(iii) Regression loss $\mathcal{L}_{regress}$. For each pair (i, j) we have a ground-truth indirect score trueindirectscore$_{ij} \in [0, 1]$. Let

$$e_{ij} = I_{\text{learned}}(i, j) - \text{trueindirectscore}_{ij}.$$

We use the Huber loss with parameter $\delta > 0$:

$$\mathcal{L}_{\text{regress}}(i, j) = \begin{cases} \frac{1}{2}e_{ij}^2, & |e_{ij}| \leq \delta, \\ \delta|e_{ij}| - \frac{1}{2}\delta^2, & |e_{ij}| > \delta. \end{cases}$$

Averaging over all training pairs yields $\mathcal{L}_{\text{regress}}$.

Output. A learned path-aggregation model $LPA^{(0)}_{\text{model}}$ consisting of a path encoder and an attention-based aggregator capable of mapping sets of paths to calibrated indirect scores.

(c) **The EM-Refinement Loop**

Goal. The objective of this step is to refine the pre-trained (v0.1) models from Step 4.2 on the real, unlabeled document corpus by using their own predictions as a self-supervised signal. The refinement is formulated as an Expectation–Maximization (EM) style loop with a Mean-Teacher architecture to ensure stability.

We maintain two sets of parameters for each learnable component (e.g., fusion model, path aggregator):

- *Student parameters* θ_S: actively updated by gradient-based optimization.
- *Teacher parameters* θ_T: updated as an exponential moving average (EMA) of θ_S and used to generate pseudo-labels.

For concreteness, let $\theta_{S,\text{fusion}}$ and $\theta_{T,\text{fusion}}$ denote the parameters of the student and teacher fusion models, and analogously for the LPA model.

We perform R refinement rounds:

$$\text{for } r = 1, \ldots, R.$$

A. E-Step (Expectation / Pseudo-Label Estimation).

Goal. For a fixed round r, the E-step uses the current teacher models to compute the best available estimate of the causal answer key $C^{(r)}_{\text{prior}}$ on the real corpus.

Procedure.

i. We run the complete CoCaD pipeline (Phase 3: Steps 3.1–3.3) on the real document corpus, using the teacher parameter sets from the previous round, $\theta_T^{(r-1)}$, for all learnable modules (fusion model, LPA model, CPC model, retriever, etc.). This yields, for each candidate pair $(i, j) \in E_{\text{prior}}$, a final score and associated statistics:

$$(i, j) \mapsto \big(\text{finalscore}^{(r)}_{ij},\, p(i, j),\, v_{ij},\, H_p(i, j),\, \ldots\big),$$

where $p(i, j)$ is the empirical p-value from Step 3.3 and v_{ij} is the feature vector used by the fusion model.

ii. In round $r = 1$, the conditional LLM component in Step 3.1 is executed explicitly to obtain scores $\mu_{\text{LLMcond}}(v_{ij})$ for all (i, j), and we store the pairs

$$\mathcal{D}_{\text{distill}} = \{(v_{ij},\, \mu_{\text{LLMcond}}(v_{ij}))\}.$$

iii. In the M-step of round 1 (see below), we train a student surrogate model f_{student} to approximate the LLM scores. In subsequent rounds $r > 1$, the expensive LLM calls in Step 3.1 are deterministically replaced by $f_{\text{student}}(v_{ij})$, so that the E-step remains computationally feasible while preserving the interface of the pipeline.

The output of the E-step at round r is the sparse answer key

$$C^{(r)}_{\text{prior}} = \{(i, j, \text{finalscore}^{(r)}_{ij}, p(i, j), \ldots)\}.$$

B. M-Step (Maximization / Student Update).

Goal. For fixed pseudo-labels $C^{(r)}_{\text{prior}}$, the M-step updates the student parameters $\theta^{(r)}_S$ by minimizing a combination of pseudo-label losses and consistency regularization on real data.

Soft, Confidence-Weighted Pseudo-Labels. For each pair (i, j) in $C^{(r)}_{\text{prior}}$ we define:

$$\begin{aligned} y^{\text{soft}}_{ij} &= \text{finalscore}^{(r)}_{ij} \in [0, 1], \\ w_{ij} &= 1 - p(i, j), \end{aligned}$$

where w_{ij} acts as a confidence weight: highly significant links (small $p(i, j)$) receive weights close to 1, whereas non-significant links receive weights close to 0.

We also define a hard binary label for pruning decisions,

$$y^{\text{bin}}_{ij} = \begin{cases} 1, & \text{if finalscore}^{(r)}_{ij} > 0, \\ 0, & \text{otherwise.} \end{cases}$$

B.1. Fusion Model Update.

The student fusion model $f^{(r)}_{\text{fusion,S}}$ with parameters $\theta^{(r)}_{S,\text{fusion}}$ maps feature vectors v_{ij} to scores in $(0, 1)$. We update $\theta^{(r)}_{S,\text{fusion}}$ by minimizing a confidence-weighted binary cross-entropy loss:

$$\mathcal{L}_{\text{fusion-real}} = \sum_{(i,j)\in C^{(r)}_{\text{prior}}} w_{ij} \cdot \text{BCE}\big(f^{(r)}_{\text{fusion,S}}(v_{ij}),\, y^{\text{soft}}_{ij}\big),$$

where $\text{BCE}(p, y) = -\left(y \log p + (1 - y) \log(1 - p)\right)$.

