CHORUS: LEARNING RELIABLE SUPERVISION FROM NOISY LLM ANNOTATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

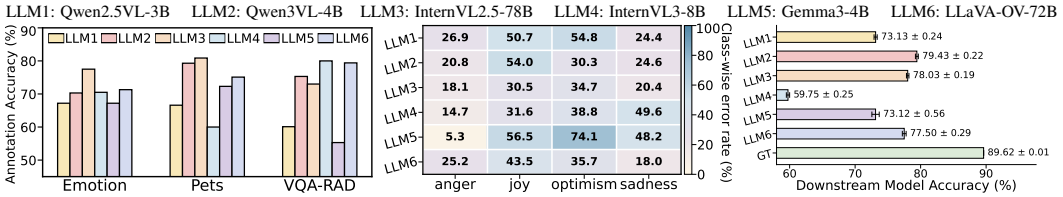
Large language models (LLMs) offer a scalable alternative to human annotation, but their reliability varies across models and tasks, and their errors are often class-dependent. Such heterogeneous errors make it difficult to obtain reliable supervision for downstream learning. To address this challenge, we propose Chorus, a closed-loop framework for learning reliable supervision from noisy and heterogeneous LLM annotations. It couples supervision estimation with downstream model training and uses the resulting evidence to guide subsequent LLM annotation. In Chorus, unlabeled data is processed in sequential annotation rounds. Within each round, a downstream-aware Bayesian estimation combines LLM annotations with task-model predictions, used as instance-specific label priors, to jointly infer latent labels and annotator-specific reliability. Across rounds, Chorus identifies recurring discrepancies between individual annotations and high-confidence inferred labels, and converts them into annotator-specific feedback for the next batch. In this way, Chorus operates through two coupled loops: supervision estimation and task-model training alternate within each round, while inferred supervision is converted into feedback that guides annotation across rounds. Extensive evaluations across five datasets show that Chorus improves annotation and downstream accuracy by up to 5.29% and 6.20%, respectively. With 3–8B annotators, it approaches clean-supervision performance on multiple datasets and remains competitive with methods using 72–78B annotators, while achieving further gains with stronger annotators.

1 INTRODUCTION

High-quality labeled data remains a central prerequisite for training effective supervised task models, but human annotation is expensive, time-consuming, and difficult to scale, especially when the target labels require fine-grained judgments or consistent interpretation of ambiguous samples (Song et al., 2022; Northcutt et al., 2021; Wang et al., 2021). While large language models (LLMs) have recently emerged as an attractive alternative for scalable data annotation, their annotation quality remains imperfect and highly heterogeneous across annotators and tasks (Lin et al., 2026; Tseng et al., 2025; Tan et al., 2024). Our motivation study reveals three characteristics of this heterogeneity. First, annotation quality varies substantially across LLM annotators and tasks, with larger models not uniformly outperforming smaller ones (Figure 1, left). Second, annotation errors are class-dependent: even stronger annotators, such as InternVL2.5-78B and LLaVA-OV-72B, exhibit higher error rates on particular classes than other annotators (Figure 1, middle). Such differences suggest opportunities for complementary supervision and highlight the limits of representing annotator reliability through a single aggregate score. Third, downstream performance varies substantially across LLM annotation sources, yet remains far below that achieved with ground-truth supervision (Figure 1, right). These observations highlight the downstream impact of annotation quality and motivate modeling class-dependent annotator reliability and exploiting complementary evidence to construct reliable supervision.

Existing approaches address annotation noise through different forms of reliability estimation and task learning. Traditional aggregation methods combine annotations through consensus or probabilistic annotator modeling. Majority Voting treats annotators equally, whereas Dawid-Skene, GLAD, and MACE jointly estimate latent labels and different aspects of annotator reliability (Dawid & Skene, 1979; Whitehill et al., 2009; Hovy et al., 2013). However, their standard formulations primarily rely on the annotation matrix without incorporating input-dependent task predictions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



	anger	joy	optimism	sadness
LLM1	26.9	50.7	54.8	24.4
LLM2	20.8	54.0	30.3	24.6
LLM3	18.1	30.5	34.7	20.4
LLM4	14.7	31.6	38.8	49.6
LLM5	5.3	56.5	74.1	48.2
LLM6	25.2	43.5	35.7	18.0

Figure 1: **Left**: Annotation quality varies across LLM annotators and tasks, with larger models not uniformly better. **Middle**: Class-dependent errors reveal complementary expertise beyond overall accuracy. **Right**: Downstream accuracy varies by annotation source and remains substantially below ground-truth supervision (GT).

Learning-from-crowds methods incorporate this evidence by jointly or alternately learning task predictors and annotator noise models (Rodrigues & Pereira, 2018; Chu et al., 2021; Khetan et al., 2017). However, they typically optimize task predictions and noise parameters discriminatively rather than using evolving predictions as instance-specific priors in posterior inference. In parallel, LLM-based approaches use semantic reasoning through model discussion or supervisor-guided instruction refinement (Du et al., 2023; Kim & Yoon, 2026), but lack an explicit statistical model of annotator-specific, class-dependent errors. Thus, existing methods do not jointly capture structured annotator reliability and evolving task-model evidence within a unified probabilistic formulation.

Beyond providing training labels for the downstream model, supervision inferred from multiple annotators can also serve as a reference for guiding subsequent annotation. This is particularly relevant to LLM annotators, whose predictions can be influenced through feedback (Kim & Yoon, 2026; Kim et al., 2026). Reliable supervision can provide better-informed feedback to annotators, while improved annotations can in turn provide stronger evidence for subsequent supervision estimation and task learning. This creates the potential for a closed loop between annotation, supervision estimation, and downstream learning. However, without reliable estimation, such feedback may instead reinforce annotation errors and propagate them across iterations. This leads to our research question:

How can we learn reliable supervision from noisy and heterogeneous LLM annotators and use the inferred evidence to improve subsequent annotation and downstream task learning?

In this paper, we propose Chorus, a closed-loop framework for learning reliable supervision from noisy and heterogeneous LLM annotators. Its central idea is to use inferred supervision for two purposes: training the downstream model and improving subsequent annotation. To this end, Chorus develops two interconnected modules: (i) **downstream-aware Bayesian supervision estimation**, which jointly infers latent labels and annotator-specific confusion matrices using LLM annotations and task-model evidence; and (ii) **feedback-guided annotation refinement**, which uses the estimated supervision to guide subsequent LLM annotation. Specifically, Chorus processes the data in sequential annotation batches. Within each round, downstream-aware Bayesian supervision estimation combines accumulated annotations with task-model evidence to infer soft supervision and annotator-specific reliability. The inferred supervision updates the downstream model, whose refined predictions provide additional evidence for subsequent estimation. Across rounds, feedback-guided annotation refinement converts recurring, high-confidence discrepancies into annotator-specific guidance for the next batch. Thus, the inner loop couples supervision estimation with downstream learning, while the outer loop couples annotation with feedback, allowing annotation quality, supervision estimation, and task learning to progressively reinforce one another. Our contributions are as follows:

- We propose Chorus, a closed-loop framework that formulates supervision construction from heterogeneous LLM annotators as a progressive process, using inferred supervision for both downstream training and annotator-specific feedback.
- We develop a downstream-aware Bayesian formulation that jointly infers latent labels and class-dependent annotator reliability, using evolving task-model predictions as instance-specific priors and posterior soft labels for model training.
- Extensive evaluations across five datasets show that Chorus improves annotation and downstream accuracy by up to 5.29% and 6.20%, respectively. With 3–8B annotators, it approaches clean-supervision performance and remains competitive with methods using 72–78B annotators, while benefiting further from stronger annotators.

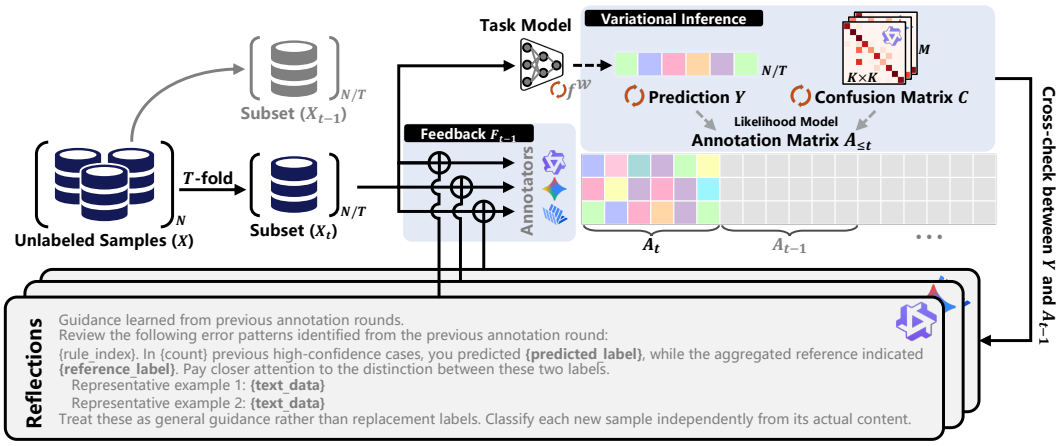


Figure 2: Chorus overview: Unlabeled samples are divided into annotation batches. In each annotation round t, heterogeneous LLM annotators label the current batch using feedback F_{t-1} from $t-1$ round. Variational inference combines their accumulated annotations with task-model predictions to estimate latent labels and annotator-specific confusion matrices, while the inferred labels supervise task-model training. High-confidence discrepancies between annotations and inferred labels are then summarized into annotator-specific feedback for the next round.

2 Chorus Design

2.1 Chorus Overview

Consider an unlabeled dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ with K classes and M LLM annotators. Each sample $\mathbf{x}_i$ has an unknown label $y_i \in \{1, \ldots, K\}$, and the annotation provided by annotator m is denoted by $A_{i,m}$. Chorus aims to train a downstream model $f^{\mathcal{W}}$ from these noisy LLM annotations without access to ground-truth labels. To support this objective, it estimates soft supervision from multiple annotators and uses the inferred evidence for task-model training and guiding subsequent annotation. Specifically, we partition $\mathbf{X}$ into T disjoint batches $\{\mathbf{X}_t\}_{t=1}^T$ and process them sequentially. At annotation round t, annotators label the current batch $\mathbf{X}_t$ guided by the task instructions and feedback F_{t-1} from the previous round. The first round uses no feedback. Newly collected annotations are added to those from previous rounds, yielding the cumulative dataset and annotation matrix: $\mathbf{X}_{\leq t} = \bigcup_{r=1}^t \mathbf{X}_r$ and $\mathbf{A}_{\leq t} \in \{1, \ldots, K\}^{N_t \times M}$, where $N_t = |\mathbf{X}_{\leq t}|$ and rows of $\mathbf{A}_{\leq t}$ correspond to samples in $\mathbf{X}_{\leq t}$. Supervision estimation and downstream training operate on this accumulated data, while feedback guides the next annotation batch. All notations are summarized in Appendix A.

Figure 2 illustrates the two interacting loops of Chorus: supervision estimation and task-model training within each annotation round, and feedback-guided annotation across rounds. The central idea is to use estimated supervision from variational inference both to train the task model and to guide subsequent annotation, through two complementary designs. **Downstream-aware Bayesian supervision estimation** accounts for annotator-specific, class-dependent errors while incorporating task-model evidence. Rather than treating aggregated labels as fixed targets, it alternates posterior inference with soft-label training: inferred supervision informs task learning, while updated task predictions inform subsequent supervision estimation. **Feedback-guided annotation refinement** extends this interaction across annotation rounds. It converts recurring disagreements with high-confidence inferred labels into annotator-specific guidance for new samples. These designs connect the estimation and use of supervision with the collection of future annotations.

2.2 Downstream-aware Bayesian Supervision Estimation

At annotation round t, Chorus estimates supervision from the accumulated inputs $\mathbf{X}_{\leq t}$ and annotations $\mathbf{A}_{\leq t}$. For notational simplicity, we write these as $\mathbf{X}$ and $\mathbf{A}$ in this subsection and also index the accumulated N_t samples as N. Within each annotation round, variational inference jointly estimates latent labels and annotator-specific confusion using downstream predictions as instance-specific priors. The inferred label distributions provide soft supervision to update the task model, whose revised predictions refine subsequent posterior inference.

