j}{p_{y,j}} + \Pr(y=-1|C_j)\cdot\frac{K_y d_j}{1-p_{y,j}}\right) \\
&= \sum_{j=1}^{s} \Pr(f(x)\in C_j)\cdot 2K_y d_j = 2K_y d_{\mathcal{C}}
\end{aligned}$$

Next, we bound $\mathbb{E}_{x,y,\lambda_i\neq 0}\left[D_{\text{KL}}(\Pr(\lambda_i|y,x,\lambda_i\neq 0)||\Pr(\lambda_i|y,C(x),\lambda_i\neq 0))\right]$. Using the same approach, we have that $\mathbb{E}_{x,y}\left[D_{\text{KL}}(\Pr(\lambda_i|y,x)||\Pr(\lambda_i|y,C(x)))\right] \leq 2K_\lambda d_{\mathcal{C}}$. We also have that $\mathbb{E}_x\left[D_{\text{KL}}(\Pr(\lambda_i\neq 0|x)||\Pr(\lambda_i\neq 0|C(x)))\right] \leq 2K_{\lambda,0}d_{\mathcal{C}}$. □

D.2 PROOFS FOR SECTION 4.2

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

$$Bias \leq 2d_{\mathcal{C}}K_y + 2m(d_{\mathcal{C}} + 2\max_i r_i)(K_\lambda + K_{\lambda,0}).$$

Proof. The term $\mathbb{E}_x\left[D_{\text{KL}}(\Pr(y|x)||\Pr(y|C(x))\right]$ in the bias is unchanged since the distribution of y given x is not impacted by $\boldsymbol{\lambda}$. We next look at $\mathbb{E}_{x,y,\bar{\lambda}_i\neq 0}\left[D_{\text{KL}}(\Pr(\bar{\lambda}_i|y,x,\bar{\lambda}_i\neq 0)||\Pr(\bar{\lambda}_i|y,C(x),\bar{\lambda}_i\neq 0))\right]$. Using the approach in Lemma 7, recall that $\Pr(\bar{\lambda}_i|y,x,\bar{\lambda}_i\neq 0) = \Pr(\lambda_i(x)|y,x,\lambda_i(x)\neq 0)$ 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, $|\Pr(\bar{\lambda}_i = 1|y,\bar{\lambda}_i\neq 0,x) - \Pr(\bar{\lambda}_i = 1|y,\lambda_i\neq 0,x')| \leq K_\lambda\max\{\rho(f(\text{NN}(x)),f(\text{NN}(x'))),\rho(f(x),f(\text{NN}(x'))),\rho(f(\text{NN}(x)),f(x')),\rho(f(x),f(x'))\}$. The greatest 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

$$\mathbb{E}_{x,y,\bar{\lambda}_i\neq 0}\left[D_{\text{KL}}(\Pr(\bar{\lambda}_i|y,x,\bar{\lambda}_i\neq 0)||\Pr(\bar{\lambda}_i|y,C(x),\bar{\lambda}_i\neq 0))\right] \leq 2K_\lambda(d_{\mathcal{C}} + 2r_i).$$

We consider $\mathbb{E}_x\left[D_{\text{KL}}(\Pr(\lambda_i\neq 0|x)||\Pr(\lambda_i\neq 0|C(x)))\right]$. Similarly, $\Pr(\bar{\lambda}_i(x)\neq 0|x)$ 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,

$$\mathbb{E}_x\left[D_{\text{KL}}(\Pr(\lambda_i\neq 0|x)||\Pr(\lambda_i\neq 0|C(x)))\right] \leq 2K_{\lambda,0}(d_{\mathcal{C}} + 2r_i),$$

and we obtain the desired bound. □

Lemma 1. *The average accuracy of $\bar{\lambda}_i$ on the extended region is at least $\bar{a}_i(r_i) \geq a_i - (1+\beta_i)M(r_i)$.*

