Learn Before Represent: Unifying Knowledge Acquisition and Contrastive Alignment for Domain-Specific Retrieval

Anonymous ACL submission

Abstract

Contrastive learning (CL) adapts LLMs into effective general-purpose embedding models, yet transfers poorly to specialized domains. The core issue is that CL can align existing representations but cannot create new ones for domain concepts absent from the pretraining data. A natural remedy, first injecting domain knowledge via generative learning (GL) then aligning via CL, actually degrades retrieval below CL-only baselines due to destructive interference between the two objectives. We propose **Learn Before Represent (LBR)**, a two-stage framework that unifies knowledge acquisition and representation learning in a single causal LLM. In Stage 1, an information-bottleneck attention mask forces the model to compress input semantics into a few bottleneck tokens before generating domain-specific targets. In Stage 2, the same bottleneck states serve as sequence embeddings for contrastive alignment. Because generation absorbs domain knowledge while CL refines only the bottleneck, gradient conflict is eliminated. Experiments on medical, chemistry, and code retrieval show that LBR consistently outperforms CL-only and naive GL+CL baselines by wide margins, with gains validated on additional external benchmarks. Our code is publicly available at `https://anonymous.4open.science/r/LLaMA-Factory-4-SumAndRepeat-7EE5`.

1 Introduction

LLM-based embedding methods (BehnamGhader et al., 2024; Zhang et al., 2025b) leverage the rich world knowledge of pretrained language models and employ contrastive learning (CL) to achieve strong performance on general benchmarks such as MTEB (Muennighoff et al., 2023). However, this success does not consistently transfer to specialized domains—medicine, chemistry, and code—where models must handle domain-specific terminology and long-tail entities rarely seen during pretraining (Kasmaee et al., 2024; Tang and Yang, 2025).

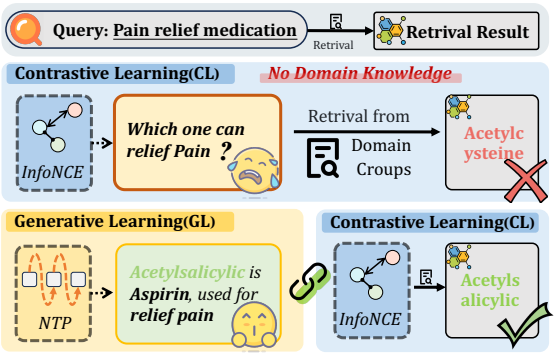

Figure 1: **Contrastive learning alone cannot align what it does not know.** *Top:* A CL-only model retrieves "Acetylcysteine" instead of "Acetylsalicylic acid" (Aspirin) for the query "pain relief medication," because it lacks the pharmaceutical knowledge that these are distinct entities with different functions. *Bottom:* When generative learning (GL) first internalizes domain knowledge, the model correctly identifies Acetylsalicylic acid as Aspirin—a common painkiller—and retrieves it accurately.

Consider a concrete failure case (Figure 1, top): when retrieving pain relievers, a model lacking pharmaceutical knowledge cannot recognize that "Acetylsalicylic acid" is "Aspirin," because the entity equivalence was never internalized during pretraining.

This example highlights a fundamental limitation of the prevalent **"LLM+CL"** paradigm: contrastive objectives can align and separate existing representations (Wang and Isola, 2020), but cannot reliably induce representations for domain concepts absent from the model's parametric knowledge (Tang and Yang, 2024). Existing mitigations—hard negative mining (Lei et al., 2023) or domain-specific data construction (Sun et al., 2025b; Kasmaee et al., 2025)—improve coverage but do not close this gap. Without internalized knowledge of entity equivalences (e.g., chemical nomenclature), contrastive learning falls back on surface-form sim-

ilarity rather than true semantic equivalence. Therefore, domain knowledge must be acquired *before* representation alignment—a model cannot align what it does not understand.

This reasoning naturally suggests an **"LLM+GL+CL"** paradigm: inject domain knowledge via generative learning (GL) and *then* align representations via contrastive learning (CL). To test whether this intuitive combination actually works, we conduct a controlled study on the medical domain, measuring both generative quality and retrieval accuracy across four model variants (Tables 3 and 2). The results reveal a paradox: GL improves domain-accurate generation, confirming knowledge internalization, but yields only modest retrieval gains over the base model. CL substantially improves retrieval yet completely destroys generative capability. Critically, naive GL→CL *degrades* retrieval below CL-only baselines, rather than combining the strengths of both.

The failure stems from two structural conflicts. First, an **architectural mismatch**: leading embedding methods (LLM2Vec (BehnamGhader et al., 2024), NV-Embed (Lee et al., 2024a)) convert decoder-only LLMs into bidirectional encoders for CL, yet autoregressive knowledge acquisition inherently requires causal attention; switching between these two modes across stages either disrupts the generative capability learned in GL or sacrifices the bidirectional context that CL relies on (Lin et al., 2025; Cui et al., 2025). Second, an **objective conflict**: GL optimizes token-level cross-entropy while CL optimizes sample-level InfoNCE (Zhou et al., 2025); without explicit decoupling, the two objectives interfere destructively.

These two conflicts suggest that a successful "LLM+GL+CL" paradigm requires a mechanism that (i) maintains a single, consistent architecture across both learning phases, and (ii) structurally isolates the generative and contrastive gradients. To address these challenges, we propose **Learn Before Represent (LBR)**, a two-stage framework that unifies knowledge acquisition and representation learning within a consistent causal architecture. The key mechanism is an **Information Bottleneck (IB)** (TISHBY, 2000; Hjelm et al., 2018) that structurally decouples the generative and contrastive objectives: in **Stage 1**, bottleneck tokens compress all input semantics before generation, simultaneously acquiring domain knowledge and learning compact representations; in **Stage 2**, the hidden states of these bottleneck tokens serve as sequence embeddings and are refined via contrastive learning, reusing the same causal architecture for seamless knowledge transfer. Unlike standard continued pretraining followed by contrastive fine-tuning, the bottleneck ensures domain semantics are expressed in compact token states directly reusable as embeddings. Unlike joint-training approaches (e.g., GritLM (Muennighoff et al., 2025)), LBR *sequentially* decouples the two objectives, avoiding gradient conflict.