2.2.1 PROBABILISTIC MODELING OF LLM ANNOTATION

For sample $\mathbf{x}_i$, unknown label y_i follows a categorical distribution parameterized by the task model:

$$y_i \sim \mathrm{Cat}(\boldsymbol{\pi}_i), \qquad \boldsymbol{\pi}_i = \mathrm{softmax}\left(f^{\mathcal{W}}(\mathbf{x}_i)\right), \tag{1}$$

where $f^{\mathcal{W}}$ outputs class logits and $\boldsymbol{\pi}_i \in [0,1]^K$ denotes the label distribution. The prior is instance-specific and evolves as the task model is updated.

We model the reliability of annotator m with a class-dependent confusion matrix $\mathbf{C}^{(m)} \in [0,1]^{K\times K}$, where $\mathbf{C}^{(m)}_{k,l}$ is the probability of annotating label l given latent label k, with $\sum_{l=1}^{K} \mathbf{C}^{(m)}_{k,l} = 1$. To quantify the uncertainty in annotator confusion, we place independent Dirichlet priors over its rows:

$$\mathbf{C}^{(m)}_{k,:} \sim \mathrm{Dir}(\boldsymbol{\alpha}^{(m)}_k), \quad k = 1, \ldots, K, \tag{2}$$

where $\boldsymbol{\alpha}^{(m)}_k$ denotes the prior hyperparameters for annotator m when the latent label is k. At each annotation round, its confusion-matrix posterior is re-estimated from the accumulated annotations.

Annotation likelihood. We now define the likelihood of LLM m annotating data sample $\mathbf{x}_i$ as the probability conditioned on the LLM reliability matrix $\mathbf{C}^{(m)}$ and the latent label y_i:

$$p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}) = \prod_{l=1}^{K} (\mathbf{C}^{(m)}_{y_i,l})^{\mathbb{I}[A_{i,m}=l]}, \tag{3}$$

where $\mathbb{I}[\cdot]$ is an indicator function returning 1 if the statement is true and 0 otherwise. This likelihood models how each LLM annotator generates a potentially noisy label conditioned on the latent label. Figure 3 summarizes the resulting dependencies. Learning jointly estimates the task-model parameters $\mathcal{W}$, LLM-specific reliability parameters $\{\mathbf{C}^{(m)}\}_{m\in\mathcal{M}}$, and posterior inference over the latent labels y_i.

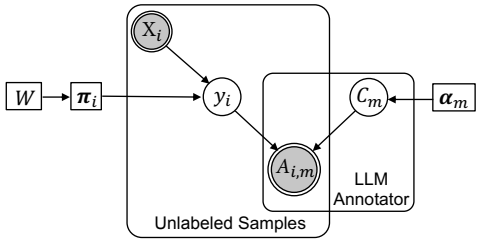


Figure 3: Graphical model for supervision estimation. Circles denote random variables: single-circled nodes indicate hidden variables, while doubly (greyed) circled nodes indicate observed variables. Squares represent learnable parameters of the probabilistic model. Edges denote conditional dependencies when generating LLM annotations.

For each sample $\mathbf{x}_i$, a set of LLM annotators $\mathcal{M}$ provides noisy annotations. The likelihood of the observed annotations is given by:

$$p(\{A_{i,m}\}_{m\in\mathcal{M}} \mid y_i, \{\mathbf{C}^{(m)}\}_{m\in\mathcal{M}}) = \prod_{m\in\mathcal{M}} p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}). \tag{4}$$

Since latent labels and annotator reliability are unknown, we next formulate Chorus under a unified likelihood objective, where reliable soft supervision is inferred jointly with task-model updating.

2.2.2 VARIATIONAL POSTERIOR INFERENCE

We perform posterior inference by maximizing the marginal likelihood of the observed annotations. Conditioned on the input samples $\mathbf{X}$, the likelihood is obtained by marginalizing over the latent labels $\mathbf{Y}$ and annotator reliability matrices $\mathbf{C}$:

$$p(\mathbf{A} \mid \mathbf{X}; W) = \sum_{\mathbf{Y}} \int p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; W)\, d\mathbf{C}. \tag{5}$$

Equation 5 consists of an integral with two latent variables, rendering it computationally infeasible to optimize (Tzikas et al., 2008). We therefore introduce a variational distribution $q(\mathbf{Y}, \mathbf{C})$ and decompose the log marginal likelihood as

$$\log p(\mathbf{A} \mid \mathbf{X}; \mathcal{W}) = \underbrace{\sum_{\mathbf{Y}} \int q(\mathbf{Y}, \mathbf{C}) \log \frac{p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W})}{q(\mathbf{Y}, \mathbf{C})}\, d\mathbf{C}}_{\mathcal{L}(\mathcal{W}, q)} + \underbrace{\sum_{\mathbf{Y}} \int q(\mathbf{Y}, \mathbf{C}) \log \frac{q(\mathbf{Y}, \mathbf{C})}{p(\mathbf{Y}, \mathbf{C} \mid \mathbf{A}, \mathbf{X}; \mathcal{W})}\, d\mathbf{C}}_{\mathrm{KL}(q \| p_{\mathcal{W}})} \tag{6}$$

where $\mathcal{L}(\mathcal{W}, q)$ is the evidence lower bound (ELBO) and $KL(\cdot)$ is the KL divergence between two distributions (Blei et al., 2017). We optimize the two parts of the objective function using variational expectation-maximization (Tzikas et al., 2008). Specifically, we iterate between two steps: 1) the E-step where we approximate the distribution of latent variables $p(\mathbf{Y}, \mathbf{C} \mid \mathbf{A}, \mathbf{X}; \mathcal{W})$ with the variational distribution $q(\mathbf{Y}, \mathbf{C})$, by minimizing the KL-divergence, and 2) the M-step where we update the estimate of the parameters of the task model $\mathcal{W}$ by maximizing the ELBO (Blei et al., 2017) under the current variational posterior.

E-step. We use the mean-field variational inference (Blei et al., 2017) by assuming that $q(\mathbf{Y}, \mathbf{C})$ factorizes over the latent variables:

$$q(\mathbf{Y}, \mathbf{C}) = \prod_{i=1}^{N} q(y_i) \prod_{m=1}^{M} \prod_{k=1}^{K} q(\mathbf{C}_{k,:}^{(m)}). \tag{7}$$

We further assume the following forms for variational factors:

$$q(y_i) = \mathrm{Cat}(\boldsymbol{\phi}_i), \quad q(\mathbf{C}_{k,:}^{(m)}) = \mathrm{Dir}(\boldsymbol{\lambda}_k^{(m)}), \tag{8}$$

where $\boldsymbol{\phi}_i$ and $\boldsymbol{\lambda}_k^{(m)}$ are variational parameters. Specifically, $\boldsymbol{\phi}_i$ represents the posterior label distribution of sample $\mathbf{x}_i$, while $\boldsymbol{\lambda}_k^{(m)}$ parameterizes the posterior over the k-th row of annotator m's confusion matrix. These factors are optimized by coordinate ascent to minimize the KL-divergence, alternating between the label and annotator-reliability posteriors.

We first derive the update rule for $q(y_i)$. Given the current task-model prior $p(y_i \mid \mathbf{x}_i; \mathcal{W})$ and the annotator-reliability posteriors, the optimal variational factor satisfies

$$q(y_i) \propto p(y_i \mid \mathbf{x}_i; \mathcal{W}) \prod_{m \in \mathcal{M}} \exp\{g_{\mathbf{C}^{(m)}}(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))\}. \tag{9}$$

$g_z(\cdot)$ refers to the expectation term $\mathbb{E}_z[log(\cdot)]$ with z being a variational distribution. We can derive the update rule for $q(y_i)$ in Equation 11, by combining the instance-specific prior from the downstream model with the annotation evidence weighted by the inferred reliability of each annotator. Similarly, for the annotator-reliability posterior, keeping only the terms that depend on $\mathbf{C}^{(m)}$ gives

$$q(\mathbf{C}^{(m)}) \propto p(\mathbf{C}^{(m)}) \prod_{i=1}^{N} \exp\{g_{\phi_i}(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))\}, \tag{10}$$

where $p(\mathbf{C}^{(m)})$ denotes Dirichlet prior over annotator m's confusion matrix and $\boldsymbol{\phi}_i$ is the label posterior distribution from current iteration. We can derive the update rule for $q(\mathbf{C}^{(m)})$ in Equation 12.

Theorem 2.1 (Label Posterior Update) *The posterior label distribution $q(y_i)$ of sample $\mathbf{x}_i$ can be updated from the task-model prior $\boldsymbol{\pi}_i$ and the annotator-reliability posterior $\{\boldsymbol{\lambda}_k^{(m)}\}_{m \in \mathcal{M}}$:*

$$q(y_i = k) \propto \boldsymbol{\pi}_{i,k} \prod_{m \in \mathcal{M}} \exp\{\Psi(\boldsymbol{\lambda}_{k,A_{i,m}}^{(m)}) - \Psi(\sum_{l=1}^{K} \boldsymbol{\lambda}_{k,l}^{(m)})\}, \tag{11}$$

where $\Psi(\cdot)$ is the Digamma function. The resulting posterior $q(y_i)$ provides a soft supervision distribution over the K classes.

Theorem 2.2 (Annotator Reliability Posterior Update) *The posterior reliability distribution $q(\mathbf{C}_{k,:}^{(m)})$ can be updated from Dirichlet prior $\boldsymbol{\alpha}_k^{(m)}$ and current posterior label probabilities $\boldsymbol{\phi}_{i,k}$:*

$$q(\mathbf{C}_{k,:}^{(m)}) = \mathrm{Dir}(\mathbf{C}_{k,:}^{(m)} \mid \boldsymbol{\lambda}_{k,:}^{(m)}), \quad \boldsymbol{\lambda}_{k,l}^{(m)} = \boldsymbol{\alpha}_{k,l}^{(m)} + \sum_{i=1}^{N} \boldsymbol{\phi}_{i,k} \mathbb{I}[A_{i,m} = l], \quad l = 1, \ldots, K. \tag{12}$$

Detailed proof of Theorem 2.1 and Theorem 2.2 can be found in Appendix B.

M-step. Given the variational posterior inferred in the E-step, the M-step fixes $q(\mathbf{Y}, \mathbf{C})$ and maximizes the ELBO with respect to the task-model parameters $\mathcal{W}$. Since the annotation likelihood and annotator-reliability terms are independent of $\mathcal{W}$, the relevant part of the ELBO reduces to

$$\begin{aligned} \mathcal{L}(\mathcal{W}, q) &= \sum_{y_i} \int q(y_i, \mathbf{C}^{(m)}) \log[p(A_{i,m}|y_i, \mathbf{C}^{(m)}) p(y_i|\mathbf{x}_i; \mathcal{W})] d\mathbf{C}^{(m)} + const \\ &= \underbrace{\sum_{y_i} \int q(y_i, \mathbf{C}^{(m)}) \log p(A_{i,m}|y_i, \mathbf{C}^{(m)}) d\mathbf{C}^{(m)}}_{\mathcal{M}_1} + \underbrace{\sum_{y_i} q(y_i) \log p(y_i|\mathbf{X}_i; \mathcal{W})}_{\mathcal{M}_2} + const, \end{aligned} \tag{13}$$

where $const = \mathbb{E}_{q(y_i, \mathbf{C}^{(m)})}[\log \frac{1}{q(y_i, \mathbf{C}^{(m)})}]$ is a constant. From the derived equation, only $\mathcal{M}_2$ depends on the task model's parameters $\mathcal{W}$ that the M-step aims to learn. Obviously, $\mathcal{M}_2$ is the negative cross-entropy between $q(y_i)$ and $p(y_i|\mathbf{x}_i; \mathcal{W})$, which is used as the loss function for downstream task model (Nielsen, 2015). The updated task model then provides refreshed instance-specific label priors for the next E-step, coupling supervision estimation with downstream learning.

2.3 Feedback-guided Annotation Refinement

The inferred supervision supports both downstream training and subsequent LLM annotation. After the estimation–training updates in round t, Chorus cross-checks each annotator's predictions against inferred label posteriors and summarizes recurring, high-confidence discrepancies into annotator-specific feedback. Although posterior inference uses all accumulated annotations, feedback is constructed from the current batch $\mathbf{X}_t$, reflecting each annotator's behavior under its latest instructions.

