LANGUAGE-GUIDED GRAPH REASONING FOR CONCEPT BOTTLENECK MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present MoE-SGT, a concept bottleneck model that combines editable concept scores with offline question–answer (QA) knowledge. Cached answer–question graphs encode class semantics; answer–concept graphs combine sample-dependent concept evidence with semantic and co-occurrence priors. Present and absent concept embeddings let corrected scores update predictions without retraining. A structure-aware router selects two experts alongside a shared expert. MoE-SGT achieves the highest Top-1 accuracy among the evaluated CBMs on all four single-label datasets, including CUB-200, and the highest ROC-AUC and F1 on both medical datasets. On CUB-200, graph reasoning raises accuracy from 75.15% with the linear head to 80.31%. With ten concept corrections, accuracy reaches 86.39% under graph-informed selection and 89.92% under oracle selection.

1 INTRODUCTION

Fine-grained recognition and medical image analysis benefit from predictions whose supporting attributes can be inspected and corrected. Concept bottleneck models (CBMs) provide this interface by predicting human-readable concepts before assigning a class label (Koh et al., 2020). Concept corrections can change the final decision through this pathway. Extensions address editable, incremental, and semi-supervised concept representations (Hu et al., 2024; Shang et al., 2024; Hu et al., 2025). Language and vision–language models broaden the available concept vocabularies and supply image–concept supervision (Oikarinen et al., 2023; Srivastava et al., 2024; Radford et al., 2021), while joint distributions and learned graphs capture dependencies among concepts (Vandenhirtz et al., 2024; Xu et al., 2026). An attribute's relevance depends on the candidate class, and its absence can be as informative as its presence. For example, a bird's wing color and bill shape jointly distinguish a species, so correcting one attribute should update how the remaining evidence supports that class. Class-level language descriptions relate these attributes to candidate classes; concept corrections update the evidence used to score them.

We introduce MoE-SGT (Figure 1) to combine class-level language knowledge with sample-specific concept evidence through answer-conditioned graphs. Each candidate class is represented by offline question–answer (QA) text, and its graph receives the concepts most relevant to that class. The QA branch encodes shared class semantics; the concept branch carries the current image evidence through editable scores. Present and absent concept embeddings retain both forms of evidence, and corrections propagate through candidate retrieval and graph reasoning. MoE-SGT outperforms the evaluated CBMs on all six classification benchmarks. Our three contributions are: **(1)** We construct answer–concept and answer–question graphs that fuse cached class-level QA semantics with sample-dependent concept evidence, using candidate-class selection and class-relevant concept subgraphs to organize classification. **(2)** We build an editable graph interface with present/absent concept embeddings and concept-conditioned affine transformations. The interface supports graph-informed query selection and prediction updates without retraining. **(3)** We design a structure-aware sparse mixture-of-experts block with a shared expert and top-2 routing. Node semantics, node types, and structural context determine routing through an expert pool shared across the three graph families. The sparse expert block improves on a dense FFN matched in total parameters.

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.

2 Related Work

2.1 Concept Bottleneck Models

CBMs connect prediction and intervention by using explicit concepts as an intermediate representation (Koh et al., 2020). CEM replaces scalar concept representations with embeddings while retaining a concept-level intervention interface (Espinosa Zarlenga et al., 2022), and probabilistic CBMs represent uncertainty in concept predictions (Kim et al., 2023a). SCBM models concept dependencies through a joint distribution, allowing interventions to inform correlated concepts (Vandenhirtz et al., 2024). Post-hoc CBMs retrofit concept-based predictors to pretrained networks and support model editing through concepts (Yuksekgonul et al., 2023). MoE-SGT uses scalar concept scores to gate present and absent semantic embeddings, so corrected scores directly change the node features used for graph reasoning. For concept acquisition, language-guided CBMs reduce dependence on manually annotated concepts by drawing on pretrained vision–language representations such as CLIP (Radford et al., 2021). Label-free CBM constructs a concept bottleneck without concept annotations (Oikarinen et al., 2023); VLG-CBM uses grounded object detectors to obtain visually grounded concept supervision (Srivastava et al., 2024); and Sparse-CBM learns sparse concept activations (Semenov et al., 2024). Cross-modal learning (Alukaev et al., 2023) and visual concept filtering (Kim et al., 2023b) address concept acquisition and selection. In MoE-SGT, concept descriptions define the vocabulary, and class-level QA text supplies context for combining predicted concepts. GraphCBMs learn latent concept graphs and update concept embeddings and activations through message passing, supporting interventions in both concept-supervised and label-free settings (Xu et al., 2026). MoE-SGT organizes this relational computation around candidate answers. Answer–concept graphs associate editable evidence with each class, while answer–question graphs encode its cached language knowledge. Class-specific concept selection retains discriminative attributes regardless of their predicted presence or absence, so the same graph interface supports classification and concept correction.

2.2 Graph-Based Multimodal Reasoning

Graph Transformers incorporate structural information into attention to combine relational priors with learned feature interactions (Zhao et al., 2021; Min et al., 2022). Heterogeneous Graph Transformers distinguish node and relation types (Hu et al., 2020). For visual commonsense reasoning, MSGT constructs answer–vision and answer–question graphs and injects spatial and linguistic structure into graph attention (Zhu et al., 2024). We adopt its answer-centered organization for concept-based classification. In MoE-SGT, semantic similarity and concept co-occurrence define structural priors, while editable concept scores supply sample-dependent node evidence. Concept corrections update these node features. Mixture-of-experts models use a gating function to combine specialized predictors (Masoudnia & Ebrahimpour, 2014). GShard and Switch Transformers scale conditional computation by routing tokens to a small subset of experts (Lepikhin et al., 2020; Fedus et al., 2022). MoE-SGT applies sparse routing within graph Transformer layers, where concept, answer, and question nodes have distinct semantic roles. The router combines node features with node type and structural context to distinguish these roles and neighborhoods. A shared expert transforms every node; routed experts share parameters across graph families.

3 Concept Evidence and Language Knowledge

For an input $x = \{x^{(v)}\}_{v=1}^{V}$ with V image views, we predict scores over K concepts $\mathcal{C} = \{c_i\}_{i=1}^{K}$ and classify among C labels $\mathcal{Y} = \{1, \dots, C\}$. A single-label target is $y \in \mathcal{Y}$; a multi-label target is $\mathbf{y} \in \{0,1\}^C$. A single image has $V = 1$.

3.1 Offline Knowledge and Concept Supervision

CUB-200 uses a fixed pool of 112 official attributes. For other datasets, an LLM extracts concepts from training reports or generates them from class names. Frozen aligned text/image encoders g_T, g_I produce normalized concept embeddings $\mathbf{t}_i = g_T(c_i)/\|g_T(c_i)\|_2$ and view features $\mathbf{v}^{(v)} = g_I(x^{(v)})/\|g_I(x^{(v)})\|_2$, both in $\mathbb{R}^{d_e}$; d_e is the encoder dimension and $\|\cdot\|_2$ the Euclidean norm. Concept-pool construction is detailed in Appendix A.1.