Our main contributions are:

- We identify that vertical-domain retrieval requires *internalizing domain-specific knowledge*—contrastive learning can only re-weight existing representations but cannot create new ones for unseen domain concepts. Naively combining generative and contrastive learning fails due to architectural mismatch and objective conflict.
- We propose **Learn Before Represent (LBR)**, a two-stage framework that resolves both conflicts through an information-bottleneck mechanism implemented purely via attention masking, compressing domain knowledge into bottleneck tokens (Stage 1) then reusing them as embeddings for contrastive alignment (Stage 2), all within a single causal architecture.
- LBR (Qwen2.5-1.5B) outperforms the strongest baseline by +7.8 Recall@10 averaged across medicine, chemistry, and code retrieval, while being the only contrastive method that retains full generative capability.

2 Related Work

2.1 LLM-based Embedding Models

Recent work adapts decoder-only LLMs for text embedding. E5-Mistral (Wang et al., 2024), Gecko (Lee et al., 2024b), LLM2Vec (BehnamGhader et al., 2024), and NV-Embed (Lee et al., 2024a) achieve strong general-domain performance on MTEB (Muennighoff et al., 2023) through synthetic data, instruction tuning, or bidirectional attention, with recent scaling efforts (Zhao et al., 2025; Zhang et al., 2025b) pushing the frontier further. However, bidirectional methods sacrifice native generation capability, and contrastive fine-tuning alone degrades in vertical domains with specialized terminology (Tang and Yang, 2024; Kasmaee et al., 2024; Tang and Yang, 2025), as it primarily re-weights existing features rather than acquiring new knowledge.

Domain-adaptive continued pretraining (DAPT) (Gururangan et al., 2020; Izacard et al., 2022) before contrastive fine-tuning is a natural remedy, but introduces architectural inconsistency when causal generation is followed by bidirectional embedding, and competing gradients between token-level and instance-level objectives (Zhou et al., 2025). These challenges motivate architecturally unified approaches like ours.

2.2 Unifying Generation and Representation

Joint Training. GritLM (Muennighoff et al., 2025) jointly optimizes generative and embedding objectives via instruction-based routing, but simultaneous optimization can suffer from objective interference (Zhou et al., 2025; Tsukagoshi and Sasano, 2025). Our framework instead *sequentially* decouples the two stages through an explicit bottleneck.

Causal Embedding. Causal2Vec (Lin et al., 2025), Echo Embeddings (Springer et al., 2024), and Nomic Embed (Nussbaum et al., 2024) maintain causal attention for embedding, avoiding architectural inconsistency with generative pretraining. Our method similarly preserves causal attention throughout both stages, ensuring seamless knowledge transfer.

Reasoning-Embedding. Think&Embed (Cui et al., 2025), GIRCSE (Tsai et al., 2025), CCoT (Cheng and Van Durme, 2024), GRACE (Sun et al., 2025a), and LREM (Liu et al., 2025) enhance embedding via reasoning depth or RL-based contrastive learning. These address the reasoning gap but not the domain knowledge gap central to vertical embedding—a complementary challenge we tackle through generative knowledge acquisition.

2.3 Compression and Information Bottleneck for Representations

The Information Bottleneck (IB) principle (TISHBY, 2000; Hjelm et al., 2018) provides theoretical grounding for compact yet informative representations. In NLP, RetroMAE (Liu and Shao, 2022) and UniMAE (Qiao et al., 2025) apply masked auto-encoding for text embeddings but rely on asymmetric architectures (Zhang et al., 2025a; Deng et al., 2025). Perceiver (Jaegle et al., 2021), Gist Tokens (Mu et al., 2023), and Memory Tokens (Sastre and Rosá, 2025) route information through latent tokens for compression or efficiency.

Our bottleneck mechanism differs from these in three key aspects: (1) it targets *domain knowledge acquisition* rather than inference efficiency, forcing the model to internalize domain semantics during generative learning (Table 2); (2) it operates *intrinsically* via a causal attention mask without external decoders or architectural modifications (§3.1); and (3) it employs two-stage *sequential* decoupling rather than joint optimization, resolving objective interference (Table 2).

3 Method

We propose **L**earn **B**efore **R**epresent (**LBR**) , a two-stage framework for adapting decoder-only LLMs into domain-specific embedding models (Figure 2). Stage 1 performs IB-constrained generative learning for domain knowledge acquisition through bottlenecked compression; Stage 2 applies contrastive learning for representation alignment.

3.1 Information-Theoretic Motivation

We draw motivation from the Information Bottleneck (IB) principle (TISHBY, 2000), which posits that an optimal representation Z of input X for predicting target Y should solve:

$$\min_{p(Z|X)} \underbrace{I(X;Z)}_{\text{Compression}} - \beta \underbrace{I(Z;Y)}_{\text{Prediction}} \tag{1}$$

where $I(\cdot;\cdot)$ denotes mutual information and β controls the compression–prediction trade-off. Directly optimizing Eq. (1) requires intractable mutual information estimation. Instead, we realize the IB *desiderata* through architectural constraints (attention masking for compression) and tractable proxy objectives (autoregressive generation for prediction), as detailed in §3.2.

3.2 Stage 1: IB-Constrained Generative Learning

Structural Factorization via Attention Mask. Let $T = [X; Z; Y]$ denote the concatenated input–bottleneck–target sequence. We define the attention mask $\mathbf{M}$ such that for any target position t corresponding to $y_t \in Y$:

$$\mathbf{M}_{t,j} = \begin{cases} 1, & \text{if } j \in Z \cup Y_{\leq t} \\ 0, & \text{if } j \in X \quad \textit{(cut-off)} \end{cases} \tag{2}$$

This mask ensures that the model's predictive distribution factors through Z:

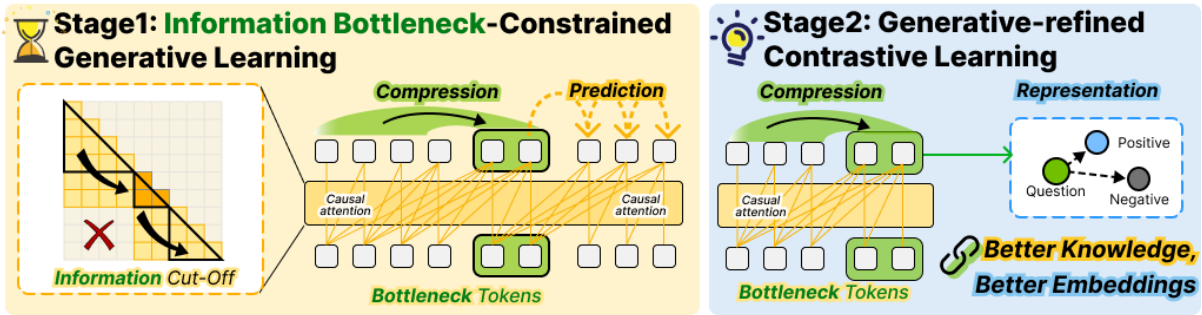

Figure 2: **Overview of the LBR framework.** In **Stage 1**, the model performs generative learning under an Information Bottleneck constraint, *compressing* input semantics into bottleneck tokens and *predicting* the target solely from these tokens. In **Stage 2**, the bottleneck tokens serve as embeddings for *representing* semantics through contrastive learning. Both stages share unified causal attention to maximize knowledge transfer.

Proposition 1 (Model-Induced Conditional Independence)**.** *Under the LBR attention mask (Eq. 2), the model's conditional distribution satisfies:*

$$q_\theta(Y \mid X, Z) = q_\theta(Y \mid Z) \quad \textit{for all } X \tag{3}$$

That is, the input X influences predictions only through the bottleneck Z in the model's forward computation.

Proof sketch. Since $\mathbf{M}_{t,j} = 0$ for all $j \in X$ at every target position t, the hidden states of Y depend only on $\{Z, Y_{\leq t}\}$ at every layer. The full proof is in Appendix C.1.

This establishes that the LBR architecture enforces the Markov chain $X \to Z \to Y$ *within the model class*—the target can only access input information that has been compressed into Z.

Capacity Constraint via Compression Ratio. The bottleneck Z consists of k token positions, each producing a hidden state $\mathbf{h}_{z_i} \in \mathbb{R}^d$. We parametrize bottleneck strength by the compression ratio $R = |X|/|Z|$. If we consider each bottleneck position as transmitting a discrete symbol from an effective codebook of size $|\mathcal{V}|$ (e.g., via the LM head), the channel capacity is at most:

$$C(Z) = k \cdot \log |\mathcal{V}| = \frac{|X|}{R} \cdot \log |\mathcal{V}| \tag{4}$$

Increasing R reduces $C(Z)$, forcing stronger compression of X into Z and thus instantiating the $\min I(X; Z)$ term of the IB objective through *architectural constraint* rather than explicit optimization.

The feasibility of such extreme compression is supported by recent findings on memory tokens (Sastre and Rosá, 2025), which demonstrate that LLMs can distill hundreds of tokens into a single embedding while maintaining full reconstructability.

Maximizing $I(Z; Y)$ via Generative Loss. With the information flow constrained to pass through Z (Proposition 1), we maximize the predictive information $I(Z; Y)$ through the standard next-token prediction loss:

$$\mathcal{L}_{\text{gen}} = -\mathbb{E}_{p(X,Y)} \left[\sum_{t=1}^{|Y|} \log q_\theta(y_t \mid Z, Y_{<t}) \right] \tag{5}$$

As shown in Proposition 2 (Appendix C.2), minimizing $\mathcal{L}_{\text{gen}}$ maximizes a variational lower bound on $I(Z; Y)$, i.e., $I(Z; Y) \geq H(Y) - \mathcal{L}_{\text{gen}}$. This connects our autoregressive training objective directly to the IB prediction term (Alemi et al., 2017; TISHBY, 2000).

Depending on data availability, this phase has two variants, both designed to instill domain knowledge. When labeled query-answer pairs are available, we adopt SFT-style GL with $X = Q$ and $Y = A$, guiding the model to compress query semantics into Z for answer generation. When only unlabeled domain corpora are available, we employ PT-style generative learning via self-supervised objectives: either passage reconstruction ($X = Y = D$) or prefix-suffix prediction ($X = D_{:k}, Y = D_{k:}$). This variant enables unsupervised knowledge acquisition at the cost of increased computation (see Section B.3 for detailed analysis).

Information-Theoretic Interpretation. Combining Propositions 1 and 2 with the capacity constraint (Eq. 4), we can interpret the LBR Stage 1 as an *architectural instantiation* of the IB principle:

- **Compression** ($\min I(X;Z)$): enforced structurally via the attention mask $\mathbf{M}$ and bounded by the compression ratio R, without requiring an explicit MI minimizer.
- **Prediction** ($\max I(Z;Y)$): optimized via the generative loss $\mathcal{L}_{\text{gen}}$, which provably maximizes a variational lower bound on $I(Z;Y)$ (Proposition 2).

The compression ratio R plays an analogous role to the Lagrange multiplier β in the IB objective: increasing R tightens the capacity constraint on Z, shifting the operating point toward stronger compression—though we note this is a qualitative analogy rather than a formal equivalence, since R controls discrete architectural capacity rather than a continuous regularization coefficient. We empirically verify this trade-off in Section 5.1, where performance follows an inverted-U shape—mild compression ($R \leq 20$) provides insufficient bottleneck pressure, while extreme compression ($R = 1000$) causes information loss, with the optimum at $R = 500$.

To be explicit: our contribution is *not* a mutual-information estimator but rather a *practical bottleneck-token mechanism with constrained attention* that achieves the compression–prediction trade-off prescribed by the IB principle. This is analogous to how dropout regularizes without formally optimizing a Bayesian posterior.

3.3 Stage 2: Generative-Refined Contrastive Learning

Building on the semantic compression capability established in Stage 1, Stage 2 aligns representations through **Generative-Refined Contrastive Learning (GR-CL)**. We preserve the native causal attention mechanism to ensure architectural consistency with Stage 1.

Formally, given the input structure $[X; Z]$ where X denotes the domain sequence and Z the bottleneck tokens, we extract the hidden state of the final token in Z as the sequence representation $\mathbf{v}$. We align query-passage pairs via the InfoNCE loss:

$$\mathcal{L}_{\text{contrast}} = -\log \frac{e^{s(q,p^+)/\tau}}{e^{s(q,p^+)/\tau} + \sum_{p^-} e^{s(q,p^-)/\tau}} \tag{6}$$

where $s(q,p) = \text{sim}(\mathbf{v}_q, \mathbf{v}_p)$, τ is the temperature, and the sum runs over in-batch negatives.