Let $\boldsymbol{\phi}_{i,k}^{(t)}$ denote the inferred posterior probability of class k at the end of round t. For each sample in the current batch, we define its reference label and confidence as

$$\hat{y}_i^{(t)} = \arg\max_k \boldsymbol{\phi}_{i,k}^{(t)}, \qquad c_i^{(t)} = \max_k \boldsymbol{\phi}_{i,k}^{(t)}. \tag{14}$$

We retain discrepancies between annotator m's prediction and the reference label only when $c_i^{(t)} \geq \tau$, where τ is a confidence threshold, ensuring that feedback is derived from strong posterior support. Retained discrepancies are grouped by the annotator prediction l and inferred reference label k:

$$\mathcal{S}_{t,l\to k}^{(m)} = \left\{ i : \mathbf{x}_i \in \mathbf{X}_t,\ A_{i,m} = l,\ \hat{y}_i^{(t)} = k,\ c_i^{(t)} \geq \tau \right\}, \qquad l \neq k. \tag{15}$$

For each discrepancy pair, we compute its occurrence count as $n_{t,l\to k}^{(m)} = \left|\mathcal{S}_{t,l\to k}^{(m)}\right|$ and retain up to the R most frequent pairs for each annotator, prioritizing recurring discrepancies while keeping feedback concise. Each selected pair is then converted into a textual instruction that identifies confused classes, summarizes the observed discrepancy, and encourages closer attention to their distinction; representative examples may also be included when textual content is available. These instructions form annotator-specific feedback $F_t^{(m)}$, which is appended to annotator m's task prompt when labeling $\mathbf{X}_{t+1}$. The annotator is instructed to treat the feedback as contextual guidance while classifying each new sample based on its own content. The resulting annotations are then incorporated into subsequent posterior inference and downstream training, completing the cross-round feedback loop.

Algorithm 1 summarizes the overall procedure of Chorus. At each round t, LLM annotators label the current batch $\mathbf{X}_t$ using feedback from the previous round, and the resulting annotations are added to the accumulated annotation matrix $\mathbf{A}_{\leq t}$. Chorus then alternates between variational posterior inference and downstream task-model training over the accumulated data (VEM$(\cdot)$). After convergence, recurring, high-confidence discrepancies on the current batch are summarized into annotator-specific feedback for the next round. The detailed variational EM updates are provided in Appendix D.

Algorithm 1: Chorus

Input : Unlabeled data batches $\{\mathbf{X}_t\}_{t=1}^{T}$;
LLM annotators $\{m\}_{m=1}^{M}$
Output : Trained task model $f^{\mathcal{W}}$
Initialize: $\mathbf{A}_{\leq 0} \leftarrow \emptyset$; $F_0 \leftarrow \emptyset$

for $t = 1, \ldots, T$ **do**
  $\mathbf{A}_t \leftarrow \text{Annotate}(\mathbf{X}_t, F_{t-1})$
  $\mathbf{A}_{\leq t} \leftarrow \mathbf{A}_{\leq t-1} \cup \mathbf{A}_t$;
  /* Supervision estimation */
  $(\boldsymbol{\Phi}_t, \mathbf{C}_t, \mathcal{W}) \leftarrow \text{VEM}(\mathbf{X}_{\leq t}, \mathbf{A}_{\leq t}, \mathcal{W})$;
  /* Feedback Construction */
  $F_t \leftarrow \text{Feedback}(\mathbf{X}_t, \mathbf{A}_t, \boldsymbol{\Phi}_t)$;
return $f^{\mathcal{W}}$

3 Experiments

3.1 Experimental Setup

Datasets and Downstream Models. Since LLMs are capable of handling multiple data types, we consider a comprehensive variety of annotation tasks, including image classification: FashionMNIST (Xiao et al., 2017) and Pets (Parkhi et al., 2012); text classification: Emotion and Irony (Barbieri et al., 2020); visual question-answering (VQA): closed-set VQA-RAD (Lau et al., 2018). We employ three pretrained models for downstream evaluation. We use ImageNet-1K-pretrained ResNet-18 (He et al., 2016; Russakovsky et al., 2015) for image classification, RoBERTa-base (Liu et al., 2019) for text classification, BLIP-VQA-base (Li et al., 2022) for VQA.

LLM annotators. We utilize seven open-source LLMs as annotators throughout our experiments. We first evaluate Chorus with four relatively compact models from different families: Qwen2.5VL-3B (Bai et al., 2025b), Qwen3VL-4B (Bai et al., 2025a), InternVL3-8B (Zhu et al., 2025), and Gemma3-4B (Team et al., 2025), examining its ability to learn reliable supervision from noisy and heterogeneous annotations. As larger models generally exhibit stronger critique capabilities (Song et al., 2025), we further evaluate LLaVA-OneVision-72B (LLaVA-OV-72B) (Li et al., 2024), Qwen2.5VL-72B (Bai et al., 2025b), and InternVL2.5-78B (Chen et al., 2024) to assess whether Chorus's benefits persist with more capable annotators.

Table 1: Annotation accuracy (Ann.) and downstream test accuracy (Down.) across five datasets, both reported in %. The annotator pool comprises Qwen2.5VL-3B, Qwen3VL-4B, InternVL3-8B, and Gemma3-4B. Results are the mean ± standard deviation over three runs. **Bold** values are the best results within each column. Under clean supervision (training with ground-truth labels), downstream accuracies are 82.62%, 74.11%, 94.21%, 89.62%, and 69.32%, respectively.

Method	Emotion		Irony		FashionMNIST		Pets		VQA-RAD	
	Ann.	Down.	Ann.	Down.	Ann.	Down.	Ann.	Down.	Ann.	Down.
MV	$73.74_{\pm 0.46}$	$77.71_{\pm 0.87}$	$57.72_{\pm 0.52}$	$59.72_{\pm 1.10}$	$62.38_{\pm 0.06}$	$68.03_{\pm 0.90}$	$82.38_{\pm 0.11}$	$81.57_{\pm 0.31}$	$75.46_{\pm 0.27}$	$62.02_{\pm 0.61}$
DS	$75.39_{\pm 0.07}$	$79.06_{\pm 0.51}$	$57.74_{\pm 0.52}$	$59.72_{\pm 1.08}$	$73.43_{\pm 0.05}$	$78.64_{\pm 0.88}$	$86.68_{\pm 0.23}$	$83.20_{\pm 0.15}$	$76.95_{\pm 0.27}$	$62.82_{\pm 0.83}$
OneCoinDS	$73.04_{\pm 0.49}$	$76.92_{\pm 0.71}$	$51.83_{\pm 0.07}$	$41.56_{\pm 0.39}$	$67.25_{\pm 0.10}$	$73.47_{\pm 2.27}$	$82.51_{\pm 0.13}$	$81.18_{\pm 0.10}$	$74.93_{\pm 0.33}$	$62.35_{\pm 1.04}$
GLAD	$74.21_{\pm 0.47}$	$78.06_{\pm 0.97}$	$56.51_{\pm 0.25}$	$57.55_{\pm 1.68}$	$72.59_{\pm 0.09}$	$77.51_{\pm 0.25}$	$82.95_{\pm 0.30}$	$81.51_{\pm 0.25}$	$77.71_{\pm 0.69}$	$64.01_{\pm 1.02}$
MACE	$74.68_{\pm 0.82}$	$77.92_{\pm 1.04}$	$56.11_{\pm 2.09}$	$60.78_{\pm 3.62}$	$66.23_{\pm 0.18}$	$73.54_{\pm 0.10}$	$83.18_{\pm 0.38}$	$81.48_{\pm 0.75}$	$62.87_{\pm 0.85}$	$58.70_{\pm 1.14}$
CrowdLayer	$76.39_{\pm 0.49}$	$78.65_{\pm 0.46}$	$57.20_{\pm 0.63}$	$56.20_{\pm 2.49}$	$73.63_{\pm 1.10}$	$73.62_{\pm 0.96}$	$85.34_{\pm 0.03}$	$82.30_{\pm 0.06}$	$67.02_{\pm 0.80}$	$60.03_{\pm 0.30}$
CoNAL	$76.69_{\pm 0.96}$	$78.52_{\pm 0.36}$	$56.65_{\pm 0.55}$	$55.05_{\pm 1.46}$	$74.18_{\pm 0.50}$	$73.73_{\pm 0.22}$	$83.89_{\pm 0.25}$	$81.87_{\pm 0.27}$	$68.90_{\pm 1.94}$	$61.35_{\pm 1.05}$
MAD	$74.45_{\pm 0.45}$	$78.59_{\pm 0.35}$	$58.70_{\pm 0.42}$	$63.39_{\pm 1.21}$	$70.90_{\pm 0.16}$	$76.39_{\pm 0.21}$	$82.12_{\pm 0.23}$	$81.68_{\pm 0.41}$	$71.17_{\pm 0.60}$	$62.54_{\pm 0.91}$
DiZiNER	$70.71_{\pm 0.36}$	$76.45_{\pm 0.52}$	$56.06_{\pm 0.46}$	$61.27_{\pm 2.03}$	$65.44_{\pm 0.26}$	$72.07_{\pm 0.39}$	$69.38_{\pm 0.28}$	$71.74_{\pm 0.31}$	$72.30_{\pm 0.45}$	$60.16_{\pm 1.22}$
Chorus	$\mathbf{79.84}_{\pm 0.48}$	$\mathbf{80.09}_{\pm 0.19}$	$\mathbf{60.94}_{\pm 0.57}$	$\mathbf{69.59}_{\pm 1.48}$	$\mathbf{79.47}_{\pm 0.18}$	$\mathbf{81.13}_{\pm 0.55}$	$\mathbf{88.72}_{\pm 0.59}$	$\mathbf{86.80}_{\pm 0.25}$	$\mathbf{81.12}_{\pm 0.55}$	$\mathbf{66.04}_{\pm 0.14}$

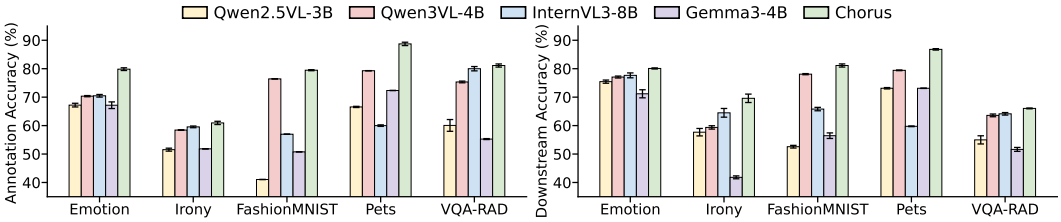


Figure 4: Annotation accuracy (left) and downstream accuracy (right) using individual LLM annotators and Chorus across five datasets.

Baselines and Metrics. We compare Chorus against nine baselines spanning three categories: (1) probabilistic methods, including Majority Voting (MV), Dawid–Skene (DS), OneCoinDS, GLAD, and MACE; (2) learning-from-crowds methods, including CrowdLayer and CoNAL; and (3) LLM-based annotation refinement methods, including Multi-Agent Debate (MAD) and a task-adapted version of DiZiNER. We also include downstream models trained on individual LLM annotations and a clean-supervision reference trained on ground-truth labels. We report annotation accuracy on the unlabeled datasets and downstream accuracy on the held-out test set. For Chorus, inferred labels are obtained by taking the highest-probability posterior class. Each experiment is repeated three times, and we report the mean and standard deviation.

Implementation details. For each dataset, all compared methods use the same downstream architecture, with the total number of downstream optimization steps matched between Chorus and the baselines. For all LLM annotators, we use a top-p of 0.9, temperature of 0.7, top-k of 50, and a maximum of 250 new tokens when applicable. Unless otherwise specified, we use five annotation batches and set the feedback confidence threshold to $\tau = 0.8$. For each annotator, we retain up to three recurring discrepancy pairs, each observed at least five times in the current batch. More experimental details can be found in Appendix C.

3.2 Performance Evaluation

Chorus Effectiveness. Table 1 reports annotation and downstream accuracy with four relatively small LLM annotators (3–8B parameters). Despite their imperfect individual annotations, Chorus achieves the highest mean accuracy on both metrics across all five datasets. Compared with the strongest reported baseline for each metric and dataset, it improves annotation accuracy by 2.04%–5.29% and downstream accuracy by 1.03%–6.20%. Notably, downstream accuracy approaches clean supervision's performance on Emotion and Pets. Results show that relatively small annotators can provide effective training supervision when their noisy outputs are appropriately integrated.