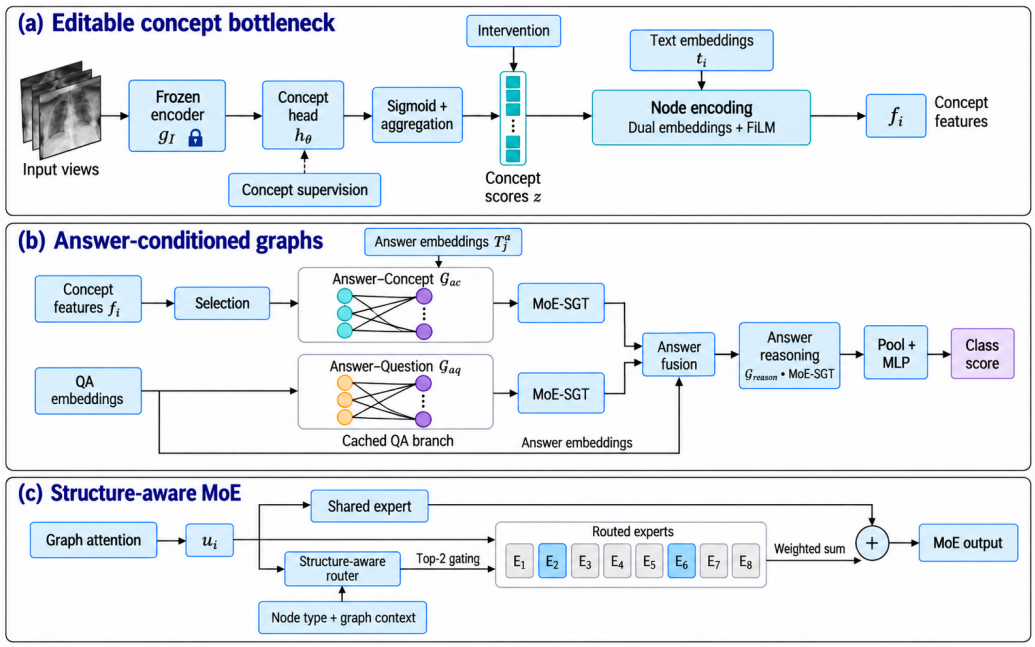


Figure 1: Overview of MoE-SGT. (a) Frozen image features yield editable concept scores and dual embeddings modulated by FiLM. The dashed arrow denotes training supervision. (b) For each candidate class, we combine concepts selected by class-weight magnitude with cached QA semantics through answer fusion and graph reasoning. (c) Node features, types, and structural context determine top-2 routing among eight experts; a shared expert processes every node. Expert pools are shared across graph families at matching layers.

For class j, offline QA knowledge consists of attribute-oriented questions Q_j and an answer A_j containing the class name and associated concepts. Text encoding and projection give $T_j^q \in \mathbb{R}^{n_j^q \times d}$ and $T_j^a \in \mathbb{R}^{n_j^a \times d}$, where n_j^q, n_j^a count question/answer tokens and d is the graph width. Questions specify which attributes to examine; answer descriptions relate them to the candidate class. We generate and cache the text offline without test images or test labels. Medical image–text alignment uses MedCLIP (Wang et al., 2022).

A head $h_\theta : \mathbb{R}^{d_e} \to \mathbb{R}^K$ with parameters θ predicts concept logits. We standardize each concept similarity using training-set statistics, then apply a temperature-scaled sigmoid to control the spread of teacher scores. The probability-preserving aggregator $\mathcal{A}_\psi$, with parameters ψ, fuses views and is the identity for $V = 1$:

$$
\begin{aligned}
p_i(x) &= \mathcal{A}_\psi\left(\{\sigma([h_\theta(\mathbf{v}^{(v)})]_i)\}_{v=1}^{V}\right), \\
\pi_i(x) &= \mathcal{A}_\psi\left(\left\{\sigma\left(\frac{\mathbf{v}^{(v)\top}\mathbf{t}_i - \mu_i}{\tau_\pi \nu_i}\right)\right\}_{v=1}^{V}\right), \\
\mathbf{z}(x) &= [p_1(x), \ldots, p_K(x)]^\top.
\end{aligned}
\tag{1}
$$

Here, μ_i and ν_i are the mean and standard deviation of concept i's cosine similarities across training views; temperature $\tau_\pi > 0$ controls target sharpness. These quantities are fixed during bottleneck training; calibration details appear in Appendix A.1. The function σ is logistic sigmoid, $[\cdot]_i$ selects coordinate i, $p_i \in [0,1]$ is a predicted concept score, $\pi_i \in [0,1]$ a soft supervision prior, and $\mathbf{z} \in [0,1]^K$ the editable bottleneck. The priors express relative visual support; an available binary annotation a_i supplies an explicit presence/absence target. Thus the training target $\bar{\pi}_i$ equals a_i when available and otherwise equals π_i. We convert human text into explicit concept/value corrections (Figure 2).

3.2 Concept-Conditioned Node Features

Following CEM (Espinosa Zarlenga et al., 2022), we use separate present/absent embeddings to retain semantic evidence at both ends of the concept-score range. For concept score $z_i = z_i(x)$, we mix the two embeddings as $\mathbf{e}_i = z_i W^+ \mathbf{t}_i + (1 - z_i) W^- \mathbf{t}_i$, where $W^+, W^- \in \mathbb{R}^{d \times d_e}$ encode presence/absence. A network $F_\eta : \mathbb{R}^{1+d_e} \to \mathbb{R}^{2d}$, with parameters η, predicts $(\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i) = F_\eta([z_i; \mathbf{t}_i])$ from the score and concept text. These scale/bias vectors $\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i \in \mathbb{R}^d$ produce the final feature $\mathbf{f}_i = \boldsymbol{\gamma}_i \odot \mathbf{e}_i + \boldsymbol{\beta}_i$. Both $\mathbf{e}_i, \mathbf{f}_i \in \mathbb{R}^d$; brackets denote concatenation and $\odot$ element-wise multiplication. Setting $\boldsymbol{\gamma}_i = \mathbf{1}$, the all-ones vector, gives additive-only conditioning. Image information enters this mapping through z_i, while the fixed text embeddings supply concept semantics. A correction therefore propagates through both the present/absent mixture and its affine modulation.

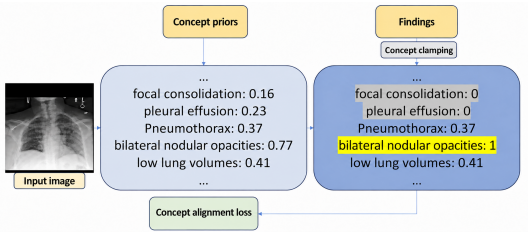


Figure 2: Example of concept-score clamping during the training intervention phase: after fusing multi-view priors, explicitly present (or absent) concepts are clamped to 1 or 0.