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

Define the expected error $\varepsilon = \mathbb{E}_{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0}}\left[\Pr_{x,y\sim\mathcal{P}_{\hat{S},r_i}}(\bar{\lambda}_i\neq y|x,\bar{\lambda}_i(x)\neq 0)\right] = \mathbb{E}_{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0}}\left[\Pr_{x,y\sim\mathcal{P}_{\hat{S},r_i}}(\bar{\lambda}_i\neq y|x)\right]$. Let $\hat{S}$ also be written as a set of n_0 random variables $\{x_1,\ldots,x_{n_0}\}$. Denote $\text{NN}_{\hat{S}}(x) = \text{argmin}_{x'\in\hat{S}}\,\rho(f(x),f(x'))$ to be x's

nearest neighbor in $\hat{S}$ (in the body, this is just referred to as $\text{NN}(x)$), so $\bar{\lambda}_i(x) := \lambda_i(\text{NN}_{\hat{S}}(x))$. Then, we decompose ε based on which point in $\hat{S}$ is x's nearest neighbor:

$$\varepsilon = \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y\sim\mathcal{P}_{\hat{S},r_i}}}(\lambda_i(\text{NN}_{\hat{S}}(x)) \neq y|x) = \sum_{j=1}^{n_0} \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0}, \\ x\sim\mathcal{P}_{\hat{S},r_i}}}(\text{NN}_{\hat{S}}(x) = x_j) \cdot \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y,\sim\mathcal{P}_{\hat{S},r_i}}}(\lambda_i(x_j) \neq y|\text{NN}_{\hat{S}}(x) = x_j). \tag{14}$$

Let y_j denote the label corresponding to x_j, drawn from $\mathcal{P}_{\lambda_i}(\cdot|x_j)$. The probability $\Pr_{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0},x,y,\sim\mathcal{P}_{\hat{S},r_i}}(\lambda_i(x_j) \neq y|\text{NN}_{\hat{S}}(x) = x_j)$ 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,

$$\begin{aligned}
&\Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y,\sim\mathcal{P}_{\hat{S},r_i}}}(\lambda_i(x_j) \neq y|\text{NN}_{\hat{S}}(x) = x_j) \qquad (15)\\
&= \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y\sim\mathcal{P}_{\hat{S},r_i}, \\ y_j\sim\mathcal{P}_{\lambda_i}(\cdot|x_j)}}(\lambda_i(x_j) = y_j, y_j \neq y|\text{NN}_{\hat{S}}(x) = x_j) + \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y\sim\mathcal{P}_{\hat{S},r_i}, \\ y_j\sim\mathcal{P}_{\lambda_i}(\cdot|x_j)}}(\lambda_i(x_j) \neq y_j, y_j = y|\text{NN}_{\hat{S}}(x) = x_j)\\
&\leq \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0}, \\ y_j\sim\mathcal{P}_{\lambda_i}(\cdot|x_j)}}(\lambda_i(x_j) = y_j, \exists(x,y) \in \hat{S}_{r_i} : \text{NN}_{\hat{S}}(x) = x_j, y_j \neq y) + \Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0}, \\ y_j\sim\mathcal{P}_{\lambda_i}(\cdot|x_j)}}(\lambda_i(x_j) \neq y_j, \exists(x,y) \in \hat{S}_{r_i} : \text{NN}_{\hat{S}}(x) = x_j, y_j = y).
\end{aligned}$$

Next, we use the fact that these probabilities only depend on one $(x_j, y_j) \sim \mathcal{P}_{\lambda_i}$, and observe that $\text{NN}_{\hat{S}}(x) = x_j$ implies that $\rho(f(x), f(x')) \leq r_i$. Therefore, the expression in (15) satisfies

$$\begin{aligned}
\Pr_{\substack{\hat{S}\sim\mathcal{P}_{\lambda_i}^{n_0} \\ x,y,\sim\mathcal{P}_{\hat{S},r_i}}}(\lambda_i(x_j) \neq y|\text{NN}_{\hat{S}}(x) = x_j) \leq &\Pr_{x_j,y_j\sim\mathcal{P}_{\lambda_i}}(\lambda_i(x_j) = y_j, \exists(x,y) \in \mathcal{X}\backslash S : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y)\\
&+ \Pr_{x_j,y_j\sim\mathcal{P}_{\lambda_i}}(\lambda_i(x_j) \neq y_j, \exists(x,y) \in \mathcal{X}\backslash S : \rho(f(x_j), f(x)) \leq r_i, y_j = y).
\end{aligned}$$