Baseline performance varies across tasks: DS achieves the highest downstream accuracy on Emotion, FashionMNIST, and Pets, MAD on Irony, and GLAD on VQA-RAD. MV treats annotators equally, whereas probabilistic aggregation methods model annotator reliability but do not incorporate input-dependent task-model predictions. CrowdLayer and CoNAL also incorporate downstream learning but underperform Chorus, highlighting the importance of how task-model evidence informs supervision estimation. MAD and the adapted DiZiNER baseline use semantic interaction and instruction refinement, respectively, but do not explicitly infer posterior distributions over latent labels

Table 2: Performance with larger annotators across five datasets. The annotator pool comprises LLaVA-OV-72B, Qwen2.5VL-72B, and InternVL2.5-78B. Ann. and Down. denote annotation accuracy and downstream test accuracy (%), respectively. Single-annotator rows use labels from the indicated model, while aggregation methods and Chorus use all three annotators.

Method	Emotion		Irony		FashionMNIST		Pets		VQA-RAD	
	Ann.	Down.	Ann.	Down.	Ann.	Down.	Ann.	Down.	Ann.	Down.
LLaVA-OV	$71.34_{\pm 0.29}$	$77.53_{\pm 0.47}$	$63.14_{\pm 0.76}$	$61.05_{\pm 1.20}$	$68.91_{\pm 0.08}$	$76.36_{\pm 0.25}$	$75.13_{\pm 0.35}$	$77.50_{\pm 0.29}$	$79.40_{\pm 0.68}$	$66.73_{\pm 0.59}$
Qwen2.5VL	$76.62_{\pm 1.05}$	$79.64_{\pm 0.68}$	$64.98_{\pm 0.43}$	$67.58_{\pm 3.04}$	$71.25_{\pm 0.15}$	$77.96_{\pm 0.04}$	$86.61_{\pm 0.30}$	$84.67_{\pm 0.33}$	$81.15_{\pm 0.84}$	$65.18_{\pm 1.46}$
InternVL2.5	$77.50_{\pm 0.44}$	$79.25_{\pm 0.18}$	$63.59_{\pm 0.10}$	$63.48_{\pm 0.22}$	$75.28_{\pm 0.05}$	$81.75_{\pm 0.35}$	$80.87_{\pm 0.44}$	$78.03_{\pm 0.19}$	$73.01_{\pm 0.66}$	$64.28_{\pm 1.79}$
DS	$77.41_{\pm 0.09}$	$79.45_{\pm 0.39}$	$64.15_{\pm 0.39}$	$63.56_{\pm 3.14}$	$78.90_{\pm 0.10}$	$84.04_{\pm 0.59}$	$88.73_{\pm 0.35}$	$86.30_{\pm 0.46}$	$81.22_{\pm 1.28}$	$66.53_{\pm 0.56}$
MACE	$78.80_{\pm 0.50}$	$79.57_{\pm 0.21}$	$63.48_{\pm 0.34}$	$62.80_{\pm 1.86}$	$75.84_{\pm 0.08}$	$82.46_{\pm 0.06}$	$87.09_{\pm 0.32}$	$84.92_{\pm 0.33}$	$81.70_{\pm 1.23}$	$65.74_{\pm 3.19}$
CrowdLayer	$78.39_{\pm 0.24}$	$79.46_{\pm 0.11}$	$64.64_{\pm 0.23}$	$64.30_{\pm 0.81}$	$78.72_{\pm 0.83}$	$83.38_{\pm 0.12}$	$86.98_{\pm 0.57}$	$84.33_{\pm 0.23}$	$75.74_{\pm 1.50}$	$63.61_{\pm 1.51}$
CoNAL	$78.79_{\pm 0.12}$	$79.15_{\pm 0.51}$	$64.57_{\pm 0.28}$	$62.76_{\pm 1.43}$	$78.09_{\pm 0.95}$	$82.90_{\pm 0.04}$	$87.66_{\pm 0.17}$	$85.05_{\pm 0.25}$	$78.30_{\pm 1.21}$	$65.21_{\pm 2.65}$
Chorus	$\mathbf{79.50}_{\pm 0.40}$	$\mathbf{80.17}_{\pm 0.10}$	$\mathbf{66.18}_{\pm 0.34}$	$\mathbf{70.31}_{\pm 0.95}$	$\mathbf{83.38}_{\pm 0.16}$	$\mathbf{85.03}_{\pm 0.11}$	$\mathbf{90.23}_{\pm 0.48}$	$\mathbf{87.32}_{\pm 0.32}$	$\mathbf{82.39}_{\pm 0.37}$	$\mathbf{67.23}_{\pm 0.14}$

and annotator-specific confusion matrices. In contrast, Chorus combines these posterior estimates with task-model evidence and annotator-specific feedback. Figure 4 shows improvements over every individual annotator in both annotation accuracy and downstream performance. These results demonstrate that Chorus constructs more effective supervision from heterogeneous LLM annotations, translating improved label estimates into downstream learning gains.

Performance with Larger Annotators. We examine whether the benefits of Chorus persist with larger annotation models. As shown in Table 2, larger annotators generally provide more accurate labels, improving downstream performance and narrowing the gap to clean supervision. However, combining annotations from several large models does not always improve on using a single model's labels. On Irony, training on Qwen2.5VL's labels yields 67.58% downstream accuracy, compared with 63.56% and 62.80% when using labels aggregated by DS and MACE, respectively. Chorus instead reaches 70.31%, improving downstream accuracy by 2.73% over the best individual annotator Qwen2.5VL-72B. Compared with Table 1, the larger pool further improves Chorus's downstream accuracy, with the largest gain of 3.90% on FashionMNIST (81.13% to 85.03%). Besides, Chorus with 3–8B annotators remains competitive with methods using 72–78B annotators, achieving comparable or higher mean downstream accuracy. On Irony, Chorus achieves 69.59% using small annotators, exceeding the strongest reported large-pool alternative, supervision from Qwen2.5VL-72B alone, by 2.01%. These results show that Chorus provides effective supervision from smaller annotators while enhancing the performance of larger annotation models.

Ablation Study. We evaluate task-model evidence and annotator-specific feedback on Irony and VQA-RAD using the 3–8B annotator pool, as shown in Table 3. All variants retain Bayesian aggregation, annotator-specific confusion matrices, and downstream model training. Disabling task-model evidence replaces model predictions with a uniform label prior during posterior inference. Without feedback, annotators use only the base task prompt throughout annotation; all other settings remain unchanged. Each component independently improves annotation and downstream accuracy over the variant using neither, while their combination performs best. Specifically, incorporating task-model evidence improves downstream accuracy by 0.71/2.99% on Irony/VQA-RAD, while feedback-guided annotation provides gains of 1.03/1.14%. These results confirm that both components improve supervision and provide complementary gains when combined.

Table 3: Ablation analysis of downstream task-model evidence and annotator-specific feedback.

Evidence	Feedback	Irony		VQA-RAD	
		Ann.	Down.	Ann.	Down.
×	×	$57.55_{\pm 0.39}$	$67.38_{\pm 0.73}$	$76.70_{\pm 0.19}$	$62.23_{\pm 0.67}$
✓	×	$58.59_{\pm 0.48}$	$68.56_{\pm 0.59}$	$77.45_{\pm 0.13}$	$64.90_{\pm 0.21}$
×	✓	$58.97_{\pm 0.46}$	$68.88_{\pm 0.79}$	$77.22_{\pm 0.78}$	$63.05_{\pm 0.16}$
✓	✓	$\mathbf{60.94}_{\pm 0.57}$	$\mathbf{69.59}_{\pm 1.48}$	$\mathbf{81.12}_{\pm 0.55}$	$\mathbf{66.04}_{\pm 0.14}$

Figure 5 further evaluates feedback effectiveness on VQA-RAD. The first subset provides a common starting point, as no feedback is available. On subsequent subsets, feedback generally improves annotation accuracy, although the gains vary across annotators and subsets. Qwen2.5VL-3B gains 4.89%–11.70% annotation accuracy, while Gemma3-4B gains up to 17.02% on Subset 4. These results indicate that feedback derived from earlier batches can improve annotations on new samples, contributing to better supervision before aggregation.

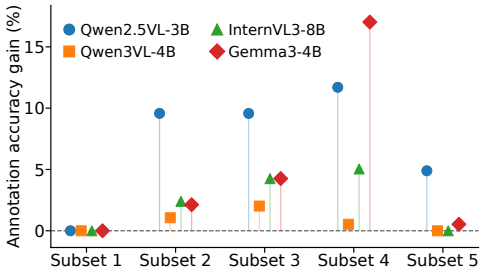


Figure 5: Annotation accuracy gains from feedback across VQA-RAD subsets.

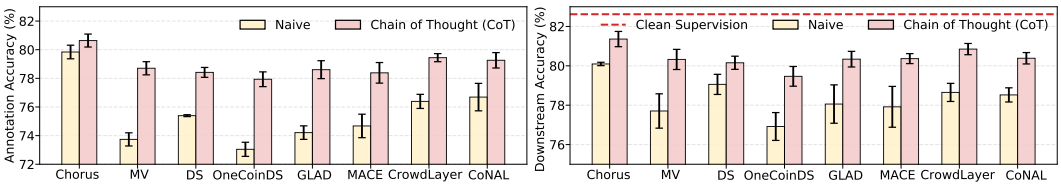


Figure 6: Impact of annotation strategies on the Emotion dataset. The dashed line in the right panel indicates downstream accuracy learned from ground-truth labels.

Impact of Annotation Strategies. Figure 6 compares naive prompting, which requests a label directly, with chain-of-thought (CoT) prompting, which requests a rationale before the label. CoT improves both annotation and downstream accuracy across all methods, showing that LLM-generated supervision is sensitive to annotation instructions. This observation supports the motivation for Chorus's feedback mechanism, which refines annotator-specific instructions based on recurring errors. Chorus achieves the highest accuracy under both prompts. With CoT, its downstream performance moves closer to clean supervision, suggesting that Chorus can benefit from improved individual annotations while continuing to exploit complementary evidence across annotators. More results and detailed prompt strategies are provided in Appendix E.1, F.

4 Related Work

LLM-based data annotation has been widely studied (Ding et al., 2023; Gilardi et al., 2023; He et al., 2024). However, annotation quality varies across tasks and models, making individual LLM predictions an uneven source of supervision. This variability has motivated research on combining annotation sources and refining LLM-generated labels (Li, 2024; Elumar et al., 2025; Xiao et al., 2023). These efforts address a central requirement of downstream learning: obtaining high-quality training labels from imperfect annotators to provide reliable supervision.

Probabilistic Aggregation and Learning from Crowds. Probabilistic aggregation estimates reliable supervision from noisy annotations by explicitly modeling annotator reliability. Dawid–Skene estimates annotator-specific confusion matrices (Dawid & Skene, 1979), GLAD models annotator expertise and item difficulty (Whitehill et al., 2009), and MACE estimates annotator competence and guessing behavior (Hovy et al., 2013). Bayesian classifier combination further represents uncertainty in latent labels and annotator reliability (Kim & Ghahramani, 2012), with variational formulations providing efficient posterior inference over labels and confusion matrices. Yao et al. (2024) model answer probabilities and semantic relationships to estimate answer truthfulness. Learning-from-crowds methods jointly estimate classifiers and annotator reliability (Raykar et al., 2010). Khetan et al. (2017) alternate between worker-quality estimation and noise-aware model updates. CrowdLayer learns annotator-specific transformations via backpropagation (Rodrigues & Pereira, 2018); related approaches regularize confusion-matrix estimation (Tanno et al., 2019) or distinguish shared from individual confusions (Chu et al., 2021).

LLM-as-a-Judge. LLM judges assess model outputs through scoring or pairwise comparison (Zheng et al., 2023), with specialized evaluators such as Prometheus 2 supporting user-defined criteria (Kim et al., 2024). These evaluations can also inform annotation refinement: DiZiNER employs a supervisor LLM to analyze inter-annotator disagreements (Kim & Yoon, 2026). Related approaches revise predictions through multi-agent debate (Du et al., 2023) or optimize instructions using critiques of labeled examples (Pryzant et al., 2023). Chorus connects reliability-aware aggregation with annotation refinement by using Bayesian label posteriors, informed by annotator reliability and downstream predictions, both as evidence for feedback on subsequent annotations.