4 Answer-Conditioned Graph Classification

4.1 Candidate Graphs and Structural Priors

Preliminary logits $\mathbf{s}^{\text{lin}}(x) = W_l \mathbf{z}(x) + \mathbf{b}_l$ use weights $W_l \in \mathbb{R}^{C \times K}$ and bias $\mathbf{b}_l \in \mathbb{R}^C$. Single-label inference reranks the top-M classes, defining candidate set $\mathcal{Y}_x$, with $M \leq C$; training uses the true class plus the $M - 1$ highest-scoring negatives. The linear head concentrates graph computation on plausible classes, while the graph scores each candidate within its own QA context. Candidate recall therefore bounds the accuracy attainable by reranking. Multi-label inference scores every pathology independently, so $\mathcal{Y}_x = \mathcal{Y}$. A class can be distinguished by both present and absent attributes. We therefore select concept indices $S_m^{(j)} = \text{Top}_m(|[W_l]_{j,:}|)$ for candidate j, where Top_m returns the indices of the $m \leq K$ largest magnitudes in row j of W_l; ties use concept index. This selection uses the preliminary classifier's coefficient magnitudes to retain class-relevant concepts irrespective of their current scores. A low-scoring but class-relevant concept can thus contribute negative evidence. Correcting its score to zero retains its node within that class's graph and activates the absent embedding. Before attention, we gather their features and prior submatrices to form answer–concept graph $G_{ac}^{(j)}$ and answer–question graph $G_{aq}^{(j)}$. Their initial features are $H_{ac}^{(j,0)} = [F_{S_m^{(j)}}; T_j^a]$ and $H_{aq}^{(j,0)} = [T_j^q; T_j^a]$, where $F_{S_m^{(j)}} \in \mathbb{R}^{m \times d}$ stacks the selected $\mathbf{f}_i$, and brackets stack rows. The graphs have $m + n_j^a$ and $n_j^q + n_j^a$ nodes; competing classes have separate answer nodes. During training, $S_m^{(j)}$ follows the current W_l; after training, each class has a fixed concept set, while node features vary with the sample.

For concept indices i, k, relations combine cosine similarity $s_{ik} = \mathbf{t}_i^\top \mathbf{t}_k$ and smoothed training-set pointwise mutual information $P_{ik} = \log \frac{\widehat{P}_{ik} + \epsilon}{\widehat{P}_i \widehat{P}_k + \epsilon}$. Here, $\widehat{P}_i, \widehat{P}_k$ are empirical marginal presence probabilities, $\widehat{P}_{ik}$ their joint probability, and $\epsilon > 0$ a smoothing constant. The co-occurrence feature P_{ik} is omitted when training concept annotations are unavailable. Text similarity captures semantic relatedness, while co-occurrence reflects associations among attributes in the training data. These priors guide graph attention with both linguistic and empirical relations among concepts. The bias between nodes u, v is $E_{uv} = \Psi_{\tau(u),\tau(v)}(\mathbf{b}_{uv})$: τ gives concept/question/answer type, Ψ is a learned scalar map for that type pair, and $\mathbf{b}_{uv}$ is a static relation descriptor. It contains (s_{ik}, P_{ik}) for concept pairs, token similarity and relative position within text sequences, and text similarity for cross-type pairs. This yields square priors $E_{ac}^{(j)}$ and $E_{aq}^{(j)}$ matching the respective graph sizes. These initial relation priors are shared across samples for a fixed candidate class, while node features carry the current concept evidence. A correction updates this evidence, and graph attention recomputes sample-dependent interactions from the same initial priors.

4.2 Graph Attention and Sparse Experts

Attention combines structural biases with sample-dependent node evidence. At layer ℓ, a graph with N_g nodes has features $H^{(\ell)} \in \mathbb{R}^{N_g \times d}$ and bias $B^{(\ell)} \in \mathbb{R}^{N_g \times N_g}$. One attention head computes

$$A^{(\ell)} = \mathrm{softmax}_{\mathrm{row}}\left(B^{(\ell)} + \frac{(H^{(\ell)}W_Q^{(\ell)})(H^{(\ell)}W_K^{(\ell)})^\top}{\sqrt{d_h}}\right), \qquad B^{(0)} = E. \tag{2}$$

Here, E is the graph's initial prior; $W_Q^{(\ell)}, W_K^{(\ell)}, W_V^{(\ell)} \in \mathbb{R}^{d \times d_h}$ project queries, keys, and values to head width d_h. Row-normalized attention $A^{(\ell)}$ aggregates values as $A^{(\ell)}H^{(\ell)}W_V^{(\ell)}$; concatenating and projecting heads gives $U^{(\ell)} \in \mathbb{R}^{N_g \times d}$. The next bias is $B^{(\ell+1)} = \Psi_{\mathrm{sgt}}(s_{\mathrm{sgt}}\bar{A}^{(\ell)})$, where $\bar{A}^{(\ell)}$ averages heads, s_{sgt} is a scale, and Ψ_{sgt} a learned element-wise map.

The feed-forward block contains one shared expert $\mathcal{E}_s$ and $N_e = 8$ routed experts $\mathcal{E}_e$, each a $d \to 2d \to d$ network. The shared expert transforms every node, while the router selects two additional experts for each node. Nodes with similar features can occupy different relational roles; structural context lets routing distinguish these roles. For node row $\mathbf{u}_i \in \mathbb{R}^d$ of $U^{(\ell)}$, the router combines semantics, type, and structural context:

$$\begin{aligned} \bar{\mathbf{e}}_i &= \sum_{j=1}^{N_g} [\mathrm{softmax}_{\mathrm{row}}(B^{(\ell)})]_{ij}\, \mathrm{LN}(\mathbf{u}_j), \\ \mathbf{r}_i &= W_r[\mathrm{LN}(\mathbf{u}_i); \mathbf{t}_i^{\mathrm{type}}; \bar{\mathbf{e}}_i]. \end{aligned} \tag{3}$$

The operator LN denotes layer normalization. The vector $\bar{\mathbf{e}}_i \in \mathbb{R}^d$ contains structural context, and $\mathbf{t}_i^{\mathrm{type}} \in \mathbb{R}^{d_t}$ is a learned embedding of type $\tau(i)$. The matrix $W_r \in \mathbb{R}^{N_e \times (2d+d_t)}$ maps these inputs to router logits $\mathbf{r}_i \in \mathbb{R}^{N_e}$. The symbol d_t denotes type-embedding width; router/expert layer superscripts are suppressed.