Unlike methods that rely on auxiliary decoders or structural changes (Qiao et al., 2025; Deng et al., 2025; Su et al., 2025), our approach preserves the LLM's native architecture and generative capacity for domain knowledge acquisition. Preserving the causal attention mechanism yields two key benefits: (1) **Consistency**: seamless transition from Stage 1, fully leveraging the acquired domain knowledge and compression abilities; (2) **Extensibility**: retaining the LLM's instruction-following capabilities and compatibility with future reasoning-enhanced embeddings (Tsai et al., 2025).

From an IB perspective, the two stages naturally decompose the optimization: Stage 1 maximizes $I(Z;Y)$ under capacity constraint to produce informative yet compact representations, while Stage 2 discards task-irrelevant information through contrastive invariance. Sequential optimization avoids the gradient conflict between token-level retention and coarse semantic invariance, as confirmed in Table 2. A detailed analysis is provided in Appendix D.1.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate on three specialized domains: medicine, chemistry, and code. For each domain, we allocate 150k samples for Stage 1 (GL/IB-GL), 50k for Stage 2 (CL), and 20k for held-out evaluation, following a deterministic partition protocol with verified zero overlap (Appendix A).

Evaluation. We use the MTEB (Muennighoff et al., 2023) retrieval pipeline with cosine similarity, reporting Recall@10 and NDCG@10. Evaluation sets are strictly isolated from training data (Appendix A.4). We further validate on external benchmarks in Section 4.2.

Baselines. We compare against: (1) *general-purpose embedders* evaluated off-the-shelf—BGE (Xiao et al., 2024), GTE (Li et al., 2023), and LLM2Vec (BehnamGhader et al., 2024); (2) *domain-adapted LLMs*—ChemLLM (Yao et al., 2024) and HuaTuo (Wang et al., 2023); and (3) *SFT+CL*, which uses the same domain data and budget as LBR but without the bottleneck constraint. Full baseline protocols are in Appendix A.6.

Implementation. We use Qwen2(.5)-1.5B and Llama3.2-1B as backbone LLMs. Stage 1 (IB-GL) trains for 3,000 steps with AdamW (lr=2e-5, batch size 128); Stage 2 (CL) trains for 1,000 steps (lr=1e-5, batch size 256, temperature τ=0.02). The default

Models	Backbone	Attn.	Chemistry		Medical		Code		Avg. (Rank)
			R@10	N@10	R@10	N@10	R@10	N@10	
Pre-trained LLMs *(LLM+GL)*									
Qwen2	Qwen2-1.5B	Causal	0.100	0.078	0.410	0.335	0.151	0.108	20.0 (10)
Qwen2.5	Qwen2.5-1.5B	Causal	0.146	0.115	0.353	0.286	0.116	0.081	18.3 (11)
ChemLLM	InternLM-2B	Causal	0.237	0.168	0.034	0.020	0.116	0.082	11.0 (12)
HuaTuo	LLaMA-7B	Causal	0.201	0.188	0.346	0.283	0.13	0.101	31.2 (9)
Contrastive Learning-based Methods *(LLM+CL)*									
BGE	XLMR-Large	Bi.	0.689	0.557	0.947	0.871	0.882	0.754	78.3 (5)
GTE	Qwen2-1.5B	Bi.	0.766	0.677	0.958	0.867	0.360	0.286	65.2 (6)
LLM2Vec	SLLaMA-1.3B	Bi.	0.712	0.615	0.961	0.890	0.850	0.729	79.3 (4)
Contrastive Learning on Domain LLMs *(LLM+GL+CL)*									
SFT+CL	Qwen2-1.5B	Causal	0.449	0.405	0.732	0.625	0.888	0.764	64.4 (7)
SFT+CL	Qwen2.5-1.5B	Causal	0.436	0.391	0.664	0.558	0.855	0.732	60.6 (8)
Ours Method *(LLM+IB-GL+CL)*									
LBR	Llama3.2-1B	Causal	0.722	0.619	0.976	0.897	0.979	0.879	84.5 (3)
LBR	Qwen2-1.5B	Causal	**0.802**	**0.727**	0.978	0.899	0.978	0.882	87.8 (2)
LBR	Qwen2.5-1.5B	Causal	0.797	0.723	**0.979**	**0.906**	**0.980**	**0.887**	**87.9 (1)**

Table 1: Performance comparison on domain-specific retrieval tasks across three vertical domains (Chemistry, Medical, Code). **R@10** and **N@10** denote Recall@10 and NDCG@10 (reported as proportions in [0,1]). **Avg.** is the mean of all six metrics, scaled to [0,100] for readability. **Attn.** denotes the attention architecture (*Causal* = unidirectional, *Bi.* = bidirectional). Highlighted tags categorize training paradigms: **LLM+GL**: generative-only baselines; **LLM+CL**: contrastive learning methods; **LLM+GL+CL**: naive sequential combination; **LLM+IB-GL+CL**: our proposed method. **Bold** = best per column.

compression ratio is R=500. Full hyperparameters are in Appendix B.2.

4.2 Main Results

Vertical Domain Benchmarks. Table 1 compares performance across Chemistry, Medical, and Code domains with Recall@10 as the primary metric. LBR (Qwen2.5-1.5B) achieves the highest Recall@10 in all three domains, outperforming the strongest contrastive baseline LLM2Vec by an average of +7.8 points. The gains are especially pronounced in knowledge-intensive settings: in Code, LBR recalls 98.0% of relevant documents versus 85.0% for LLM2Vec, and in Chemistry it improves by +8.5 points. Even our smaller Llama3.2-1B variant surpasses all baselines on Medical and Code, confirming that LBR's bottleneck-guided learning enables compact models to match or exceed larger conventionally trained ones. Notably, LBR achieves these gains while retaining a causal attention mask, outperforming bidirectional models such as BGE and GTE that rely on full self-attention, which demonstrates that bottleneck-guided knowledge internalization, rather than architectural modification, is the key enabler. In contrast, the naive sequential baseline (SFT+CL) suffers severe recall degradation, falling well below even standard contrastive baselines on Chemistry, confirming that naively combining the two objectives causes catastrophic forgetting.