5 Conclusion

We propose Chorus, a closed-loop framework for learning reliable supervision from noisy and heterogeneous LLM annotators without ground-truth labels. Chorus integrates downstream-aware Bayesian inference with feedback-guided annotation refinement, jointly estimating latent labels and annotator-specific reliability. Experiments across five datasets show improvements of up to 5.29% in annotation accuracy and 6.20% in downstream accuracy. By exploiting complementary expertise, Chorus enables relatively small, imperfect LLMs to collectively approach clean-label performance and remain competitive with substantially larger annotators, while further improving when equipped with 72–78B annotators. These results show that Chorus's gains stem from effectively exploiting complementary annotator expertise rather than merely relying on model scale.

AI USE STATEMENT

Large language models (LLMs) were used as assistive tools during the development of this work. Specifically, we used generative AI tools to assist with code implementation and debugging, checking mathematical derivations, discussing experimental design, preparing figures, and refining the language and presentation of the manuscript.

All research questions, methodological choices, experimental protocols, analyses, and conclusions were determined and validated by the authors. AI-assisted code, derivations, experimental suggestions, figures, and manuscript text were reviewed and, where necessary, revised by the authors before being incorporated into the work. The authors take full responsibility for the scientific content, experimental results, and claims presented in this paper.

The use of LLMs as annotators is part of the research methodology and is documented in the method and experimental sections, with prompt templates provided in Appendix F.

REPRODUCIBILITY STATEMENT

We describe the proposed framework and its optimization procedure in Section 2. The experimental section specifies the datasets, annotator models, downstream models, and training configurations. Appendix F.1 provides the annotation prompt templates, while Appendix F.2 documents the feedback template and its implementation settings. Together, these descriptions specify the annotation, supervision estimation, and downstream training procedures used in our experiments.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the association for computational linguistics: EMNLP 2020*, pp. 1644–1650, 2020.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Zhendong Chu, Jing Ma, and Hongning Wang. Learning from crowds by modeling common confusions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 5832–5840, 2021.

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

Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Boyang Li, Shafiq Joty, and Lidong Bing. Is gpt-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11173–11195, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Eray Can Elumar, Cem Tekin, and Osman Yagan. Cost-aware LLM-based online dataset annotation. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/054e9f9a286671ababa3213d6e59c1c2-Abstract-Conference.html.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Xingwei He, Zhenghao Lin, Yeyun Gong, A-Long Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, and Weizhu Chen. AnnoLLM: Making large language models to be better crowdsourced annotators. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pp. 165–190. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-industry.15. URL https://aclanthology.org/2024.naacl-industry.15/.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. Learning whom to trust with mace. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pp. 1120–1130, 2013.

Ashish Khetan, Zachary C Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. *arXiv preprint arXiv:1712.04577*, 2017.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In *Artificial Intelligence and Statistics*, pp. 619–627. PMLR, 2012.

Kon Woo Kim, Jin-Dong Kim, and Akiko Aizawa. Refining and reusing annotation guidelines for llm annotation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 37951–37964, 2026.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.248. URL https://aclanthology.org/2024.emnlp-main.248/.

Siun Kim and Hyung-Jin Yoon. Diziner: Disagreement-guided instruction refinement via simulating pilot annotation for zero-shot named entity recognition. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17498–17519, 2026.

Jason J Lau, Soumya Gayen, Asma Ben Abacha, and Dina Demner-Fushman. A dataset of clinically generated visual questions and answers about radiology images. *Scientific data*, 5(1):180251, 2018.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

Jiyi Li. A comparative study on annotation quality of crowdsourcing and LLM via label aggregation. In *ICASSP 2024 – 2024 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2024. doi: 10.1109/ICASSP48485.2024.10447803. URL https://arxiv.org/abs/2401.09760.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Lequan Lin, Dai Shi, Andi Han, Feng Chen, Qiuzheng Chen, Jiawen Li, Zhaoyang Li, Jiyuan Zhang, Zhenbang Sun, and Junbin Gao. Act as human: Multimodal large language model data annotation with critical thinking. *Advances in Neural Information Processing Systems*, 38:24107–24138, 2026.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA, 2015.

Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*, 2021.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 7957–7968, 2023.

Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of machine learning research*, 11(4), 2010.

Filipe Rodrigues and Francisco Pereira. Deep learning from crowds. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE transactions on neural networks and learning systems*, 34(11):8135–8153, 2022.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *International Conference on Learning Representations*, volume 2025, pp. 39894–39931, 2025.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. Large language models for data annotation and synthesis: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 930–957, 2024.

Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C Alexander, and Nathan Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11236–11245. IEEE, 2019.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Yu-Min Tseng, Wei-Lin Chen, Chung-Chi Chen, and Hsin-Hsi Chen. Evaluating large language models as expert annotators. *arXiv preprint arXiv:2508.07827*, 2025.

Dimitris G Tzikas, Aristidis C Likas, and Nikolaos P Galatsanos. The variational approximation for bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146, 2008.

Dmitry Ustalov, Nikita Pavlichenko, and Boris Tseitlin. Learning from crowds with Crowd-Kit. *Journal of Open Source Software*, 9(96):6227, 2024. doi: 10.21105/joss.06227.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. Want to reduce labeling cost? gpt-3 can help. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 4195–4205, 2021.

Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier Movellan, and Paul Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in neural information processing systems*, 22, 2009.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Ruixuan Xiao, Yiwen Dong, Junbo Zhao, Runze Wu, Minmin Lin, Gang Chen, and Haobo Wang. Freeal: Towards human-free active learning in the era of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14520–14535, 2023.

Peiran Yao, Jerin George Mathew, Shehraj Singh, Donatella Firmani, and Denilson Barbosa. A bayesian approach towards crowdsourcing the truths from LLMs. In *NeurIPS 2024 Workshop on Bayesian Decision-making and Uncertainty*, 2024. URL https://openreview.net/forum?id=oRW8i4EF0Z.

Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. *Journal of Machine Learning Research*, 17(102): 1–44, 2016.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2306.05685.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

A NOTATION

Table 4 summarizes the main notation used in this paper.

Table 4: Main symbols and notation in this paper.

Symbol	Explanation
T	Number of annotation rounds
M	Number of LLM annotators
K	Number of classes
$\mathbf{X}_t$	Input samples annotated at round t
$\mathbf{X}_{\leq t}$	Input samples accumulated up to round t
N_t	Number of samples in $\mathbf{X}_{\leq t}$
$\mathbf{A}_t$	Annotation matrix collected at round t
$\mathbf{A}_{\leq t}$	Annotation matrix accumulated up to round t
$A_{i,m}$	Annotation of sample i from annotator m
y_i	Latent label of sample i
$\boldsymbol{\phi}_i$	Variational posterior probabilities of y_i
$\boldsymbol{\pi}_i$	Task-model predictive prior for sample i
$\mathbf{C}^{(m)}$	Confusion matrix of annotator m
$\boldsymbol{\alpha}_k^{(m)}$	Dirichlet prior parameters for row k of $\mathbf{C}^{(m)}$
$\boldsymbol{\lambda}_k^{(m)}$	Variational Dirichlet parameters for row k
$f^{\mathcal{W}}$	Downstream task model with parameters $\mathcal{W}$
$\hat{y}_i^{(t)}$	Posterior reference label of sample i at round t
$c_i^{(t)}$	Posterior confidence of sample i at round t
$F_t^{(m)}$	Feedback generated for annotator m at round t

B PROOFS OF THEOREM 2.1 AND THEOREM 2.2

For clarity, we present the derivation for a generic annotation round t and omit the round index. We use $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ to denote the input samples in the accumulated dataset $\mathbf{X}_{\leq t}$, and write the accumulated annotation matrix $\mathbf{A}_{\leq t}$ simply as $\mathbf{A}$. All latent variables, variational distributions, annotator-reliability matrices, and task-model predictions are therefore understood to be defined over the accumulated data available at round t. This simplification does not affect the derivation and is used only to avoid cumbersome notation.

B.1 PRELIMINARIES

We formulate a Bayesian framework shown in Figure 3, wherein the parameters are estimated by estimated by the maximum likelihood estimate (MLE) approach. We first briefly introduce MLE under latent variable settings, and from the derive the parameter estimate rules for our framework. The general MLE problem is formulated as follows:

$$\theta_{MLE}^* = \arg\max_{\theta \in \Theta} p(\mathbf{O}; \theta), \tag{16}$$

where θ are the parameters of the probabilistic model and $\mathbf{O}$ are a set of observations.

When it is infeasible to directly model the likelihood function $p(\mathbf{O}; \theta)$ (as in our problem), we introduce variables $\mathbf{L}$ to connect the observations to unknown parameters. In this way, the likelihood function is transformed into

$$p(\mathbf{O}; \theta) = \int p(\mathbf{O}, \mathbf{L}; \theta) d\mathbf{L}. \tag{17}$$

The expectation maximization (EM) algorithm is employed for MLE with latent variables. The algorithm iteratively updates the elements in the likelihood function. Specifically, the E-step iteratively optimizes the latent variables $\mathbf{L}$, and the M-step iteratively optimize the parameters θ.

For the problem considered in this paper, the observed variables consist of the input samples $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and their noisy LLM annotations $\mathbf{A} = [A_{i,m}] \in \{1, \ldots, K\}^{N \times M}$. The latent variables include the unknown true labels $\mathbf{Y} = \{y_i\}_{i=1}^N$ and the annotator-specific reliability matrices $\mathbf{C} = \{\mathbf{C}^{(m)}\}_{m=1}^M$. The task model parameters are denoted by $\mathcal{W}$. The data sample and task model parameters are denoted as $\mathcal{P} = \{\mathbf{x}_i, \mathcal{W}\}$. A detailed expression of likelihood function is as follows:

$$\begin{aligned} p(\mathbf{A} \mid \mathbf{X}; \mathcal{W}) &= \prod_i \prod_m p(A_{i,m} \mid \mathcal{P}) \\ &= \prod_i \prod_m \int_{y_i, \mathbf{C}^{(m)}} p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P}) dy_i, \mathbf{C}^{(m)} \\ &= \int p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W}) d\mathbf{Y}, \mathbf{C}. \end{aligned} \tag{18}$$

We first decompose the likelihood function in Equation 5 into two terms as shown in Equation 6 of Section 2.2.2. We then employ variational EM to optimize variables in the Bayesian framework, wherein the E-step and M-step are designed to iteratively update the variables related to both the two modules. The reason for choosing the variational EM instead of the closed-form EM will also be discussed in the following section.

After applying the logarithm on Equation 18, we get

$$\begin{aligned} \log p(\mathbf{A} \mid \mathbf{X}; \mathcal{W}) &= \log(\prod_i \prod_m \int_{y_i, \mathbf{C}^{(m)}} p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P}) dy_i, \mathbf{C}^{(m)}) \\ &= \sum_i \sum_m \log(\int_{y_i, \mathbf{C}^{(m)}} p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P}) dy_i, \mathbf{C}^{(m)}) \\ &= \sum_i \sum_m \log(\int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})} dy_i, \mathbf{C}^{(m)}), \end{aligned} \tag{19}$$

where the approximation term $q(y_i, \mathbf{C}^{(m)})$ can be any probability density function.

According to Jensen's Inequality, we have described the entire process of

$$\begin{aligned} &\log(\int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})} dy_i, \mathbf{C}^{(m)}) \\ &= \log(\mathbb{E}_{q(y_i, \mathbf{C}^{(m)})}[\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}]) \\ &\geq \mathbb{E}_{q(y_i, \mathbf{C}^{(m)})}[\log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})})] \\ &= \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}) dy_i, \mathbf{C}^{(m)}. \end{aligned} \tag{20}$$

Therefore, we have

$$\log p(\mathbf{A} \mid \mathbf{X}; \mathcal{W}) \geq \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}) dy_i, \mathbf{C}^{(m)}. \tag{21}$$

The difference between the two sides of the inequality is

$$
\begin{aligned}
\Delta &= \log p(\mathbf{A} \mid \mathbf{X}; \mathcal{W}) - \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}) dy_i, \mathbf{C}^{(m)} \\
&= \log(\prod_i \prod_m p(A_{i,m} \mid \mathbf{x}_i; \mathcal{P})) - \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}) dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m \log(p(A_{i,m} \mid \mathcal{P})) - \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})}) dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} [q(y_i, \mathbf{C}^{(m)}) \log(p(A_{i,m} \mid \mathcal{P})) - q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}{q(y_i, \mathbf{C}^{(m)})})] dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m} \mid \mathcal{P}) q(y_i, \mathbf{C}^{(m)})}{p(A_{i,m}, y_i, \mathbf{C}^{(m)} \mid \mathcal{P})}) dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{p(A_{i,m} \mid \mathcal{P}) q(y_i, \mathbf{C}^{(m)})}{p(y_i, \mathbf{C}^{(m)} \mid A_{i,m}, \mathcal{P}) p(A_{i,m} \mid \mathcal{P}))}) dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m \int_{y_i, \mathbf{C}^{(m)}} q(y_i, \mathbf{C}^{(m)}) \log(\frac{q(y_i, \mathbf{C}^{(m)})}{p(y_i, \mathbf{C}^{(m)} \mid A_{i,m}, \mathcal{P}))}) dy_i, \mathbf{C}^{(m)} \\
&= \sum_i \sum_m KL(q || p(y_i, \mathbf{C}^{(m)} \mid A_{i,m}, \mathcal{P})).
\end{aligned}
$$

The gap Δ refers to the second term in Equation 6, which we need to minimize in the E-step. However, the closed-form EM updates do not work in the case of discrete-continuous variables. We develop a mean field variational inference, following the idea of approximating the posterior distribution of latent variables $p(y_i, \mathbf{C}^{(m)} \mid A_{i,m}, \mathcal{P})$ with the variational distribution $q(\mathbf{Y}, \mathbf{C}) = \prod_i \prod_m q(y_i, \mathbf{C}^{(m)})$.