Let $\tilde{\mathbf{r}}_i$ denote router logits with independent Gaussian noise during training and the noise-free logits at inference (Appendix A.2). The set $\mathcal{T}_i$ contains the top-κ noisy-logit indices, with $\kappa = 2$:

$$\begin{aligned} \mathrm{MoE}(\mathbf{u}_i) &= \mathcal{E}_s(\mathbf{u}_i) + \sum_{e \in \mathcal{T}_i} G_{i,e}\mathcal{E}_e(\mathbf{u}_i), \\ G_{i,e} &= \begin{cases} \dfrac{\exp(\tilde{r}_{i,e})}{\sum_{j \in \mathcal{T}_i}\exp(\tilde{r}_{i,j})}, & e \in \mathcal{T}_i, \\ 0, & e \notin \mathcal{T}_i. \end{cases} \end{aligned} \tag{4}$$

The weight $G_{i,e}$ normalizes noisy logits $\tilde{r}_{i,e}$ over selected experts only; e, j index routed experts. Every node passes through the shared expert. Router and expert parameters are shared across graph families at matching layer indices and independent across layers. Sharing lets concept and text nodes reuse transformations and keeps the expert-pool size independent of the number of graph families. Excluding biases and router overhead, the expert block uses $1.5\times$ the active weights and $4.5\times$ the total weights of a dense $d \to 4d \to d$ FFN; implementation and parameter accounting appear in Appendix A.2.

The answer–question branch encodes fixed class semantics, which we fuse with sample-dependent answer–concept features. After D_c contextualization layers, answer rows $H_{ac,A}^{(j)}, H_{aq,A}^{(j)} \in \mathbb{R}^{n_j^a \times d}$ are fused and classified:

$$\begin{aligned} R_j &= \mathrm{concat}_{\mathrm{feat}}(T_j^a, H_{ac,A}^{(j)}, H_{aq,A}^{(j)})W_f, \\ H_j^{\mathrm{cls}} &= \Phi_{D_r}(R_j, E_{AA}^{(j)}), \qquad s_j(x) = f_{\mathrm{out}}\left(\frac{1}{n_j^a}\sum_{u=1}^{n_j^a}[H_j^{\mathrm{cls}}]_{u,:}\right). \end{aligned} \tag{5}$$

Feature-column concatenation and $W_f \in \mathbb{R}^{3d \times d}$ produce $R_j \in \mathbb{R}^{n_j^a \times d}$. The answer-reasoning graph $G_{\mathrm{reason}}^{(j)} = (R_j, E_{AA}^{(j)})$ uses the static answer–answer block $E_{AA}^{(j)}$ of $E_{aq}^{(j)}$. Its D_r-layer stack Φ_{D_r} returns $H_j^{\mathrm{cls}} \in \mathbb{R}^{n_j^a \times d}$; $[\cdot]_{u,:}$ selects row u, and MLP $f_{\mathrm{out}} : \mathbb{R}^d \to \mathbb{R}$ maps the row mean to class logit

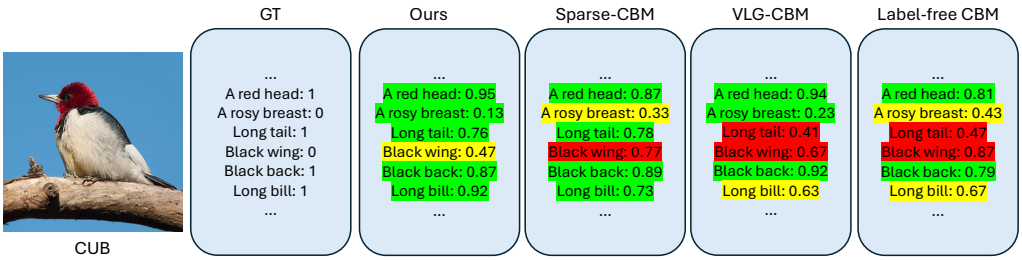


Figure 3: CUB-200 concept predictions from MoE-SGT, Sparse-CBM (Semenov et al., 2024), VLG-CBM (Srivastava et al., 2024), and Label-free CBM (Oikarinen et al., 2023). GT denotes binary human-annotated attributes; the remaining columns show predicted concept scores.

s_j. Single-label prediction uses softmax over candidates; multi-label prediction uses independent sigmoids.

After training, we cache the text-only answer–question branch and class-specific concept sets and priors. Concept features and first-layer key/value projections can be reused for concepts shared by candidate graphs; subsequent representations are answer-dependent. At fixed width, answer–concept attention costs $O((m + n_j^a)^2)$ per candidate per layer.

4.3 Training Objective

Task losses $\mathcal{L}_{\text{graph}}, \mathcal{L}_{\text{linear}}$ use cross-entropy for single-label classification and binary cross-entropy for multi-label classification. The graph loss uses training candidates; the linear loss uses all classes to learn candidate retrieval and the coefficients used for concept selection. We optimize

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda_1 \mathcal{L}_{\text{CBL}} + \lambda_2 \mathcal{L}_{\text{balance}}, \qquad \lambda_2 = 0.01. \tag{6}$$

The total loss $\mathcal{L}$ combines classification $\mathcal{L}_{\text{cls}}$, bottleneck training $\mathcal{L}_{\text{CBL}}$ weighted by $\lambda_1 \geq 0$, and Switch-style routing balance $\mathcal{L}_{\text{balance}}$ weighted by λ_2 (Fedus et al., 2022). Classification combines graph and linear-head losses; bottleneck training aligns concept scores with annotation/prior targets and sparsifies W_l. Routing balance pools tokens across all samples and graphs at each shared expert layer and excludes the shared expert. Appendix A.3 gives the loss components, normalization, and gradient treatment.

4.4 Concept Intervention

For corrected concept indices I and supplied values $a_i \in \{0, 1\}$, set $z_i^I(x) = a_i$ for $i \in I$ and $z_i^I(x) = z_i(x)$ for $i \notin I$, keeping model parameters fixed. The corrected vector $\mathbf{z}^I$ updates node features, candidate classes, and graph predictions. Each candidate retains its fixed set $S_m^{(j)}$: correcting a selected concept to zero keeps its node and activates its absent embedding. Newly retrieved classes use their own cached sets and priors. For a budget of k corrections, oracle selection ranks erroneous concepts by $|z_i - a_i|$, where an error means $\mathbb{I}[z_i \geq 0.5] \neq a_i$. Ties use graph importance, then concept index. Graph-informed selection ranks unqueried concepts by $b_i^{(j)} = (1 - |2z_i - 1|)\,\text{imp}_i^{(j)}$, multiplying uncertainty $1 - |2z_i - 1|$ by importance $\text{imp}_i^{(j)} = \frac{1}{n_j^a} \sum_{u \in A_j} \bar{A}_{ac,ui}^{(j,\text{last})}$. Here, $\bar{A}_{ac,ui}^{(j,\text{last})}$ is head-averaged attention from answer token $u \in A_j$ to concept i in the final contextualization layer. Importance averages this attention over answer tokens and is zero for inactive concepts. Target j is the predicted class or inspected pathology. Uncertainty-only selection sets $\text{imp}_i^{(j)} = 1$ for every unqueried concept, giving score $1 - |2z_i - 1|$. Random selection samples unqueried concepts uniformly without replacement. All policies apply supplied values through the same correction rule. We recompute predictions, candidate classes, and attention after each correction before selecting the next concept.