The first probability on the RHS can be written as $\Pr_{x_j,y_j\sim\mathcal{P}_{\lambda_i}}(\lambda_i(x_j) = y_j|\exists(x,y) \in \mathcal{X}\backslash S : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) \Pr_{x_j,y_j\sim\mathcal{P}_{\lambda_i}}(\exists(x,y) \in \mathcal{X}\backslash S : \rho(f(x_j), f(x)) \leq r_i, y_j \neq y) \leq \frac{1+\beta_i}{2}M(r_i)$, and the second one is at most $\Pr_{x_j,y_j\sim\mathcal{P}_{\lambda_i}}(\lambda_i(x_j) \neq y_j) = \frac{1-a_i}{2}$. Therefore, putting this back into (14), $\varepsilon \leq \frac{1+\beta_i}{2}M(r_i) + \frac{1-a_i}{2}$. Since $\bar{a}_i(r_i) = 2(1-\varepsilon) - 1$, we now have our desired bound

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

□

Theorem 2. *Suppose that data follows the model in* (1)*. The irreducible error decreases by the following amount when using $\bar{\lambda}_i$ rather than λ_i in Algorithm 1:*

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

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

$$\begin{aligned}
H(y|\boldsymbol{\lambda}, x) - H(y|\bar{\boldsymbol{\lambda}}, x) &= \mathbb{E}_{x,y,\boldsymbol{\lambda}}\left[-\log \Pr(y|\boldsymbol{\lambda}, x)\right] + \mathbb{E}_{x,y,\bar{\boldsymbol{\lambda}}}\left[\log \Pr(y|\bar{\boldsymbol{\lambda}}, x)\right]\\
&= \mathbb{E}_{x,y,\lambda_{-i}}\left[\mathbb{E}_{\bar{\lambda}_i}\left[\log \frac{\Pr(\bar{\lambda}_i|x,y)\Pr(\lambda_{-i}|x,y)\Pr(y|x)}{\Pr(\bar{\lambda}_i, \lambda_{-i}|x)}\Big|x,y\right] - \mathbb{E}_{\lambda_i}\left[\log \frac{\Pr(\lambda_i|x,y)\Pr(\lambda_{-i}|x,y)\Pr(y|x)}{\Pr(\lambda_i, \lambda_{-i}|x)}\Big|x,y\right]\right].
\end{aligned} \tag{16}$$

$\Pr(\lambda_{-i}|x,y)$ and $\Pr(y|x)$ are the same when using $\bar{\lambda}_i$ versus λ_i, so (16) becomes

$$H(y|\boldsymbol{\lambda}, x) - H(y|\bar{\boldsymbol{\lambda}}, x) = \mathbb{E}_{x,y,\lambda_{-i}}\left[\mathbb{E}_{\bar{\lambda}_i}\left[\log \frac{\Pr(\bar{\lambda}_i|x,y)}{\Pr(\bar{\lambda}_i, \lambda_{-i}|x)}\Big|x,y\right] - \mathbb{E}_{\lambda_i}\left[\log \frac{\Pr(\lambda_i|x,y)}{\Pr(\lambda_i, \lambda_{-i}|x)}\Big|x,y\right]\right]. \tag{17}$$