Generalization to External Benchmarks. We further evaluate on three public retrieval tasks—ChemTEB-Retrieval (Kasmaee et al., 2024), NFCorpus (Boteva et al., 2016), and CodeSearchNet (Husain et al., 2019)—to verify generalization. LBR again achieves the best average performance, most strikingly on ChemTEB-Retrieval, where it outperforms LLM2Vec and SFT+CL by over 15 and 12 Recall@10 points respectively (Appendix A.5).

4.3 Analysis of LBR

Q1: Does domain knowledge improve retrieval? Yes. Table 3 compares four model variants on both generation and retrieval. **GL** achieves the highest generation scores (14.32 B-4), confirming successful knowledge internalization. It also improves retrieval over the base model (54.91 vs. 41.03 R@10), yet significantly lags behind contrastive methods, indicating that next-token prediction alone is insufficient for robust semantic representations.

Stages	Medical	Code
LLM	33.52	10.78
LLM+GL	45.98	40.71
LLM+IB-GL	80.60	73.31
LLM+CL	64.35	76.91
LLM+GL+CL	62.45	76.36
LLM+IB-GL+CL	89.85	88.22

Table 2: **Ablation on training stages** (NDCG@10, Medical and Code domains, Qwen2-1.5B backbone). Each row adds one component. "LLM" = pretrained base model; "GL" = standard generative learning; "IB-GL" = our IB-constrained GL; "CL" = contrastive learning. Bar lengths are proportional to NDCG@10 scores.

Method	Retrieval		Generation	
	R@10	N@10	B-4	R-L
Base	41.03	26.68	4.42	23.71
GL	54.91	45.98	**14.32**	**35.79**
CL	75.59	64.35	–	–
IB-GL	**90.33**	**80.60**	13.98	35.58

Table 3: **Representation and generation capability on the medical domain.** We report Recall@10 (R@10) and NDCG@10 (N@10) for retrieval tasks, alongside BLEU-4 (B-4) and ROUGE-L (R-L) scores for generation tasks. The **best** performance is highlighted in bold, and the second best is underlined.

Compression Ratio	NDCG@10	Recall@10
$R = 10$	8.17	10.65
$R = 20$	10.84	13.26
$R = 50$	17.14	20.66
$R = 100$	18.75	21.98
$R = 500$	**27.31**	**30.09**
$R = 1000$	20.97	23.57

Table 4: **Impact of compression ratio on retrieval performance.** We evaluate six compression ratios by stratifying training samples by length and assigning different numbers of bottleneck tokens accordingly.

Q2: Does naive GL+CL work? A natural attempt is to apply GL then CL sequentially. However, Table 2 reveals that "GL+CL" *underperforms* CL alone in the medical domain. This validates the Introduction's hypothesis: when generative and contrastive objectives are applied sequentially without explicit decoupling, the resulting representations degrade rather than improve. The root cause is twofold—architectural mismatch (causal vs. bidirectional) and objective conflict (token-level cross-entropy vs. sample-level InfoNCE).

Q3: How does the IB bottleneck resolve both issues? *First*, on its own, IB-GL achieves state-of-the-art retrieval (**90.33** R@10), outperforming CL, while preserving generative capability. This validates that the bottleneck simultaneously enables knowledge acquisition *and* semantic compression (Table 3). *Second*, when further combined with CL (Table 2), IB-GL+CL achieves the best overall performance across all three domains. Unlike naive GL+CL, the bottleneck structurally decouples the two objectives: domain semantics are compressed into bottleneck tokens during Stage 1, and CL refines only these tokens in Stage 2—eliminating gradient conflict. The improvement is dramatic: R@10 jumps by **+36.6%** (0.436→0.802) in Chemistry compared to naive SFT+CL.

5 Ablation Studies

Having established *why* each LBR component is necessary (Section 4.3), we now ablate *how* each design choice should be configured. We examine three dimensions: the compression ratio that controls the IB operating point (§5.1), the attention pattern during contrastive alignment (§5.2), and the data allocation between generative and contrastive objectives (§5.3).

5.1 Bottleneck Compression Ratio

Section 3.2 predicts that increasing the compression ratio R shifts the IB operating point toward stronger compression, yielding an inverted-U performance curve. We verify this by evaluating $R \in \{10, 20, 50, 100, 500, 1000\}$ on Qwen2-1.5B with Stage 1 only (1,000 steps).

Table 4 confirms the predicted inverted-U relationship. At low R ($\leq$100), the bottleneck capacity is too large to force meaningful abstraction; at the optimum $R = 500$, compression eliminates task-irrelevant redundancy while preserving sufficient semantic content for target reconstruction; beyond

Method	Base Attn.	Attention Mechanism in CL	
		Causal (Ours)	Bidirectional
GL	Causal	54.91	55.04
IB-GL (Ours)	Causal	**75.59**	64.35

Table 5: **Impact of Attention Mechanisms during Contrastive Learning** (Medical domain, NDCG@10 scaled to [0,100]). We compare Causal vs. Bidirectional attention masks during Stage 2 CL. Note: "IB-GL" here evaluates Stage 1 output directly; the full pipeline (IB-GL+CL) in Table 1 achieves higher scores after contrastive alignment.

this ($R = 1000$), information loss dominates and performance degrades. Performance remains stable across $R \in [200, 500]$, demonstrating robustness. We use $R = 500$ as the default.

5.2 Causal vs. Bidirectional Attention

We compare two Stage 2 CL variants—**Bidirectional** (removing the causal mask) and **Causal** (preserving the native mask)—to test whether Stage 2 should maintain the causal structure of IB-GL.

Table 5 shows that IB-GL+Causal CL (75.59) significantly outperforms IB-GL+Bidirectional CL (64.35), while for standard GL the two yield comparable results (54.91 vs. 55.04). This confirms that the causal advantage is specifically synergistic with IB-GL: switching to bidirectional attention creates a distribution shift that disrupts the autoregressive compression pathway learned in Stage 1. Moreover, causal attention forces the model to leverage internalized knowledge rather than shallow bidirectional pattern matching, further benefiting the bottleneck representations.

5.3 Data Allocation Efficiency

The above ablations fix data volume and vary architectural choices. We now ask a complementary question: do performance gains stem from our method or from increased data? We fix a total budget of 100k samples and vary the GL ratio r_{learn} from 0% (pure CL) to 100% (pure GL).