Table 1: Top-1 accuracy (%) on four single-label datasets. The frozen-feature probe uses the same backbone features; the linear-head row evaluates s_{lin} from the full model. Best CBM results are shown in bold.

Model	CUB-200 (Accuracy %)	ImageNet (Accuracy %)	CIFAR-10 (Accuracy %)	CIFAR-100 (Accuracy %)
Frozen-feature probe	81.24	79.64	95.33	82.41
MoE-SGT (s_{lin} only)	75.15	69.82	88.93	72.69
MoE-SGT (full)	**80.31**	**73.41**	**92.41**	**76.89**
Sparse-CBM (Semenov et al., 2024)	79.97	71.32	91.08	74.23
Label-free CBM (Oikarinen et al., 2023)	71.32	71.62	86.04	65.07
VLG-CBM (Srivastava et al., 2024)	73.11	67.19	81.72	63.17
SCBM (Vandenhirtz et al., 2024)	69.82	67.22	70.67	59.31
CEM (Espinosa Zarlenga et al., 2022)	70.37	68.63	72.35	60.79
CBM (Koh et al., 2020)	69.08	65.84	69.38	58.11

5 Experiments

5.1 Datasets and Settings

We evaluate single-label classification on CUB-200 (Wah et al., 2011), ImageNet (Deng et al., 2009), and CIFAR-10/100 (Krizhevsky, 2009), and multi-label classification on MIMIC-CXR (Johnson et al., 2019) and CheXpert (Irvin et al., 2019). CUB uses the 112 filtered official attributes; available annotations replace concept-prior targets during training, and uncertain medical labels are treated as negative. All models use a frozen ViT-B/16 backbone, with CLIP (Radford et al., 2021) or MedCLIP (Wang et al., 2022) for general or medical image–text alignment; DeepSeek-R1-32B (DeepSeek-AI, 2025) generates concepts and QA pairs offline. The graph modules use $D_c = 2$, $D_r = 1$, $d = 256$, and four heads ($d_h = 64$); per-dataset concept counts and candidate and active-concept budgets are listed in Appendix A.4. Baselines (Sparse-CBM, Label-free CBM, VLG-CBM, SCBM, CEM, CBM) use identical splits. On CUB they share the 112-attribute vocabulary under their own concept-supervision protocols; elsewhere, concept-mining baselines follow their original procedures and CEM and CBM use our concept pool. All results use one training seed; random-policy and noisy-feedback evaluations average five draws.

5.2 Single-Label Classification Results

MoE-SGT outperforms all six CBM baselines on every single-label dataset (Table 1); the closest margin is 0.34 points over Sparse-CBM on CUB-200 under the same 112 attributes. Relative to its own linear head s_{lin}, graph classification adds 5.16, 3.59, 3.48, and 4.20 points on CUB-200, ImageNet, CIFAR-10, and CIFAR-100. Candidate recall, the fraction of test examples whose true class lies among the top-$\min(M, C)$ linear-head scores, upper-bounds reranked accuracy; with $M = 20$ for ImageNet and $M = 10$ otherwise it is at least 96.55% on every dataset (Appendix Table A.2).

5.3 Multi-Label Classification Results

We train every model with a sigmoid head, one output per pathology, and binary cross-entropy for 50 epochs on the official splits. Frontal and lateral views are scored separately and combined by element-wise maximum; "uncertain" labels count as negative. In Table 2, MoE-SGT reaches 0.76 ROC-AUC and 0.80 F1 on MIMIC-CXR and 0.83 and 0.86 on CheXpert, compared with 0.72/0.73 and 0.80/0.81 for the strongest CBM baseline, Sparse-CBM. Relative to its own linear head, graph classification adds 0.05 ROC-AUC and 0.07 F1 on MIMIC-CXR and 0.04 and 0.06 on CheXpert.

5.4 Concept Prediction Performance

All models predict the same 112 official CUB attributes. We threshold concept scores at 0.5 and report balanced accuracy (the mean of positive- and negative-class recall) and F1 for attribute presence,

Table 2: ROC-AUC and F1 on two multi-label datasets. The frozen-feature probe uses the same backbone features; the linear-head row evaluates s_{lin} from the full model. Best CBM results are shown in bold.

Model	MIMIC-CXR (AUC)	MIMIC-CXR (F1)	CheXpert (AUC)	CheXpert (F1)
Frozen-feature probe	0.79	0.83	0.84	0.88
MoE-SGT (s_{lin} only)	0.71	0.73	0.79	0.80
MoE-SGT (full)	**0.76**	**0.80**	**0.83**	**0.86**
Sparse-CBM (Semenov et al., 2024)	0.72	0.73	0.80	0.81
Label-free CBM (Oikarinen et al., 2023)	0.70	0.71	0.76	0.76
VLG-CBM (Srivastava et al., 2024)	0.71	0.73	0.78	0.80
SCBM (Vandenhirtz et al., 2024)	0.67	0.68	0.71	0.74
CEM (Espinosa Zarlenga et al., 2022)	0.68	0.70	0.71	0.75
CBM (Koh et al., 2020)	0.66	0.67	0.69	0.72

both macro-averaged over the 112 attributes and expressed as percentages. Appendix A.5 gives the formal definitions.

In Table 3, MoE-SGT, CBM, SCBM, and CEM use attribute annotations; Sparse-CBM, VLG-CBM, and Label-free CBM use vision–language targets. MoE-SGT achieves the highest balanced accuracy (87.95%) and macro F1 (81.21%), exceeding CBM by 1.58 and 0.63 percentage points, respectively. The four annotation-supervised methods span 2.79 and 1.97 points in these metrics. Its balanced accuracy exceeds the language-guided baselines by 11.63–15.61 points. On Top-1 classification, MoE-SGT exceeds CBM by 11.23 points and Sparse-CBM by 0.34 points (Table 1). Figure 3 compares predicted scores with binary annotations for one test image.

Table 3: Concept prediction on 112 CUB attributes, macro-averaged (%). Blocks separate annotation and vision–language concept supervision.