Gradients of $\mathcal{L}_{\text{fusion-real}}$ with respect to $\theta^{(r)}_{S,\text{fusion}}$ are computed and applied using standard stochastic gradient descent or Adam.

B.2. LPA Model Update.

The student LPA model $LPA^{(r)}_{\text{S}}$, with parameters $\theta^{(r)}_{S,\text{LPA}}$, outputs for each pair (i, j) an aggregated indirect score $I^{(r)}_{\text{learned,S}}(i, j) \in (0, 1)$. In the refinement loop we treat this as a classifier for pruning and update it using the binary pseudo-labels:

$$\mathcal{L}_{\text{LPA-real}} = \sum_{(i,j)\in C^{(r)}_{\text{prior}}} w_{ij} \cdot \text{BCE}\big(I^{(r)}_{\text{learned,S}}(i, j),\, y^{\text{bin}}_{ij}\big).$$

The parameters $\theta^{(r)}_{S,\text{LPA}}$ are updated by backpropagation on this loss.

B.3. Distillation of the Conditional LLM (First Round Only).

In round $r = 1$, we additionally train a student surrogate f_{student} on the distillation set $\mathcal{D}_{\text{distill}}$:

$$\mathcal{L}_{\text{distill}} = \sum_{(v_{ij}, t_{ij}) \in \mathcal{D}_{\text{distill}}} \| f_{\text{student}}(v_{ij}) - t_{ij} \|_2^2,$$

where $t_{ij} = \mu_{\text{LLMcond}}(v_{ij})$ is the original LLM score. The weights of f_{student} are updated via gradient descent on $\mathcal{L}_{\text{distill}}$, and the trained f_{student} is then used in place of the LLM in all subsequent rounds.

B.4. Consistency Regularization.

To improve robustness to perturbations and prevent overfitting to noisy pseudo-labels, we introduce a consistency loss for each student model $f^{(r)}_{\text{student,S}}$ (this notation can refer generically to the fusion model, LPA model, or other learnable components). For a given (i, j), we construct two stochastic augmentations of the input, v_{ij}^{aug1} and v_{ij}^{aug2} (e.g., via feature dropout, small noise in embeddings) and enforce consistency between their predictions:

$$\mathcal{L}_{\text{consistency}} = \sum_{(i,j)} \left\| f^{(r)}_{\text{student,S}}(v_{ij}^{\text{aug1}}) - f^{(r)}_{\text{student,S}}(v_{ij}^{\text{aug2}}) \right\|_2^2.$$

B.5. Total M-Step Objective.

For each student model we define the total M-step loss as

$$\mathcal{L}_M = \mathcal{L}_{\text{pseudo-label}} + \lambda_{\text{consist}} \cdot \mathcal{L}_{\text{consistency}},$$

where $\mathcal{L}_{\text{pseudo-label}}$ denotes the corresponding pseudo-label loss (e.g., $\mathcal{L}_{\text{fusion-real}}$ for the fusion model, $\mathcal{L}_{\text{LPA-real}}$ for the LPA model), and λ_{consist} is a hyperparameter that controls the strength of consistency regularization. The student parameters $\theta_S^{(r)}$ are updated by minimizing $\mathcal{L}_M$.

C. Teacher Update (EMA).

Goal. After updating the student parameters $\theta_S^{(r)}$, we update the teacher parameters $\theta_T^{(r)}$ via an exponential moving average to obtain a smoother, more stable model that will be used in the next E-step.

For each parameter vector (e.g., for the fusion or LPA model) we apply:

$$\theta_T^{(r)} \leftarrow \alpha \cdot \theta_T^{(r-1)} + (1 - \alpha) \cdot \theta_S^{(r)},$$

where $\alpha \in (0, 1)$ is a high momentum coefficient, typically close to 1 (e.g., $\alpha = 0.999$). This update ensures that the teacher evolves slowly and acts as a low-variance target for pseudo-label generation.

D. Stopping Criteria.

The EM-refinement loop over r is terminated based on a combination of label stability and validation performance:

- *Label stability:* Let $\mathcal{H}^{(r)}$ be the set of high-confidence links in $C^{(r)}_{\text{prior}}$ (e.g., links with w_{ij} above a fixed threshold). We compute the Jaccard similarity

$$J^{(r)} = \frac{|\mathcal{H}^{(r)} \cap \mathcal{H}^{(r-1)}|}{|\mathcal{H}^{(r)} \cup \mathcal{H}^{(r-1)}|}$$

  and stop if $J^{(r)}$ exceeds a threshold (e.g., 0.99), indicating that the answer key has stabilized.

- *Validation plateau:* We monitor the total loss $\mathcal{L}_M$ and other relevant metrics (e.g., calibration error) on a held-out synthetic validation set from Step 4.1. Training stops if these metrics do not improve for a fixed number of rounds.

Final Output of EM Refinement

After convergence, the final teacher parameter sets $\theta_T^{(\text{final})}$ define the refined models used in the online CoCaD inference pipeline. Concretely, we obtain:

$$f_{\text{fusion}}^{(\text{final})}, \quad LPA_{\text{model}}^{(\text{final})}, \quad CPC_{\text{model}}^{(\text{final})}, \quad f_{\text{student}}^{(\text{final})},$$

each of which has been initialized on synthetic data and subsequently adapted to the real corpus via the EM-refinement procedure described above.