Figure 3 reveals inverted U-shaped curves for both methods: performance initially improves as GL data increases, confirming that domain knowledge acquisition is beneficial, but excessive GL allocation degrades performance—generative and contrastive learning are complementary, and neither alone suffices.

Crucially, our IB-constrained approach consistently outperforms standard generative learning

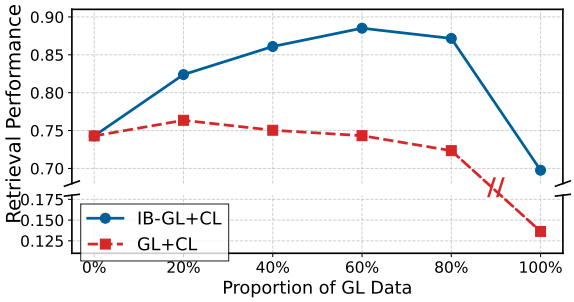

Figure 3: **Impact of GL data allocation.** We fix the total training budget at 100k samples and vary the GL data ratio (r_{learn}) from 0% to 100% in Stage 1. Both our IB-constrained method and standard GL are evaluated under identical data allocation settings.

across all allocation ratios, with a steeper ascending slope indicating more efficient knowledge utilization. This validates that explicit semantic compression better leverages acquired knowledge for retrieval, independent of data volume.

6 Conclusion

In this work, we introduced LBR , a two-stage framework that decouples domain knowledge acquisition from representation alignment for vertical domain embedding adaptation. By inserting an attention-mask-based information bottleneck during generative learning, IB-GL forces the model to compress input semantics into a small token set before generation—acquiring domain knowledge while preserving the causal architecture needed for subsequent contrastive alignment.

Experiments across medical, chemistry, and code domains show that LBR (Qwen2.5-1.5B) outperforms the strongest contrastive baseline by +7.8 Recall@10 on average, with gains up to +13.0 in Code, while remaining the only contrastive method that fully retains generative capability. More broadly, our results suggest that the tension between generative and discriminative objectives is not inherent but architectural: a well-placed bottleneck turns destructive interference into complementary synergy. We leave adaptive bottleneck mechanisms and joint generative-contrastive optimization as future directions.

While LBR demonstrates substantial improvements in vertical domain retrieval, several limitations remain.

7 Limitations

Static Compression Ratio. The compression ratio R is currently fixed per domain (default R=500). Our ablation (Table 4) shows that the optimal R already varies across domains: entity-dense corpora with long technical descriptions may benefit from milder compression, while short-form queries tolerate aggressive ratios. A natural extension is *input-adaptive* R, where a lightweight predictor estimates the information density of each sample and assigns a per-instance bottleneck size, automatically balancing the IB trade-off between compression and sufficiency without manual tuning.

Factual Knowledge vs. Reasoning. Stage 1 currently acquires factual domain knowledge (terminology, entity relationships) via next-token prediction. For domains that demand multi-step reasoning—such as legal case analysis or financial risk assessment—factual grounding alone is insufficient. A promising direction is to incorporate *Chain-of-Thought embeddings* (Li et al., 2024): by replacing or augmenting the generative target Y with explicit reasoning chains, the bottleneck tokens would be trained to compress not only domain facts but also inferential pathways, potentially yielding embeddings that capture logical structure alongside factual content.

Automatic Stage-1 Necessity Detection. LBR assumes that every target domain requires the IB-GL stage to inject missing knowledge. In practice, some domains may already be well-covered by the base LLM's pretraining data, making Stage 1 redundant or even harmful (e.g., introducing noise from low-quality domain corpora). Future work could train a *domain-gap predictor* that, given a domain corpus and base model, estimates whether IB-GL would yield meaningful gains—enabling a "learn only when needed" paradigm that saves computation and avoids negative transfer.

Evaluation Granularity. Our metrics (R@10, NDCG@10) measure aggregate retrieval quality but do not directly isolate the rare-entity grounding mechanism. Entity-frequency-stratified evaluation and targeted alias-resolution probes would more precisely quantify when and how IB-GL's knowledge acquisition helps.

References

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *International Conference on Learning Representations*.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2Vec: Large language models are secretly powerful text encoders. In *COLM*.

Asma Ben Abacha and Dina Demner-Fushman. 2019. A question-entailment approach to question answering. *BMC Bioinform.*, 20(1):511:1–511:23.

Vera Boteva, Demian Gholipour, Artem Sober, and Andrew Yates. 2016. Nfcorpus: A full-text learning to rank dataset for medical information retrieval. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 520–528.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*.

Xuanming Cui, Jianpeng Cheng, Hong-you Chen, Satya Narayan Shukla, Abhijeet Awasthi, Xichen Pan, Chaitanya Ahuja, Shlok Kumar Mishra, Yonghuan Yang, Jun Xiao, and 1 others. 2025. Think then embed: Generative context improves multimodal embedding. *arXiv preprint arXiv:2510.05014*.

Jingcheng Deng, Zhongtao Jiang, Liang Pang, Zihao Wei, Liwei Chen, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. 2025. Following the autoregressive nature of llm embeddings via compression and alignment. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12672–12688.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. In *Transactions on Machine Learning Research*.

Andrew Jaegle, Felix Gimeno, Andy Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pages 4651–4664.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577.

Ali Shiraee Kasmaee, Mohammad Khodadad, Mehdi Astaraki, Mohammad Arshi Saloot, Nicholas Sherck, Hamidreza Mahyar, and Soheila Samiee. 2025. Chembed: Enhancing chemical literature search through domain-specific text embeddings. *arXiv preprint arXiv:2508.01643*.

Ali Shiraee Kasmaee, Mohammad Khodadad, Mohammad Arshi Saloot, Nicholas Sherck, Stephen Dokas, Hamidreza Mahyar, and Soheila Samiee. 2024. Chemteb: Chemical text embedding benchmark, an overview of embedding models performance & efficiency on a specific domain. *arXiv preprint arXiv:2412.00532*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024a. Nv-embed: Improved techniques for training llms as generalist embedding models. *ArXiv*, abs/2405.17428.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R. Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, and 1 others. 2024b. Gecko: Versatile text embeddings distilled from large language models. *arXiv preprint arXiv:2403.20327*.

Yibin Lei, Liang Ding, Yu Cao, Changtong Zan, Andrew Yates, and Dacheng Tao. 2023. Unsupervised dense retrieval with relevance-aware contrastive pretraining. In *Findings of the Association for Computational Linguistics (ACL)*, pages 10932–10940.