Model	Balanced accuracy	Macro F1
MoE-SGT	**87.95**	**81.21**
CBM	86.37	80.58
SCBM	85.29	79.62
CEM	85.16	79.24
Sparse-CBM	76.32	72.04
VLG-CBM	73.68	68.93
Label-free CBM	72.34	65.30

5.5 Test-Time Concept Intervention

We evaluate sequential correction on the CUB-200 test set with the 112 attributes and class-level denoised annotations of Koh et al. (2020). Each selected attribute receives its ground-truth value with all parameters fixed; after each correction we recompute predictions, candidate classes, and attention before selecting the next attribute (Section 4.4). Oracle selection ranks errors using unqueried annotations; graph-informed and uncertainty-only selection use model outputs; random selection samples uniformly. Table 4 reports all methods under oracle selection and the full model under the other policies; $k = 0$ matches Table 1. Under oracle selection, the full model leads at every budget and reaches 89.92% at $k = 10$ (+9.61 points); its margin over the linear head stays positive, narrowing from 5.16 to 2.71 points. Graph-informed selection reaches 86.39% at $k = 10$, recovering 63% of the oracle gain, and exceeds uncertainty-only (85.28%) and random selection (81.83%) at every nonzero budget. Updating candidate retrieval alongside node features adds 2.19 points over node-only correction, and with 20% of supplied values flipped graph-informed correction still reaches 82.43%, above the uncorrected 80.31% (Appendix A.6).

5.6 Ablation Study

5.6.1 Module Contributions and Inference Efficiency

In Table 5, we add cached QA knowledge, graph attention, and sparse experts to an answer–concept GCN; the linear head in Tables 1 and 2 is the no-graph reference. A projected class-text embedding accounts for about two thirds of the QA branch's gain, and the QA graph adds a further 1.52 points on CUB-200 and 0.01 ROC-AUC on MIMIC-CXR; graph attention and sparse experts then add 1.28 and 1.17 points on CUB-200 and 0.03 and 0.01 ROC-AUC on MIMIC-CXR. Latency rises from

Table 4: CUB-200 Top-1 accuracy (%) after k sequential corrections. Upper block: oracle selection for each method; lower block: MoE-SGT policies without access to unqueried annotations. Selected attributes receive ground-truth values. Random results average five draws.

Method / policy	$k = 0$	$k = 1$	$k = 3$	$k = 5$	$k = 10$
CBM (Koh et al., 2020)	69.08	72.30	74.36	76.81	79.43
CEM (Espinosa Zarlenga et al., 2022)	70.37	72.81	74.97	77.23	80.11
SCBM (Vandenhirtz et al., 2024)	69.82	72.52	74.53	77.02	80.09
Sparse-CBM (Semenov et al., 2024)	79.97	81.22	84.18	86.15	88.20
MoE-SGT (s_{lin} only)	75.15	78.05	81.26	85.82	87.21
MoE-SGT (full), oracle	**80.31**	**82.42**	**84.92**	**87.33**	**89.92**
MoE-SGT (full), graph-informed	80.31	81.27	82.87	84.03	86.39
MoE-SGT (full), uncertainty-only	80.31	80.93	81.99	83.10	85.28
MoE-SGT (full), random	80.31	80.52	80.75	81.11	81.83

Table 5: Cumulative module ablation. QA input is none, a projected class-text embedding (text), or a cached answer–question graph (graph). GT replaces the GCN with a graph Transformer; MoE adds sparse experts. Latency is per example at batch size 1: single-image / two-view medical prediction.

Configuration	QA input	GT	MoE	MIMIC-CXR ROC-AUC	CUB-200 Accuracy (%)	Latency (ms)
Base GCN	none			0.69	73.22	81 / 95
+ class-text embedding	text			0.71	76.34	82 / 97
+ QA graph	graph			0.72	77.86	83 / 98
+ Graph Transformer	graph	✓		0.75	79.14	86 / 105
+ MoE (full model)	graph	✓	✓	**0.76**	**80.31**	94 / 113

81 to 94 ms (single-image, $M = 10$) and from 95 to 113 ms (two-view medical, $M = 14$) on one NVIDIA RTX 4090 at batch size 1, excluding offline vocabulary and QA generation.

5.6.2 Priors and Expert Routing

Combining semantic and co-occurrence priors improves on the stronger single prior by 1.08 percentage points on CUB-200 and 0.009 ROC-AUC on MIMIC-CXR (Table 6). MoE exceeds the dense-wide FFN by 0.89 points and 0.008 ROC-AUC with matched total parameter counts ($4.5\times$ the standard FFN). Adding node type and structural context to feature-only routing adds 0.21 points and 0.005 ROC-AUC. Appendix A.7 reports expert-count and training-time concept-clamping ablations.

Table 6: Ablation of the concept–concept prior (a) and the feed-forward block (b): CUB-200 Top-1 accuracy (%) and MIMIC-CXR ROC-AUC, with the full model in bold.

(a) Concept–concept prior

Setting	CUB	MIMIC
Semantic only	79.23	0.751
Co-occurrence only	78.86	0.742
Semantic + co-occurrence	**80.31**	**0.760**

(b) Feed-forward block

Setting	CUB	MIMIC
Dense FFN	79.14	0.750
Dense-wide ($4.5\times$)	79.42	0.752
MoE: features only	80.10	0.755
MoE: structure-aware	**80.31**	**0.760**

6 Conclusion

MoE-SGT combines editable concept evidence with cached class-level QA semantics through answer–concept and answer–question graphs. A graph Transformer with shared experts and structure-aware sparse routing processes these graphs. MoE-SGT achieves the highest Top-1 accuracy among the evaluated CBMs on all four single-label datasets and the highest ROC-AUC and F1 on both medical datasets. On CUB-200, graph reasoning improves accuracy from 75.15% with the linear head to 80.31%. With ten concept corrections, accuracy reaches 86.39% under graph-informed selection and 89.92% under oracle selection, without retraining.

AI USE STATEMENT

Large language models were used for concept extraction and class-level question–answer generation as described in the method. Generative AI tools also assisted language polishing and interpretation of experimental results. All AI-assisted text, analyses, and figures were reviewed by the authors, and all reported numbers were produced by our own code. We take responsibility for the final content of this work.

ETHICS STATEMENT

MIMIC-CXR and CheXpert are de-identified datasets used under their respective data-use agreements; no protected health information was accessed or released, and no data were sent to external services. The medical experiments evaluate concept-based classification on benchmark labels and are not intended for clinical deployment. The remaining datasets are public benchmarks. We are not aware of other ethical concerns.

REFERENCES

Danis Alukaev, Semen Kiselev, Ilya Pershin, Bulat Ibragimov, Vladimir Ivanov, Alexey Kornaev, and Ivan Titov. Cross-modal conceptualization in bottleneck models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5241–5253. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.318. URL https://aclanthology.org/2023.emnlp-main.318/.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, Pietro Lió, and Mateja Jamnik. Concept embedding models: Beyond the accuracy-explainability trade-off. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21400–21413, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Lijie Hu, Chenyang Ren, Zhengyu Hu, Hongbin Lin, Cheng-Long Wang, Hui Xiong, Jingfeng Zhang, and Di Wang. Editable concept bottleneck models. *arXiv preprint arXiv:2405.15476*, 2024.