In the mean-field approach, we assume that

$$
q(\mathbf{Y}, \mathbf{C}) = q(\mathbf{Y}) q(\mathbf{C}) = \prod_i q(y_i) \prod_m \prod_k q(\mathbf{C}^{(m)}_{k,:}), \tag{22}
$$

where $q(y_i) = \mathrm{Cat}(\boldsymbol{\pi}_i)$ (i.e., Equation 1) and $q(\mathbf{C}^{(m)}_{k,:}) = \mathrm{Dir}(\boldsymbol{\alpha}^{(m)}_k)$ (i.e., Equation 2).

We then derive the coordinate-ascent updates for the two variational factors $q(\mathbf{Y})$ and $q(\mathbf{C})$, corresponding to Theorem 2.1 and Theorem 2.2, respectively.

B.2 Proof of Theorem 2.1

We first update $q(\mathbf{Y})$ while keeping $q(\mathbf{C})$ fixed. Following the standard mean-field variational update, the optimal factor is obtained by retaining the ELBO terms that depend on $\mathbf{Y}$ and absorbing the remaining terms into the normalization constant

$$
q(\mathbf{Y}) \propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W}))]\}. \tag{23}
$$

Under the conditional independence assumptions of our generative model, the complete-data joint distribution can be factorized as

$$
p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W}) = \prod_i p(y_i \mid \mathcal{P}) \times \prod_m p(\mathbf{C}^{(m)}) \times \prod_i \prod_m p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}). \tag{24}
$$

This is because y_i only depends on $\mathcal{P}$. $A_{i,m}$ does not depend on $\mathcal{P}$ given y_i and $\mathbf{C}^{(m)}$.

For updating $q(y_i)$, we only retain the terms in the log-joint that depend on y_i. From Equation 24, we have

$$
\log(p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W})) = \log(p(y_i \mid \mathcal{P})) + \sum_m \log(p(\mathbf{C}^{(m)})) + \sum_m \log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})). \tag{25}
$$

Substituting Equation 25 into Equation 23, we have

$$\begin{aligned} q(y_i) &\propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P})) + \sum_m \log(p(\mathbf{C}^{(m)})) + \sum_m \log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\} \\ &\propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P}))] + \mathbb{E}_{q(\mathbf{C})}[\sum_m \log(p(\mathbf{C}^{(m)}))] + \mathbb{E}_{q(\mathbf{C})}[\sum_m \log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\}. \end{aligned} \tag{26}$$

We remove the irrelevant term related to $q(y_i)$, i.e., $\mathbb{E}_{q(\mathbf{C})}[\sum_m \log(p(\mathbf{C}^{(m)}))]$, then we get

$$\begin{aligned} q(y_i) &\propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P}))] + \mathbb{E}_{q(\mathbf{C})}[\sum_m \log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\} \\ &\propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P}))]\} \times \exp\{\mathbb{E}_{q(\mathbf{C})}[\sum_m \log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\} \\ &\propto \exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P}))]\} \times \exp\{\sum_m \mathbb{E}_{q(\mathbf{C}^{(m)})}[\log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\}. \end{aligned} \tag{27}$$

Since $\log(p(y_i \mid \mathcal{P}))$ does not contain the variable $q(\mathbf{C})$, we have

$$\exp\{\mathbb{E}_{q(\mathbf{C})}[\log(p(y_i \mid \mathcal{P}))]\} = \exp\{\log(p(y_i \mid \mathcal{P}))\} = p(y_i \mid \mathcal{P}). \tag{28}$$

Substituting Equation 28 into Equation 27, we obtain

$$\begin{aligned} q(y_i) &\propto p(y_i \mid \mathcal{P}) \times \exp\{\sum_m (\mathbb{E}_{q(\mathbf{C}^{(m)})}[\log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))])\} \\ &\propto p(y_i \mid \mathcal{P}) \prod_m \exp\{\mathbb{E}_{q(\mathbf{C}^{(m)})}[\log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\}. \end{aligned} \tag{29}$$

Equation 29 is equivalent to Equation 9 of Section 2.2.2 when we define $\mathbb{E}_{q(\mathbf{C}^{(m)})}[log(\cdot)] = g_{p(\mathbf{C}^{(m)})}(\cdot)$.

Based on Equation 29, we now derive the categorical parameter update for $q(y_i)$ given the current variational parameters $\boldsymbol{\lambda}^{(m)}$ of the annotator reliability matrices and the task-model prior $\boldsymbol{\pi}_i$.

The task model provides a label prior for each input sample. Specifically, given $\mathbf{x}_i$, the predictive distribution of the task model is denoted as

$$\boldsymbol{\pi}_{i,k} = f^{\mathcal{W}}(y_i = k \mid \mathbf{x}_i), \quad k = 1, \ldots, K. \tag{30}$$

This prior is then combined with the annotation likelihoods to update the variational posterior $q(y_i) = \mathrm{Cat}(\boldsymbol{\phi}_i)$.

According to Equation 3, the likelihood of an LLM m annotating data sample $\mathbf{x}_i$ as the probability conditioned on the LLM reliability matrix $\mathbf{C}^{(\mathbf{m})}$ and the true label y_i:

$$p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}) = \prod_{l=1}^{K} (\mathbf{C}^{(m)}_{y_i,l})^{\mathbb{I}[A_{i,m}=l]}. \tag{31}$$

We substitute the probabilities of $p(y_i \mid \mathcal{P})$ and $p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})$ in Equation 29 by their respective definitions in Equation 30 and Equation 31 and get

$$\begin{aligned} q(y_i) &\propto \boldsymbol{\pi}_i \prod_m \exp\{\mathbb{E}_{q(\mathbf{C}^{(m)})}[\log(\prod_{l=1}^{K} (\mathbf{C}^{(m)}_{y_i,l})^{\mathbb{I}[A_{i,m}=l]})]\} \\ &\propto \boldsymbol{\pi}_i \prod_m \exp\{\mathbb{E}_{q(\mathbf{C}^{(m)})}[\sum_{l=1}^{K} \mathbb{I}[A_{i,m} = l] \log(\mathbf{C}^{(m)}_{y_i,l})]\} \\ &\propto \boldsymbol{\pi}_i \prod_m \exp\{\mathbb{E}_{q(\mathbf{C}^{(m)})}[\log(\mathbf{C}^{(m)}_{y_i,A_{i,m}})]\}. \end{aligned} \tag{32}$$

In the case when $y_i = k$, Equation 32 is equivalent to

$$q(y_i = k) \propto \boldsymbol{\pi}_{i,k} \prod_m \exp\{\mathbb{E}_{q(\mathbf{C}^{(m)}_{k,:})}[\log(\mathbf{C}^{(m)}_{k,A_{i,m}})]\}. \tag{33}$$

According to the Equation 8, $q(\mathbf{C}_{k,:}^{(m)}) = \mathrm{Dir}(\boldsymbol{\lambda}_k^{(m)})$, where $\boldsymbol{\lambda}_k^{(m)} = (\boldsymbol{\lambda}_{k,1}^{(m)}, \boldsymbol{\lambda}_{k,2}^{(m)}, \ldots, \boldsymbol{\lambda}_{k,K}^{(m)})$. Using the logarithmic expectation property of the Dirichlet distribution:

$$\mathbb{E}_{q(\mathbf{C}_{k,:}^{(m)})}[\log(\mathbf{C}_{k,A_{i,m}}^{(m)})] = \Psi(\boldsymbol{\lambda}_{k,A_{i,m}}^{(m)}) - \Psi(\sum_{l=1}^{K} \boldsymbol{\lambda}_{k,l}^{(m)}). \tag{34}$$

Substituting Equation 34 into Equation 33, we can obtain the update rule of $q(y_i = k)$ as Equation 11 of Section 2.2.2:

$$q(y_i = k) \propto \boldsymbol{\pi}_{i,k} \prod_m \exp\{\Psi(\boldsymbol{\lambda}_{k,A_{i,m}}^{(m)}) - \Psi(\sum_{l=1}^{K} \boldsymbol{\lambda}_{k,l}^{(m)})\}, \tag{35}$$

where $\Psi(\cdot)$ is the Digamma function. We now conclude the proof of Theorem 2.1.

B.3 Proof of Theorem 2.2

We first update $q(\mathbf{C})$ while keeping $q(\mathbf{Y})$ fixed. Following the standard mean-field variational update, the optimal factor is obtained by retaining the ELBO terms that depend on $\mathbf{C}$ and absorbing the remaining terms into the normalization constant

$$q(\mathbf{C}) \propto \exp\{\mathbb{E}_{q(\mathbf{Y})}[\log(p(\mathbf{A}, \mathbf{Y}, \mathbf{C} \mid \mathbf{X}; \mathcal{W}))]\}. \tag{36}$$

For the m-th LLM annotator, only the prior $p(\mathbf{C}^{(m)})$ and the annotation likelihoods $p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})$ depend on $\mathbf{C}^{(m)}$. Therefore, Equation 36 gives

$$\begin{aligned} q(\mathbf{C}^{(m)}) &\propto \exp\{\mathbb{E}_{q(\mathbf{Y})}[\log p(\mathbf{C}^{(m)}) + \sum_i \log p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})]\} \\ &\propto p(\mathbf{C}^{(m)}) \prod_i \exp\{\mathbb{E}_{q(y_i)}[\log p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})]. \end{aligned} \tag{37}$$

Since $q(y_i) = \mathrm{Cat}(\boldsymbol{\phi}_i)$, we have

$$\begin{aligned} \mathbb{E}_{q(y_i)}[\log p(A_{i,m} \mid y_i, \mathbf{C}^{(m)})] &= \sum_k q(y_i = k) \log p(A_{i,m} \mid y_i = k, \mathbf{C}^{(m)}) \\ &= \sum_k \boldsymbol{\phi}_{i,k} \log p(A_{i,m} \mid y_i = k, \mathbf{C}^{(m)}) \\ &\overset{def}{=} \mathbb{E}_{\boldsymbol{\phi}_i}[\log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]. \end{aligned} \tag{38}$$

Therefore, Equation 37 is finally transformed into

$$q(\mathbf{C}^{(m)}) \propto p(\mathbf{C}^{(m)}) \prod_i \exp\{\mathbb{E}_{\boldsymbol{\phi}_i}[\log p(\boldsymbol{A}_{i,m} \mid y_i, \mathbf{C}^{(m)})]\}. \tag{39}$$

Equation 39 is equivalent to Equation 10 of Section 2.2.2 when we define $\mathbb{E}_{\boldsymbol{\phi}_i}[\log(\cdot)] = g_{\boldsymbol{\phi}_i}(\cdot)$.

Based on Equation 39, we now derive the update rule of $q(\mathbf{C}^{(m)})$ given the variational posterior $q(y_i) = \mathrm{Cat}(\boldsymbol{\phi}_i)$ and the Dirichlet prior parameters $\boldsymbol{\alpha}^{(m)}$.