When extending λ_i, there are three regions of interest in input space: $\lambda_i(x), \bar{\lambda}_i(x) \neq 0$; $\lambda_i(x) = 0, \bar{\lambda}_i(x) \neq 0$; and $\lambda_i(x) = \bar{\lambda}_i(x) = 0$. In the first region, $\bar{\lambda}_i$ has the exact same behavior as λ_i since λ_i has coverage over this region. Therefore, conditioning on $\lambda_i(x) \neq 0$, the expectation on the RHS of (17) is equal to 0. Similarly, in the third region where $\lambda_i(x) = \bar{\lambda}_i(x) = 0$, the extended and original labeler vote exactly the same, so the expectation on the RHS of (17) is again equal to 0. The primary region of interest are the points that previously had no signal from λ_i but now have signal from $\bar{\lambda}_i$. Then, (17) becomes

$$H(y|\boldsymbol{\lambda}, x) - H(y|\bar{\boldsymbol{\lambda}}, x) = p_i \mathbb{E}_{y,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\log \frac{\Pr(\bar{\lambda}_i|x,y)}{\Pr(\bar{\lambda}_i,\lambda_{-i}|x)} - \log \frac{\Pr(\lambda_i = 0|x,y)}{\Pr(\lambda_i = 0,\lambda_{-i}|x)}\right]. \tag{18}$$

We can write $\frac{\Pr(\lambda_i=0|x,y)}{\Pr(\lambda_i=0,\lambda_{-i}|x)} = \frac{\Pr(\lambda_i=0|x)}{\Pr(\lambda_i=0|x)\Pr(\lambda_{-i}|x)} = \frac{1}{\Pr(\lambda_{-i}|x)}$ by decomposing the denominator conditional on y and using Lemma 2. Using the chain rule on $\Pr(\bar{\lambda}_i, \lambda_{-i}|x) = \Pr(\bar{\lambda}_i|\lambda_{-i}, x)\Pr(\lambda_{-i}|x)$, (18) is now

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

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

$$\mathbb{E}_{y=1,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\log \frac{\Pr(\bar{\lambda}_i|x,y)}{\Pr(\bar{\lambda}_i|\lambda_{-i},x)}\right] = \tag{19}$$
$$\mathbb{E}\Bigg[\Pr(\bar{\lambda}_i = 1|y=1,x,\bar{\lambda}_i \neq 0)\log \frac{\Pr(\bar{\lambda}_i = 1|x,y=1)}{\Pr(\bar{\lambda}_i = 1|\lambda_{-i},x)} + \Pr(\bar{\lambda}_i = -1|y=1,x,\bar{\lambda}_i \neq 0)\log \frac{\Pr(\bar{\lambda}_i = -1|x,y=1)}{\Pr(\bar{\lambda}_i = -1|\lambda_{-i},x)}\Bigg].$$

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

$$\begin{aligned}\Pr(\bar{\lambda}_i = 1|\lambda_{-i},x) &= \Pr(\bar{\lambda}_i = 1|\lambda_{-i},x,y=1)\Pr(y=1|\lambda_{-i},x) + \Pr(\bar{\lambda}_i = 1|\lambda_{-i},x,y=-1)\Pr(y=-1|\lambda_{-i},x)\\ &= \alpha_i(x)p(x,\lambda_{-i}) + (1-\alpha_i(x))(1-p(x,\lambda_{-i})),\end{aligned}$$

where $p(x,\lambda_{-i})$ is shorthand for $\Pr(y = 1|\lambda_{-i},x)$ (importantly, it does not depend on $\bar{\lambda}_i$) and likewise for $\Pr(\bar{\lambda}_i = -1|\lambda_{-i},x) = \alpha_i(x)(1-p(x,\lambda_{-i})) + (1-\alpha_i(x))p(x,\lambda_{-i})$. Our expression from (19) is now

$$\begin{aligned}\mathbb{E}_{y=1,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\Bigg[&\alpha_i(x)\log \frac{\alpha_i(x)}{\alpha_i(x)p(x,\lambda_{-i}) + (1-\alpha_i(x))(1-p(x,\lambda_{-i}))}\\ &+ (1-\alpha_i(x))\log \frac{1-\alpha_i(x)}{\alpha_i(x)(1-p(x,\lambda_{-i})) + (1-\alpha_i(x))p(x,\lambda_{-i})}\Bigg].\end{aligned} \tag{20}$$