Lijie Hu, Tianhao Huang, Huanyi Xie, Xilin Gong, Chenyang Ren, Zhengyu Hu, Lu Yu, Ping Ma, and Di Wang. Semi-supervised concept bottleneck models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2110–2119, 2025.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pp. 2704–2710, 2020.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 590–597, 2019.

Alistair EW Johnson, Tom J Pollard, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Yifan Peng, Zhiyong Lu, Roger G Mark, Seth J Berkowitz, and Steven Horng. Mimic-cxr-jpg, a large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*, 2019.

Eunji Kim, Dahuin Jung, Sangha Park, Siwon Kim, and Sungroh Yoon. Probabilistic concept bottleneck models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 16521–16540. PMLR, 2023a. URL https://proceedings.mlr.press/v202/kim23g.html.

Injae Kim, Jongha Kim, Joonmyung Choi, and Hyunwoo J. Kim. Concept bottleneck with visual concept filtering for explainable medical image classification. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2023 Workshops*, pp. 225–233. Springer Nature Switzerland, 2023b. doi: 10.1007/978-3-031-47401-9_22.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

Erxue Min, Runfa Chen, Yatao Bian, Tingyang Xu, Kangfei Zhao, Wenbing Huang, Peilin Zhao, Junzhou Huang, Sophia Ananiadou, and Yu Rong. Transformer for graphs: An overview from architecture perspective. *arXiv preprint arXiv:2202.08455*, 2022.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *International Conference on Learning Representations*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Andrei Semenov, Vladimir Ivanov, Aleksandr Beznosikov, and Alexander Gasnikov. Sparse concept bottleneck models: Gumbel tricks in contrastive learning. *arXiv preprint arXiv:2404.03323*, 2024.

Chenming Shang, Shiji Zhou, Hengyuan Zhang, Xinzhe Ni, Yujiu Yang, and Yuwang Wang. Incremental residual concept bottleneck models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11030–11040, 2024.

Divyansh Srivastava, Ge Yan, and Tsui-Wei Weng. VLG-CBM: Training concept bottleneck models with vision-language guidance. In *Advances in Neural Information Processing Systems*, volume 37, pp. 79057–79094, 2024.

Moritz Vandenhirtz, Sonia Laguna, Ričards Marcinkevičs, and Julia E. Vogt. Stochastic concept bottleneck models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 51787–51810, 2024.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

Zifeng Wang, Zhenbang Wu, Dinesh Agarwal, and Jimeng Sun. MedCLIP: Contrastive learning from unpaired medical images and text. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 3876–3887. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.256. URL https://aclanthology.org/2022.emnlp-main.256/.

Haotian Xu, Tsui-Wei Weng, Lam M. Nguyen, and Tengfei Ma. Graph concept bottleneck models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=a4azUYjRhU.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=nA5AZ8CEyow.

Jianan Zhao, Chaozhuo Li, Qianlong Wen, Yiqi Wang, Yuming Liu, Hao Sun, Xing Xie, and Yanfang Ye. Gophormer: Ego-graph transformer for node classification. *arXiv preprint arXiv:2110.13094*, 2021.

Jian Zhu, Hanli Wang, and Bin He. Multi-modal structure-embedding graph transformer for visual commonsense reasoning. *IEEE Transactions on Multimedia*, 26:1295–1305, 2024. doi: 10.1109/TMM.2023.3279691.

A Implementation and Evaluation Details

A.1 Offline Knowledge Construction

CUB-200 uses a fixed pool of 112 official attributes. For other datasets, an LLM extracts concepts from training reports or generates them from class names. We use the normalized concept embeddings $\mathbf{t}_i$ defined in Section 3. For mined concept pools, we merge concepts c_i, c_k when their cosine distance satisfies $1 - \mathbf{t}_i^\top \mathbf{t}_k < \tau_c$, where τ_c is the redundancy distance threshold. We retain concepts whose mean similarity to class names reaches τ_r and select K by the standard deviation of these similarities across classes. We set $\tau_c = 0.1$, equivalent to cosine similarity greater than 0.9, and $\tau_r = 0.85$. Table A.1 lists the resulting pool sizes.

For each class, offline QA knowledge consists of attribute-oriented questions and an answer containing the class name and associated concepts. DeepSeek-R1-32B generates the concept/QA text without test images or test labels. We cache this text; after training, we also cache the text-only answer–question branch and class-specific concept sets and priors.

For teacher calibration, μ_i and ν_i are the mean and standard deviation of concept i's cosine similarities across training views, with ν_i lower-bounded by a constant $\epsilon_\pi > 0$. These statistics and the temperature $\tau_\pi > 0$ are fixed during bottleneck training; τ_π controls target sharpness. Standardizing each concept across training samples gives it a common scale across candidate classes. The normalized image–text similarities and the teacher priors are computed as in Eq. (1).

A.2 Sparse-Expert Implementation

Each shared or routed expert is a $d \to 2d \to d$ network with dropout 0.1. Training adds independent Gaussian noise: $\tilde{\mathbf{r}}_i = \mathbf{r}_i + \boldsymbol{\epsilon}_i$, with $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 I_{N_e})$, noise scale $\sigma_n > 0$, and identity matrix I_{N_e}. Inference sets $\boldsymbol{\epsilon}_i = \mathbf{0}$. We initialize the router with small-variance weights and gather/scatter tokens per expert without expert parallelism, capacity limits, or dropped tokens.

Excluding biases and router overhead, each expert has $4d^2$ weights: active and total expert weights are $12d^2$ and $36d^2$, or $1.5\times$ and $4.5\times$ those of one dense $d \to 4d \to d$ FFN.

A.3 Loss Components and Routing Balance

The objective in Eq. (6) combines classification, concept supervision, and routing balance. For Switch-style routing balance (Fedus et al., 2022), let $D = \max(D_c, D_r)$ count layer-specific expert pools and T_ℓ count tokens across all samples and graphs in the batch at layer ℓ:

$$f_e^{(\ell)} = \frac{1}{\kappa T_\ell} \sum_{t=1}^{T_\ell} \mathbb{I}[e \in \mathcal{T}_t^{(\ell)}], \qquad P_e^{(\ell)} = \frac{1}{T_\ell} \sum_{t=1}^{T_\ell} [\mathrm{softmax}(\mathbf{r}_t^{(\ell)})]_e,$$
$$\mathcal{L}_{\mathrm{balance}} = \frac{1}{D} \sum_{\ell=0}^{D-1} N_e \sum_{e=1}^{N_e} f_e^{(\ell)} P_e^{(\ell)}. \tag{A.1}$$