According to Equation 31, the $\mathbb{E}_{\boldsymbol{\phi}_i}[\log p(\boldsymbol{A}_{i,m} \mid y_i, \mathbf{C}^{(m)})]$ in Equation 40 can be reformulated to

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\phi}_i}[\log(p(A_{i,m} \mid y_i, \mathbf{C}^{(m)}))]\} &= \mathbb{E}_{\boldsymbol{\phi}_i}[\log(\prod_{l=1}^{K} (\mathbf{C}_{y_i,l}^{(m)})^{\mathbb{I}[A_{i,m}=l]})] \\ &= \mathbb{E}_{\boldsymbol{\phi}_i}[\sum_{l=1}^{K} \mathbb{I}[A_{i,m} = l] \log(\mathbf{C}_{k,l}^{(m)})] \\ &= \sum_{k=1}^{K} \boldsymbol{\phi}_{i,k} \sum_{l=1}^{K} \mathbb{I}[A_{i,m} = l] \log(\mathbf{C}_{k,l}^{(m)}) \\ &= \sum_{k=1}^{K} \sum_{l=1}^{K} \boldsymbol{\phi}_{i,k} \mathbb{I}[A_{i,m} = l] \log(\mathbf{C}_{k,l}^{(m)}). \end{aligned} \tag{40}$$

Substituting Equation 40 into Equation 39, we have

$$\begin{aligned} q(\mathbf{C}^{(m)}) &\propto p(\mathbf{C}^{(m)}) \prod_{i=1}^{N} \exp\{\sum_{k=1}^{K}\sum_{l=1}^{K} \boldsymbol{\phi}_{i,k}\mathbb{I}[A_{i,m}=l]\log(\mathbf{C}_{k,l}^{(m)})\} \\ &\propto p(\mathbf{C}^{(m)}) \prod_{i=1}^{N}\prod_{k=1}^{K}\prod_{l=1}^{K}(\mathbf{C}_{k,l}^{(m)})^{\boldsymbol{\phi}_{i,k}\mathbb{I}[A_{i,m}=l]} \\ &\propto p(\mathbf{C}^{(m)}) \prod_{k=1}^{K}\prod_{l=1}^{K}(\mathbf{C}_{k,l}^{(m)})^{\sum_{i=1}^{N}\boldsymbol{\phi}_{i,k}\mathbb{I}[A_{i,m}=l]}. \end{aligned} \tag{41}$$

According to Equation 2, each row of $\mathbf{C}^{(m)}$ defines a categorical distribution over the annotations produced by the m-th LLM given a latent true class. Using the kernel form of the Dirichlet density, the prior over the full confusion matrix can be written as

$$q(\mathbf{C}^{(m)}) \propto \prod_{k=1}^{K}\prod_{l=1}^{K}(\mathbf{C}_{k,l}^{(m)})^{\boldsymbol{\alpha}_{k,l}^{(m)}-1}. \tag{42}$$

Substituting Equation 42 into Equation 41, we have

$$q(\mathbf{C}^{(m)}) \propto \prod_{k=1}^{K}\prod_{l=1}^{K}(\mathbf{C}_{k,l}^{(m)})^{\boldsymbol{\alpha}_{k,l}^{(m)}-1+\sum_{i=1}^{N}\boldsymbol{\phi}_{i,k}\mathbb{I}[A_{i,m}=l]}. \tag{43}$$

Therefore, each row of the variational posterior remains Dirichlet-distributed:

$$q(\mathbf{C}^{(m)}) = \prod_{k=1}^{K}\mathrm{Dir}(\mathbf{C}_{k,:}^{(m)} \mid \boldsymbol{\lambda}_{k,:}^{(m)}), \tag{44}$$

where

$$\boldsymbol{\lambda}_{k,l}^{(m)} = \boldsymbol{\alpha}_{k,l}^{(m)} + \sum_{i=1}^{N}\boldsymbol{\phi}_{i,k}\mathbb{I}[A_{i,m}=l]. \tag{45}$$

This completes the proof of Theorem 2.2.

C Experimental Details

C.1 Datasets

Table 5 summarizes the five datasets used in our experiments. Emotion and Irony are drawn from TweetEval and cover four-class emotion classification (anger, joy, optimism, and sadness) and binary irony detection, respectively. FashionMNIST contains images from 10 clothing categories, while Pets uses Oxford-IIIT Pet for 37-class pet breed classification. For VQA-RAD, we retain only radiology image–question pairs with binary yes/no answers. We use the provided training splits (the `trainval` split for Pets) as annotation pools and the corresponding test splits for downstream evaluation. Ground-truth labels in the annotation pools are withheld from Chorus.

Table 5: Datasets used in our experiments.

Dataset	Input	Task	#Class	Training Size	Test Size
Emotion	Text	Text emotion classification	4	3257	1421
Irony	Text	Text irony detection	2	2,862	784
FashionMNIST	Image	Image classification of cloth item	10	60,000	10,000
Pets	Image	Image classification for Pet breed	37	3,680	3,669
VQA-RAD	Image + Text	Question-answer pairs on radiology images	2	940	262

C.2 Downstream Model Configurations

Inspired by the downstream evaluation protocol in Lin et al. (2026), we adopt task-specific fine-tuning configurations for ResNet-18, RoBERTa-base, and BLIP-VQA-base, with adaptations to our datasets and experimental setting. For each dataset, all compared methods use the same downstream model architecture, initialization, and optimization configuration. We further match the total number of downstream optimization steps between Chorus and the baselines.

ResNet-18. For the image classification tasks, FashionMNIST and Pets, we fine-tune a ResNet-18 model (He et al., 2016) initialized with ImageNet-1K pretrained weights (Russakovsky et al., 2015). For each dataset, the model is trained for 6 epochs using a batch size of 4096 for FashionMNIST and 32 for Pets. We use AdamW and select the learning rate from $[1e-2, 1e-3, 5e-4, 1e-4]$. The final classification layer is adjusted to match the number of classes in each dataset.

RoBERTa. For the text classification tasks, including Emotion and Irony, we fine-tune RoBERTa-base (Liu et al., 2019). The model is trained for 6 epochs with a batch size of 32, using the AdamW optimizer. The key hyperparameters include the learning rate, with a search space of $[1e-4, 5e-5, 2e-5, 1e-5]$.

BLIP-VQA. For the VQA task, we fine-tune BLIP-VQA-base (Li et al., 2022) on the closed-ended VQA-RAD subset for 6 epochs with a batch size of 32. We use AdamW and select the learning rate from $[2e-4, 1e-4, 5e-5, 2e-5]$ on the closed-ended subset of VQA-RAD. Following the closed-ended setting, the model is optimized to generate an answer from the predefined answer space [e.g., "Yes" or "No"].

C.3 Baseline Configurations

We use Crowd-Kit (Ustalov et al., 2024) for the five statistical aggregation methods and the annotator-noise layers of CrowdLayer and CoNAL, integrating them into our downstream training pipeline. These seven methods use the annotator labels. The method-specific hyperparameters follow the defaults documented in Crowd-Kit. Below, we specify their configurations and those of the two agent-based baselines.

Majority Voting (MV). MV assigns each sample the most frequently predicted label, giving equal weight to all annotators. It requires no learned annotator parameters.

Dawid–Skene (DS). (Dawid & Skene, 1979) DS jointly estimates latent labels and annotator-specific confusion matrices through expectation-maximization. We use a maximum of 100 iterations and a convergence tolerance of 10^{-5}.

OneCoinDS. (Zhang et al., 2016) OneCoinDS simplifies DS by representing each annotator's reliability with a single accuracy parameter and distributing error probability uniformly across incorrect classes. We use the same iteration limit and convergence tolerance as DS.

GLAD. (Whitehill et al., 2009) GLAD models annotation correctness through annotator ability and sample difficulty. We use a maximum of 100 outer iterations with tolerance 10^{-5}, retaining Crowd-Kit's default M-step settings of 25 optimization iterations and tolerance 10^{-2}.

MACE. (Hovy et al., 2013) MACE estimates annotator competence by distinguishing correct responses from responses drawn from an annotator-specific label distribution. We use variational Bayesian inference with 10 random restarts and 50 iterations per restart. The prior parameters are $\alpha = \beta = 0.5$, the initialization noise is 0.5, and the random seed is 42.

CrowdLayer. (Rodrigues & Pereira, 2018) CrowdLayer learns an annotator-specific transformation of the task model's outputs. We use the matrix-multiplication variant (`conn_type=mw`), with one identity-initialized $K \times K$ matrix per annotator. The task model and annotation layers are jointly trained using cross-entropy against the observed annotations.

CoNAL. (Chu et al., 2021) CoNAL combines shared and annotator-specific confusion matrices, with their mixture determined by sample features and annotator identity. We use a 20-dimensional common-noise embedding (`com_emb_size=20`) and one-hot annotator features. Both confusion components use Crowd-Kit's identity-based initialization.

MAD. (Du et al., 2023) Following the original MAD setup, we run one independent annotation round followed by two synchronous debate rounds. In each debate round, annotators revise their predictions using the original input and all responses from the preceding round. Final labels are determined by majority vote, with ties broken in favor of the smallest label index.

DiZiNER. (Kim & Yoon, 2026) We follow the DiZiNER refinement procedure, adapting its NER span-level agreement to sample-level label agreement for classification tasks. We use a fixed random pilot set of 25 examples and run five refinement cycles. Qwen2.5-VL-72B-Instruct (Bai et al., 2025b) acts as the supervisor and examines up to five examples with high disagreement per cycle. Refinement proceeds through four stages: pattern analysis, annotator-specific diagnosis, guideline integration, and guideline organization. The resulting guidelines contain at most five shared rules and three rules per annotator. We select the three annotator–iteration pairs with the highest mean pairwise agreement on the pilot set. Each pair independently labels the full annotation pool and trains a downstream model. Within each run, we average evaluation metrics across three candidates.

For MAD and DiZiNER, LLM generation uses deterministic decoding (`do_sample=false`), with annotator outputs limited to 384 new tokens. DiZiNER's supervisor outputs are limited to 2,048 new tokens. Malformed responses receive up to two format-repair retries. Across all baselines, downstream models use AdamW (Loshchilov & Hutter, 2019) with weight decay 0.01 and gradient clipping at a norm of 1.0. Downstream accuracy is measured on the test set. For CrowdLayer and CoNAL, both metrics use task-model predictions without the annotator-noise layers. For the remaining baselines, annotation accuracy is computed from their output labels. We report the mean and standard deviation over three runs.

D Variational EM for Reliable Supervision

Algorithm 2 details the variational EM procedure (VEM$(\cdot)$) used within each annotation round. Given the accumulated data $\mathbf{X}_{\leq t}$ and annotations $\mathbf{A}_{\leq t}$, Chorus alternates between variational inference over the latent variables and task-model optimization until convergence of the variational objective. In Lines 2-3, the label posterior $q(y_i)$ is updated for each data sample according to Theorem 2.1, using the current task-model prior and annotator reliability estimates. In Lines 4-5, the reliability posterior $q(\mathbf{C}^{(m)}_{k,:})$ is updated for each LLM annotator and each latent class according to Theorem 2.2. The updated posterior label distributions $\{\boldsymbol{\phi}_i\}_{i=1}^{N}$ then serve as soft supervision for updating the task-model parameters $\mathcal{W}$ via standard gradient descent. The inferred label distributions then provide soft supervision for updating $\mathcal{W}$, after which the task-model priors $\boldsymbol{\pi}_i = f^{\mathcal{W}}(\mathbf{x}_i)$ are refreshed for the next iteration.