Note that the expression inside the expectation is convex in both $\alpha_i(x)$ and $p(x,\lambda_{-i})$. Define $\alpha_{i,1} = \mathbb{E}_{y=1,\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}[\alpha_i(x)]$ to be the expected accuracy probability over the extended region when $y = 1$, and $p_{\lambda_{-i},1} = \mathbb{E}_{y'=1,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}[\Pr(y = y'|x,\lambda_{-i})]$ to be the expected label model performance using just λ_{-i} over the extended region when $y = 1$. Then, this expression from (20) is at least

$$\begin{aligned}&\alpha_{i,1}\log \frac{\alpha_{i,1}}{\alpha_{i,1}p_{\lambda_{-i},1} + (1-\alpha_{i,1})(1-p_{\lambda_{-i},1})}\\ &+ (1-\alpha_{i,1})\log \frac{1-\alpha_{i,1}}{\alpha_{i,1}(1-p_{\lambda_{-i},1}) + (1-\alpha_{i,1})p_{\lambda_{-i},1}}.\end{aligned} \tag{21}$$

We look at the case where $y = -1$. Similarly, we get

$$\begin{aligned}\mathbb{E}_{y=-1,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\Bigg[&\alpha_i(x)\log \frac{\alpha_i(x)}{\alpha_i(x)(1-p(x,\lambda_{-i})) + (1-\alpha_i(x))p(x,\lambda_{-i})}\\ &+ (1-\alpha_i(x))\log \frac{1-\alpha_i(x)}{\alpha_i(x)p(x,\lambda_{-i}) + (1-\alpha_i(x))(1-p(x,\lambda_{-i}))}\Bigg].\end{aligned} \tag{22}$$

Again, define $\alpha_{i,-1} = \mathbb{E}_{y=-1,\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\alpha_i(x)\right]$ and $p_{\lambda_{-i},-1} = \mathbb{E}_{y'=-1,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\Pr(y=y'|x,\lambda_{-i})\right]$, and by Jensen's inequality we have that (22) is at least

$$\begin{aligned}&\alpha_{i,-1}\log\frac{\alpha_{i,-1}}{\alpha_{i,-1}p_{\lambda_{-i},-1}+(1-\alpha_{i,-1})(1-p_{\lambda_{-i},-1})}\\&+(1-\alpha_{i,-1})\log\frac{1-\alpha_{i,-1}}{\alpha_{i,-1}(1-p_{\lambda_{-i},-1})+(1-\alpha_{i,-1})p_{\lambda_{-i},-1}}\end{aligned}\tag{23}$$

Therefore, $\mathbb{E}_{y,\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\log\frac{\Pr(\bar{\lambda}_i|x,y)}{\Pr(\bar{\lambda}_i|\lambda_{-i},x)}\right]$ is lower bounded by the weighted sum of $\Pr(y=1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)$ times (21) and $\Pr(y=-1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)$ times (23). Since (21) and (23) are convex in $\alpha_{i,1},p_{\lambda_{-i},1}$ and $\alpha_{i,-1},p_{\lambda_{-i},1}$ respectively, we can define $\alpha_i = \Pr(y=1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)\cdot\alpha_{i,1}+\Pr(y=-1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)\cdot\alpha_{i,-1} = \mathbb{E}_{\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\alpha_i(x)\right]$ as a notion of $\bar{\lambda}_i$'s accuracy in the region where we extend λ_i. We also define $p_{\lambda_{-i}} = \Pr(y=1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)\cdot p_{\lambda_{-i},1}+\Pr(y=-1|\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0)\cdot p_{\lambda_{-i},-1} = \mathbb{E}_{y',\lambda_{-i},\bar{\lambda}_i(x)\neq 0,\lambda_i(x)=0}\left[\Pr(y=y'|\lambda_{-i},x\right]$ as the label model's probability of outputting the correct label in our region of interest when relying on only λ_{-i}. Then, we have that

$$\begin{aligned}H(y|\boldsymbol{\lambda},x)-H(y|\bar{\boldsymbol{\lambda}},x)\geq p_i\Bigg(&\alpha_i\log\frac{\alpha_i}{\alpha_i p_{\lambda_{-i}}+(1-\alpha_i)(1-p_{\lambda_{-i}})}\\&+(1-\alpha_i)\log\frac{1-\alpha_i}{(1-\alpha_i)p_{\lambda_{-i}}+\alpha_i(1-p_{\lambda_{-i}})}\Bigg)\end{aligned}$$

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

$$H(y|\boldsymbol{\lambda},x)-H(y|\bar{\boldsymbol{\lambda}},x)\geq 8p_i(1-p_{\lambda_{-i}})^2\Big(\alpha_i-\frac{1}{2}\Big)^2.$$

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

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

E EXPERIMENTAL DETAILS

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

E.1 DATASET DETAILS

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).