Xiaoxi Li, Jiajie Li, Yujia Zhou, and 1 others. 2024. From matching to generation: A survey on generative information retrieval. *arXiv preprint arXiv:2404.14851*. Chain-of-Thought Embeddings discussed in context of reasoning-enhanced retrieval.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Ailiang Lin, Zhuoyun Li, Kotaro Funakoshi, and Manabu Okumura. 2025. Causal2vec: Improving decoder-only llms as versatile embedding models. *arXiv preprint arXiv:2507.23386*.

Yuxiang Liu, Tian Wang, Gourab Kundu, Tianyu Cao, Guang Cheng, Zhen Ge, Jianshu Chen, Qingjun Cui, and Trishul Chilimbi. 2025. Exploring reasoning-infused text embedding with large language models for zero-shot dense retrieval. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 4981–4985.

Zheng Liu and Yingxia Shao. 2022. Retromae: Pretraining retrieval-oriented transformers via masked auto-encoder. *CoRR*.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems*.

Niklas Muennighoff, SU Hongjin, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2025. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 2014–2037.

Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic embed: Training a reproducible long context text embedder. *arXiv preprint arXiv:2402.01613*.

Dan Qiao, Yuan Gao, Zheming Yang, Di Yang, Ziheng Wu, Pengcheng Lu, Minghui Qiu, Juntao Li, and Min Zhang. 2025. Decoder-only llms can be masked auto-encoders. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 713–723.

Ignacio Sastre and Aiala Rosá. 2025. Memory tokens: Large language models can generate reversible sentence embeddings. *arXiv preprint arXiv:2506.15001*.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raber. 2024. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*.

Chang Su, Dengliang Shi, Siyuan Huang, Jintao Du, Changhua Meng, Yu Cheng, Weiqiang Wang, and Zhouhan Lin. 2025. Training llms to be better text embedders through bidirectional reconstruction. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4351–4369.

Jiashuo Sun, Shixuan Liu, Zhaochen Su, Xianrui Zhong, Pengcheng Jiang, Bowen Jin, Peiran Li, Weijia Shi, and Jiawei Han. 2025a. Grace: Generative representation learning via contrastive policy optimization. *arXiv preprint arXiv:2510.04506*.

Yidan Sun, Mengying Zhu, Feiyue Chen, Yangyang Wu, Xiaolei Dan, Mengyuan Yang, Xiaolin Zheng, and Shenglin Ben. 2025b. Termgpt: Multi-level contrastive fine-tuning for terminology adaptation in legal and financial domain. *arXiv preprint arXiv:2511.09854*.

Yixuan Tang and Yi Yang. 2024. Do we need domain-specific embedding models? an empirical investigation. *arXiv preprint arXiv:2409.18511*.

Yixuan Tang and Yi Yang. 2025. FinMTEB: Finance massive text embedding benchmark. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3620–3638.

N TISHBY. 2000. The information bottleneck method. *Computing Research Repository (CoRR)*.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *arXiv preprint arXiv: Arxiv-2402.10176*.

Yu-Che Tsai, Kuan-Yu Chen, Yuan-Chi Li, Yuan-Hao Chen, Ching-Yu Tsai, and Shou-De Lin. 2025. Let llms speak embedding languages: Generative text embeddings via iterative contrastive refinement. *arXiv preprint arXiv:2509.24291*.

Hayato Tsukagoshi and Ryohei Sasano. 2025. Redundancy, isotropy, and intrinsic dimensionality of prompt-based text embeddings. *arXiv preprint arXiv:2506.01435*.

Haochun Wang, Chi Liu, Nuwa Xi, Zewen Qiang, Sendong Zhao, Bing Qin, and Ting Liu. 2023. Huatuo: Tuning llama model with chinese medical knowledge. *arXiv preprint arXiv:2304.06975*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning (ICML)*, pages 9929–9939. PMLR.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. 2025. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding.

Shunyu Yao, Qingqing Ke, Qiwei Wang, Kangtong Li, and Jie Hu. 2024. Lawyer gpt: A legal large language model with enhanced domain knowledge and reasoning capabilities. In *Proceedings of the 2024 3rd International Symposium on Robotics, Artificial Intelligence and Information Engineering*, pages 108–112.

Domain	Total	GL	CL	Eval
Medical	220k	150k	50k	20k
Code	220k	150k	50k	20k
Chemistry	220k	150k	50k	20k
All	**660k**	**450k**	**150k**	**60k**

Table 6: **Dataset split summary.** All domains share the same budget allocation.

Caojin Zhang, Qiang Zhang, Ke Li, Sai Vidyaranya Nuthalapati, Benyu Zhang, Jason Liu, Serena Li, Lizhu Zhang, and Xiangjun Fan. 2025a. Gem: Empowering llm for both embedding generation and language understanding. *arXiv preprint arXiv:2506.04344*.

Di Zhang, Wei Liu, Qian Tan, Jingdan Chen, Hang Yan, Yuliang Yan, Jiatong Li, Weiran Huang, Xiangyu Yue, Dongzhan Zhou, Shufei Zhang, Mao Su, Han-Sen Zhong, Yuqiang Li, and Wanli Ouyang. 2024. Chemllm: A chemical large language model. *Preprint*, arXiv:2402.06852.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025b. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, Zhenyu Liu, Dongfang Li, Xinyuan Wei, and 1 others. 2025. Kalm-embedding-v2: Superior training techniques and data inspire a versatile embedding model. *arXiv preprint arXiv:2506.20923*.

Zheli Zhou, Chenxu Zhu, Jianghao Lin, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2025. Generative representational learning of foundation models for recommendation. *arXiv preprint arXiv:2506.11999*.

A Data

A.1 Datasets

We provide a comprehensive overview of all datasets used in this work. Table 6 summarizes the key statistics across all three domains. Each domain follows an identical data budget: 150k samples for Stage 1 (GL/IB-GL), 50k for Stage 2 (CL), and 20k for held-out evaluation.

Medical Domain. We combine MedQuAD (Ben Abacha and Demner-Fushman, 2019) (NIH-sourced consumer health QA on diseases, drugs, and tests) and PubMedQA (Jin et al., 2019) (expert-level biomedical reasoning from scientific abstracts) to cover both general medical knowledge

and research reasoning. For evaluation, we use a held-out Flashcard set testing precise medical terminology retention.