Here, t indexes token occurrences, $\mathcal{T}_t^{(\ell)}$ is the selected expert set, and $\mathbf{r}_t^{(\ell)}$ contains noise-free logits. The indicator $\mathbb{I}$ is one when its condition holds. Dispatch fraction $f_e^{(\ell)}$ is normalized by κT_ℓ; $P_e^{(\ell)}$ averages the full softmax before masking. The shared expert is excluded, and gradients flow through probabilities but not dispatch counts. We pool these statistics across graphs for each shared expert pool. The classification and bottleneck terms are

$$\mathcal{L}_{\mathrm{cls}} = \mathcal{L}_{\mathrm{graph}} + \lambda_l \mathcal{L}_{\mathrm{linear}}, \qquad \mathcal{L}_{\mathrm{CBL}} = \mathcal{L}_{\mathrm{align}} + \lambda_s \mathcal{L}_{\mathrm{sparse}},$$
$$\mathcal{L}_{\mathrm{align}} = \frac{1}{K} \sum_{i=1}^{K} (p_i(x) - \bar{\pi}_i(x))^2, \qquad \mathcal{L}_{\mathrm{sparse}} = \alpha \|W_l\|_1 + \frac{1-\alpha}{2} \|W_l\|_F^2. \tag{A.2}$$

The terms $\mathcal{L}_{\mathrm{align}}, \mathcal{L}_{\mathrm{sparse}}$ align concept scores with their targets and regularize W_l; nonnegative λ_l, λ_s weight the linear-head and sparse objectives. The coefficient $\alpha \in [0, 1]$ mixes the entry-wise L_1 norm $\|\cdot\|_1$ and squared Frobenius norm $\|\cdot\|_F^2$. Task/concept losses are averaged over examples. The teacher targets $\bar{\pi}_i$ and discrete class/concept selection indices are held fixed in differentiation; W_l learns through the linear loss and sparse regularization.

A.4 Dataset Configurations

Table A.1 gives the dataset-specific configuration: C is the number of classes or pathologies, K the concept-pool size, M the number of candidate classes or independently scored pathologies, and m the number of active concepts per candidate. Across datasets, we use a frozen ViT-B/16 backbone, $D_c = 2$ contextualization layers, $D_r = 1$ answer-reasoning layer, width $d = 256$, and four attention heads with $d_h = 64$. Each MoE block has one shared expert and $N_e = 8$ routed experts with top-2 gating.

Table A.1: Dataset-specific concept and graph configurations. All classes are scored for CIFAR-10 and all pathologies for the medical datasets.

Dataset	C	K	M	m
CIFAR-10	10	120	10	16
CIFAR-100	100	824	10	32
CUB-200	200	112	10	48
ImageNet	1000	4096	20	48
MIMIC-CXR / CheXpert	14	1024	14	64

Candidate retrieval. Table A.2 reports the fraction of test examples whose true class appears among the top-$\min(M, C)$ scores of the linear head s_{lin}. Recall bounds the Top-1 accuracy achievable by reranking these candidates. The selected budgets retain the true class for 96.55% of CUB-200, 96.67% of ImageNet, 100.00% of CIFAR-10, and 96.60% of CIFAR-100 test examples.

Table A.2: Candidate recall (%) of the linear head. Bold entries mark the budgets used by the full model.

Dataset	$M = 1$	$M = 5$	$M = 10$	$M = 20$
CUB-200	75.15	93.41	**96.55**	98.31
ImageNet	69.82	89.33	93.64	**96.67**
CIFAR-10	88.93	99.48	**100.00**	100.00
CIFAR-100	72.69	92.84	**96.60**	98.71

A.5 Concept Prediction Metrics

All models predict the same 112 official CUB attributes. We predict attribute i as present when $p_i(x) \geq 0.5$, where $p_i(x)$ is its predicted presence score for test image x. For each attribute, TP_i, TN_i, FP_i, and FN_i count true positives, true negatives, false positives, and false negatives over test images, with attribute presence as the positive class. Balanced accuracy averages positive- and negative-class recall; F1 is computed for the positive class. We macro-average both metrics over $K = 112$ attributes and report percentages:

$$\begin{aligned} \mathrm{BAcc} &= \frac{1}{2K} \sum_{i=1}^{K} \left(\frac{\mathrm{TP}_i}{\mathrm{TP}_i + \mathrm{FN}_i} + \frac{\mathrm{TN}_i}{\mathrm{TN}_i + \mathrm{FP}_i} \right), \\ \mathrm{F1}_{\mathrm{macro}} &= \frac{1}{K} \sum_{i=1}^{K} \frac{2\mathrm{TP}_i}{2\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i}. \end{aligned} \tag{A.3}$$

A.6 Additional Concept Intervention Results

Table A.3 supplements the CUB-200 results in Section 5.5. All corrected runs use ten sequential attribute corrections. Under oracle selection, node-only correction updates graph features while retaining the initial candidate set; retrieval-only correction recomputes candidates from corrected scores but uses the original node features; the full update recomputes both. Their accuracies are 87.73%, 82.45%, and 89.92%, respectively. Updating candidates as well as node features adds 2.19 percentage points over node-only correction.

To measure coverage of the active concept set, we fix each image's predicted class j before intervention. At $k = 10$, we count how many of its ten oracle-corrected attribute indices belong to $S_m^{(j)}$ ($m = 48$), divide by ten, and average over test images. Mean coverage is 76.4%, measured against the fixed pre-intervention class.

For imperfect feedback, each supplied binary value is flipped with probability $\rho \in \{0, 0.1, 0.2\}$ under graph-informed selection, and results average five draws. At $\rho = 0.1$ and 0.2, accuracy is 84.41% and 82.43%, exceeding the uncorrected 80.31% by 4.10 and 2.12 points.

Table A.3: CUB-200 Top-1 accuracy (%) at $k = 10$: correction pathways under oracle selection and graph-informed feedback with flip probability ρ.

Setting	Top-1 (%)
No correction	80.31
Oracle, node-only	87.73
Oracle, retrieval-only	82.45
Oracle, full update	**89.92**
Graph-informed, $\rho = 0$	86.39
Graph-informed, $\rho = 0.1$	84.41
Graph-informed, $\rho = 0.2$	82.43

A.7 Auxiliary Ablations

Table A.4 varies the routed-expert count and training-time concept clamping separately. With all other settings fixed, increasing the number of routed experts from two to eight raises MIMIC-CXR ROC-AUC from 0.74 to 0.76; sixteen experts also yield 0.76. We use $N_e = 8$. Disabling concept clamping during training lowers ROC-AUC from 0.76 to 0.75 on MIMIC-CXR and from 0.83 to 0.82 on CheXpert. We evaluate test-time corrections separately in Section 5.5.

Table A.4: Auxiliary ablations, reported as ROC-AUC. The two panels vary one setting at a time; bold rows mark the default settings.

(a) Routed-expert count

Experts	MIMIC-CXR
2	0.74
8	**0.76**
16	0.76

(b) Training-time concept clamping

Clamping	MIMIC-CXR	CheXpert
Enabled	**0.76**	**0.83**
Disabled	0.75	0.82