[1] https://www.snorkel.org/use-cases/01-spam-tutorial

Task (Embedding)	T	m/T	**Prop**	N_{train}	N_{dev}	N_{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_{train}: The size of the unlabeled training set. N_{dev}: The size of the labeled dev set. N_{test}: The size of the held-out test set.

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.

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.

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 LF's 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 LF's 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.

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

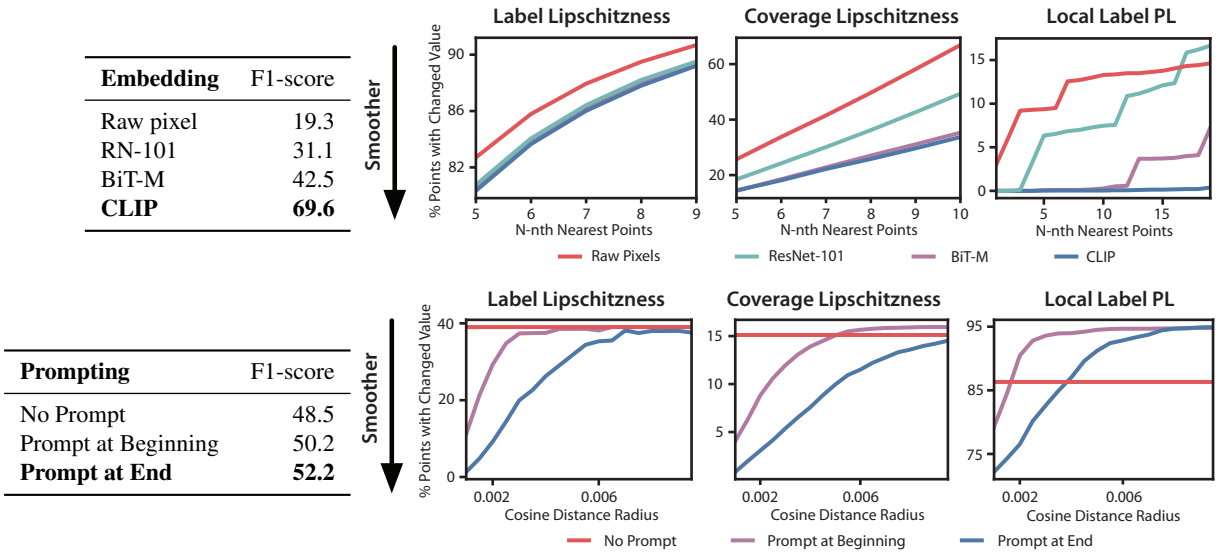


Figure 3: Top: LIGER performance and smoothness measurements of CLIP, BiT-M, ResNet-101, and raw pixels as embeddings for **Basketball**. Bottom: LIGER performance and smoothness measurements of no prompting, prompting at beginning, and prompting at end in GPT-3 for **Spouse**.

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

$$\begin{aligned} |\Pr(y=1|x) - \Pr(y=1|x')| &= |\mathbb{E}\left[\mathbb{1}\left\{y=1|x\right\} - \mathbb{E}\left[\mathbb{1}\left\{y=1|x'\right\}\right]\right] \\ &\leq \mathbb{E}\left[|\mathbb{1}\left\{y=1|x\right\} - \mathbb{1}\left\{y=1|x'\right\}|\right] \\ &= \mathbb{E}\left[\mathbb{1}\left\{y \neq y'\right\}\right] = \Pr(y \neq y') \end{aligned}$$

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 $|\Pr(\lambda_i \neq 0|x) - \Pr(\lambda_i \neq 0|x')| \leq \Pr(\mathbb{1}\left\{\lambda_i(x) \neq 0\right\}! = \mathbb{1}\left\{\lambda_i(x) \neq 0\right\})$, 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 λ_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 differerent 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.