Algorithm 2: Variational EM for Reliable Supervision

Input : Accumulated data $\mathbf{X}_{\leq t} = \{\mathbf{x}_i\}_{i=1}^{N_t}$; accumulated annotation matrix $\mathbf{A}_{\leq t}$; Dirichlet prior $\boldsymbol{\alpha}$; task model $f^{\mathcal{W}}$
Output : Variational posteriors $q(y_i)$ and $q(\mathbf{C}^{(m)}_{k,:})$; updated task model $f^{\mathcal{W}}$
Initialize: Task-model prior $\boldsymbol{\pi}_i = f^{\mathcal{W}}(\mathbf{x}_i)$; variational parameters $\boldsymbol{\phi}_i$ and $\boldsymbol{\lambda}^{(m)}_{k,:}$

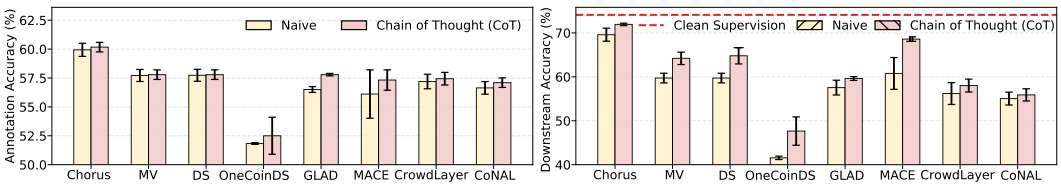
1 **while** *Equation 6 has not converged* **do**
  /* E-step: Variational posterior inference */
2 **for** $i = 1, \ldots, N_t$ **do**
3 Update $q(y_i) = \mathrm{Cat}(\boldsymbol{\phi}_i)$ using Equation 11;
4 **for** $m = 1, \ldots, M$ **do**
5 Update $q(\mathbf{C}^{(m)}_{k,:}) = \mathrm{Dir}(\boldsymbol{\lambda}^{(m)}_{k,:})$ using Equation 12
  /* M-step: Task-model update */
6 Update $\mathcal{W}$ using $\{\boldsymbol{\phi}_i\}_{i=1}^{N_t}$ as soft supervision via gradient descent
7 Refresh task-model prior $\boldsymbol{\pi}_i = f^{\mathcal{W}}(\mathbf{x}_i), \forall\, \mathbf{x}_i \in \mathbf{X}_{\leq t}$

Let I_t denote the number of variational EM iterations at annotation round t. Since the annotation matrix $\mathbf{A}_{\leq t}$ is dense, updating $q(y_i)$ for all data samples (lines 2-3) requires a time complexity of $O(K|\mathbf{A}_{\leq t}|)$, where $|\mathbf{A}_{\leq t}| = N_t M$ is the number of annotation entries. Similarly, updating $q(\mathbf{C}^{(m)}_{k,:})$ for all LLM annotators and classes (lines 4-5) has the same complexity. Therefore, the complexity of round t is $O(I_t \times K|\mathbf{A}_{\leq t}| + \mathcal{T}^{(t)}_{\mathcal{W}})$, where $\mathcal{T}^{(t)}_{\mathcal{W}}$ denotes the task-model training cost per iteration at round t. Across all T annotation rounds, the total complexity is $O(\sum_{t=1}^{T} I_t \times K|\mathbf{A}_{\leq t}| + \mathcal{T}^{(t)}_{\mathcal{W}})$.

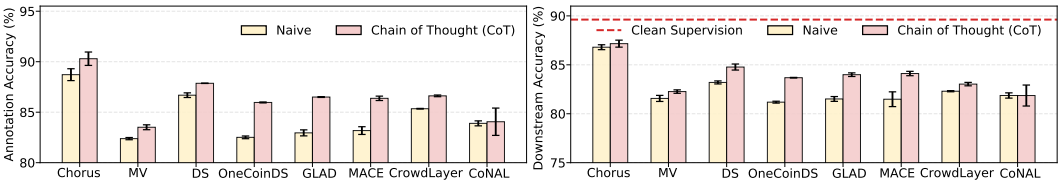
E More Results and Analysis

E.1 Additional Results with Different Annotation Prompts

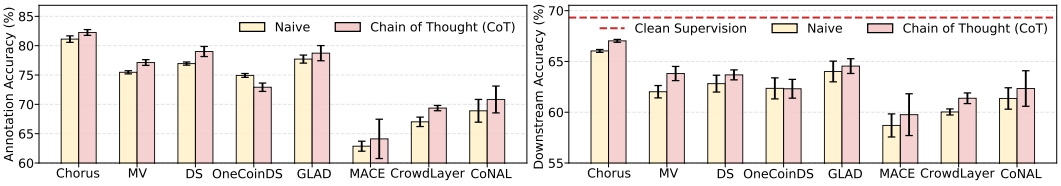
Figure 7 extends the comparison between naive and chain-of-thought (CoT) prompting to Irony, Pets, and VQA-RAD datasets, covering text classification, image classification, and visual question answering tasks. CoT generally improves annotation quality, but its effect varies across methods and datasets; for example, some baselines exhibit only marginal gains or slight degradation. In contrast, Chorus consistently benefits from CoT and achieves the highest annotation and downstream accuracies under both prompting strategies across all three datasets. On the Irony and VQA-RAD datasets, CoT further raises its downstream accuracy to the clean-supervision performance, while the gains on the Pets dataset narrow the remaining gap. These results show that Chorus remains robust to annotation strategies while effectively translating improvements in individual annotations into stronger downstream supervision.



(a) Irony dataset



(b) Pets dataset



(c) VQA-RAD dataset

Figure 7: Impact of annotation strategies on Irony, Pets, and VQA-RAD. The left and right panels report annotation and downstream accuracy, respectively. The dashed line indicates downstream accuracy under clean supervision.

E.2 Sensitivity Analysis

Impact of the number of subsets T. We vary the number of annotation subsets $T \in \{3, 5, 7, 9\}$ on the VQA-RAD dataset using the 3–8B annotator pool, keeping the total downstream optimization steps budget fixed. As shown in Figure 8, annotation accuracy ranges from 80.68% to 81.15%, while downstream accuracy ranges from 65.61% to 66.04%. These small variations indicate that Chorus is robust to T rather than relying on a carefully chosen setting. Increasing T does not consistently improve either metric: more subsets enable more frequent feedback, but provide fewer examples for identifying recurring annotation errors in each annotation round. This trade-off may help explain why more frequent feedback does not necessarily yield better supervision.

Impact of the feedback confidence threshold τ. We vary the feedback confidence threshold $\tau \in \{0.6, 0.7, 0.8, 0.9, 0.95\}$ on the VQA-RAD dataset, keeping the remaining settings fixed. As shown in Figure 9, the default $\tau = 0.8$ yields the highest mean annotation and downstream accuracies, at 81.12% and 66.04%, respectively. Lower thresholds may include less reliable reference labels in feedback, while higher thresholds reduce the amount of feedback available. This trade-off may

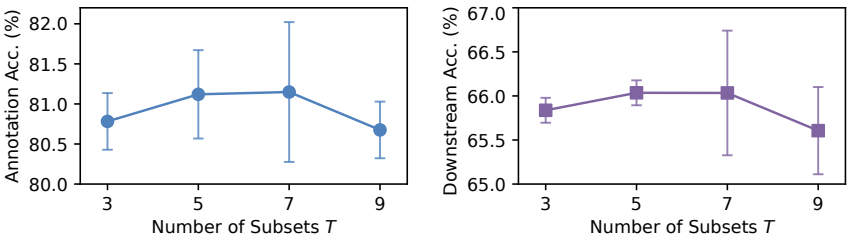


Figure 8: Sensitivity to the number of annotation subsets T on the VQA-RAD dataset. Error bars show standard deviations over three runs.

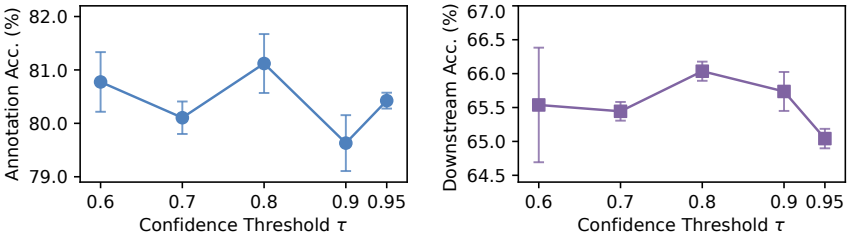


Figure 9: Sensitivity to the feedback confidence threshold τ on the VQA-RAD dataset. Error bars show standard deviations over three runs.

explain why performance peaks at $\tau = 0.8$. Mean downstream accuracy varies by less than one percentage point across the tested thresholds, indicating modest sensitivity to τ.

F Examples of Annotation Prompt and Feedback

This section presents annotation prompts and feedback templates used in Chorus. We first present task-specific annotation prompts and their expected output formats and then describe how annotator-specific feedback is constructed and incorporated into subsequent annotation rounds. The prompt wording follows the implementation, with whitespace and line breaks adjusted for readability.

F.1 Annotation Prompt Templates

Our base annotation prompts are adapted from the task-specific templates presented by Lin et al. (2026), with modifications to the task instructions and output constraints. We provide direct annotation (naive) prompts for image classification, text classification, and binary visual question answering (VQA), along with a representative chain-of-thought (CoT) prompt for image classification. The CoT format can be easily extended to other tasks by adapting their corresponding direct prompts in the same manner. In the following examples, `{purple}` denotes input fields, while `[blue]` denotes expected outputs. Curly braces indicate placeholders populated with task-specific content, whereas square brackets specify the required output format.

Input Fields and Output Formats

- `{image_data}`: input image, supplied through the model's image interface.
- `{text_data}` and `{question}`: input text and the question associated with an image, respectively.
- `{label_list_with_index}`: the mapping from class indices to label names.
- `{first_label}`: the label assigned index 0.
- `{allowed_outputs}`: the valid bracketed indices, such as `[0], [1], ..., [K-1]`.
- `[label_index]`: the predicted class index. For binary VQA, `[0]` denotes No and `[1]` denotes Yes.
- `[reasoning][label_index]`: the CoT output, containing a rationale followed by the predicted class index.

The implementation uses zero-based class indices, corresponding to the same classes indexed from 1 to K in the mathematical formulation. The outputs shown below specify the requested format.

Image Classification: Naive Annotation

Input image: {image_data}

Prompt
Determine the label of the image classification task. The list of labels is: {label_list_with_index}. The required output format is: [label_index]. For example, if the label is {first_label}, you should output [0]. Do not return other texts. The output must be exactly one of: {allowed_outputs}.

Expected output
[label_index]

Text Classification: Naive Annotation

Input text: {text_data}

Prompt
You are a strict text classifier. Your task is to classify the given text into exactly one of the provided labels. Label mapping: {label_list_with_index}. The input text is: {text_data}. The required output format is: [label_index]. Do not explain. Do not return any explanation, words, punctuation, or extra text. The output must be exactly one of: {allowed_outputs}.

Expected output
[label_index]

Binary VQA: Naive Annotation

Input image: {image_data} and {text_data}

Prompt
Answer the question based on the given image {image_data}. The question is: {text_data}. The required output format is [0] for No and [1] for Yes. Do not return other texts.

Expected output
[0] or [1]

Image Classification: CoT Annotation

Input image: {image_data}

Prompt
Determine the label of image classification task. Choose exactly one label from the candidate labels below.
Candidate labels:
{label_list_with_index}
Output format rules:
– Provide step-by-step reasoning in the first bracket pair.
– Keep the reasoning concise and self-contained.
– Provide the final label index in the second bracket pair.
– You must always choose one label from the given label mapping.
– Do not create new labels such as unknown, unclear, none, invalid, or cannot classify.
– The final label must be exactly one of: {allowed_outputs}.
– Do not output any text outside the two bracket pairs.
– Required format: [reasoning][label_index]

Expected output
[reasoning][label_index]

F.2 Feedback Prompt Template

The following template illustrates how the annotator-specific feedback described in Section 2.3 is appended to the base annotation prompt. Task inputs and expected outputs retain the color convention of Appendix F.1 ({purple} denotes input fields, and [blue] denotes expected outputs), while feedback fields are shown as uncolored placeholders.

Feedback-guided Annotation: Shared Template

Prompt
{base_task_prompt}

Guidance learned from previous annotation rounds.
Review the following error patterns identified from the previous annotation round:

1. In {count} previous high-confidence cases, you predicted "{predicted_label}", while the aggregated reference denotes "{reference_label}". Pay closer attention to the distinction between these two labels.

Representative example 1: {text_data}
Representative example 2: {text_data}

2. ...

Treat these as general guidance rather than replacement labels. Classify each new sample independently from its actual content.

Expected output
[label_index] for direct annotation, or [reasoning][label_index] for CoT annotation.

Each numbered instruction summarizes a recurring disagreement between an annotator's prediction and the inferred reference label. In our implementation, feedback includes up to three class pairs, each supported by at least five disagreements with posterior confidence of at least 0.8. Each instruction may include up to two examples, selected as the first two samples encountered in the current batch that exhibit the corresponding predicted–reference label pair and satisfy the confidence threshold. Their textual content is truncated to 180 characters. Example lines are omitted when textual content is unavailable. The resulting feedback is appended to the same annotator's prompt in the next round without changing the required output format. No feedback is provided in the first round or when no eligible pair is found. The reference labels are inferred estimates rather than ground truth, so the feedback serves as guidance rather than verified corrections.