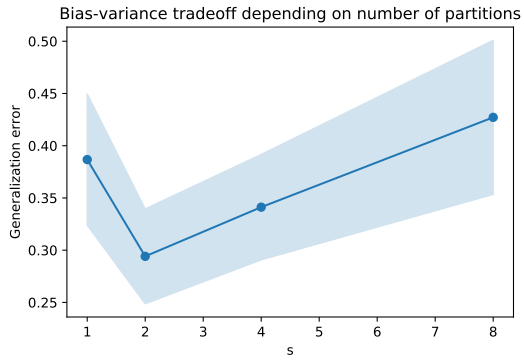


Figure 4: The bias-variance tradeoff in the generalization error based on s, the number of partitions used in our approach. When too few partitions are used, the accuracy estimates are not fine-grained and do not sufficiently approximate the true conditional distribution $\Pr(y, \boldsymbol{\lambda}|x)$, resulting in large bias. When too many partitions are used, the variance increases due to sampling error on individual partitions.

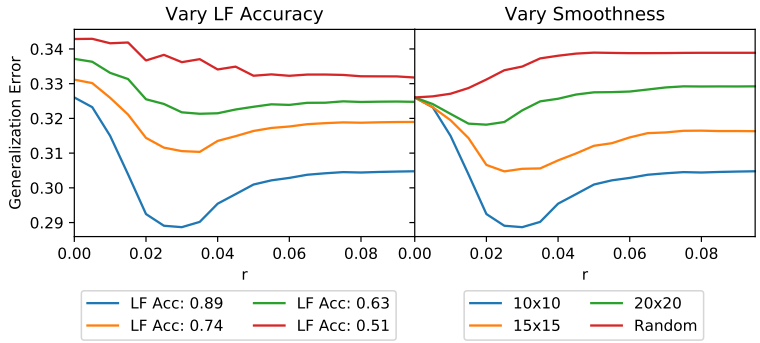


Figure 5: Reduction in generalization error from extending labeling functions of varying accuracies (left), and on embedding spaces of varying smoothness (right).

F.3 SYNTHETIC EXPERIMENTS

We evaluate LIGER on synthetic data to confirm our insights about 1) how generalization error for $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ 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 $\hat{\Pr}(y|\boldsymbol{\lambda}, x)$ (Theorem 1. We generate two sets of canonical parameters and use them in (1) to generate $(y, \boldsymbol{\lambda})$ from two different distributions, $\mathcal{P}_1$ and $\mathcal{P}_2$ over an embedding space. We generate 1000 points each for $\mathcal{P}_1$ and $\mathcal{P}_2$ to form datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, which are then concatenated to form a dataset $\mathcal{D}$ of 2000 points. We first run Algorithm 1 with $s = 1$, which means that we estimate only one set of parameters over $\mathcal{D}$ despite the dataset consisting of two different conditional distributions. We then set $s = 2$ and estimate the parameters of $\mathcal{P}_1$ and $\mathcal{P}_2$ separately over 1000 points each. Finally, we set $s = 4$ and $s = 8$ by dividing each of $\mathcal{D}_1$ and $\mathcal{D}_2$ 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 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 $\mathcal{D}_1$ and $\mathcal{D}_2$ 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 2). Suppose for simplicity that $s = 1$ and that $\Pr(y, \boldsymbol{\lambda})$ is modeled the same way as $\Pr(y, \boldsymbol{\lambda}|x)$ 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 λ_1 is extended. To understand the impact of a labeling function's accuracy, we fix a task distribution by assigning Y labels in a 10×10 "checkerboard" pattern and run our algorithm on four versions of λ_1 with varying average accuracies, keeping λ_1's support consistent. In Figure 5 (left), we extend λ_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 λ_1.