Code Domain. We integrate KodCode (Xu et al., 2025) (synthetic coding tasks with unit tests, from basic algorithms to competitive programming) and OpenMath (Toshniwal et al., 2024) (step-by-step mathematical reasoning). A held-out subset of 20k samples is reserved for evaluation.

Chemistry Domain. We use ChemData700K (Zhang et al., 2024), aggregating ∼730k samples from PubChem, ChEMBL, USPTO, and ChemXiv across nine tasks in three categories: *Molecules* (property prediction, Mol2Caption), *Reactions* (retrosynthesis, yield prediction), and *Domain-specific* queries. A held-out set of 20k samples is reserved for evaluation.

A.2 Data Format Standardization

A key preprocessing step is converting heterogeneous raw data formats into a unified QA structure compatible with both the generative learning (GL) and contrastive learning (CL) stages. We standardize all samples into the ShareGPT conversation format with two turns: a `user` turn containing the question and an `assistant` turn containing the answer. Table 7 illustrates representative examples from each domain.

For the GL/IB-GL stage, the unified format naturally maps to a conditional generation task: the model receives the user query as input and learns to generate the assistant response. For the CL stage, we reinterpret each QA pair as a (query, positive document) pair, where the question is the query and the ground-truth answer is the relevant document to be retrieved. Negatives are drawn in-batch from other samples.

This dual-use design means the *same* preprocessed data can serve both training stages without additional annotation, ensuring consistency between the knowledge acquired during generation and the representations learned during contrastive fine-tuning.

A.3 Data Partitioning Protocol

To ensure reproducibility and strict train–test isolation, we follow a deterministic partitioning protocol for each domain. The procedure is as follows:

1. **Unique ID Assignment.** Each raw sample is assigned a unique deterministic identifier based on its source index before any preprocessing. For multi-source domains (Medical, Code), IDs are prefixed with the source name (e.g., `medquad_00042`, `kodcode_12345`).
2. **Shuffling.** All samples within each domain are shuffled using a fixed random seed (`seed=42`) to ensure reproducibility.
3. **Sequential Partitioning.** From the shuffled pool, we sequentially allocate: the first 150k samples to Stage 1 (GL/IB-GL), the next 50k to Stage 2 (CL), and the following 20k to the evaluation split. Any remaining samples are discarded. This sequential allocation after shuffling guarantees zero overlap by construction.
4. **Integrity Verification.** After partitioning, we programmatically verify that $|\text{GL} \cap \text{CL}| = |\text{GL} \cap \text{Eval}| = |\text{CL} \cap \text{Eval}| = 0$ at the ID level, and that $|\text{GL}| + |\text{CL}| + |\text{Eval}| = 220\text{k}$.

This protocol ensures that (1) the data split is fully deterministic and reproducible, (2) no sample appears in more than one split, and (3) the evaluation set is strictly held out from all training phases.

A.4 Data Deduplication and Leakage Analysis

Since our primary evaluation relies on held-out splits from the same data sources used for training, we conduct rigorous deduplication and leakage checks to ensure evaluation integrity. Below we describe the full pipeline applied to all three domains.

Train–Test Isolation. For each domain, we first assign unique IDs to all raw samples before any processing. The data is then partitioned into three *mutually exclusive* splits via the deterministic protocol described in Appendix A.3: 150k for Stage 1 (GL/IB-GL), 50k for Stage 2 (CL), and 20k for evaluation. We verify zero set intersection at the ID level across all three splits. No sample appears in more than one split.

Near-Duplicate Detection. Beyond exact deduplication, we detect *near-duplicates* that could inflate retrieval scores through surface-level similarity. We apply a two-stage pipeline:

1. **MinHash + LSH** (Locality-Sensitive Hashing): We compute MinHash signatures (n-gram = 5, 128 permutations) for all training and evaluation samples, then query each evaluation sample against the training set using LSH with a Jaccard similarity threshold of

Domain	Raw Format	Original Sample (Truncated)	Unified QA Format
Medical	QA pair	`Q:` What are the symptoms of diabetes? `A:` Common symptoms include frequent urination...	`[User]` What are the symptoms of diabetes? `[Assistant]` Common symptoms include frequent urination...
Code	Problem + Solution	`Problem:` Given an array of integers...`Solution: def twoSum(nums, target):...`	`[User]` Given an array of integers...`[Assistant] def twoSum(nums, target):...`
Chemistry	Instruction–Response	`Input:` Predict the product of...`Output:` The major product is...	`[User]` Predict the product of...`[Assistant]` The major product is...

Table 7: **Data format standardization examples.** All raw formats are converted into a unified two-turn QA conversation. During GL/IB-GL, the model generates the assistant response autoregressively; during CL, the question serves as the query and the answer as the positive document.

$\tau = 0.8$. Any flagged pairs are manually inspected.

2. **Embedding-based similarity**: Using a frozen general-purpose encoder (BGE-base), we encode all training and evaluation samples, then compute cosine similarity between each evaluation query and all training samples. We flag pairs with cosine similarity > 0.95 for manual review.

Across all three domains, both methods flagged a small number of evaluation samples as potential near-duplicates. Manual inspection of flagged cases revealed that they were thematically related but semantically distinct (e.g., two questions about the same drug but asking about different properties), with no true duplicates or paraphrases identified.

Negative Difficulty Analysis. To address the concern that in-domain negatives may be insufficiently challenging, we note that the base Qwen2-1.5B model (without any domain training) achieves very low retrieval performance on our evaluation sets: R@10 of 0.100 in Chemistry, 0.410 in Medical, and 0.151 in Code (Table 1). These low scores indicate that our evaluation sets are non-trivial—even a strong pre-trained LLM cannot easily distinguish positives from negatives, confirming that the candidate pool contains genuinely challenging distractors.

We additionally evaluate on external standardized benchmarks (ChemTEB-Retrieval, NFCorpus, CodeSearchNet; see Section 4.2 and Appendix A.5 for details), which are entirely independent of our data pipeline.

A.5 External Benchmark Descriptions

To provide evaluation results that are fully independent of our training pipeline, we evaluate on three publicly available benchmarks. Below we describe each benchmark and explain why it is disjoint from our training data.

ChemTEB-Retrieval (Kasmaee et al., 2024). ChemTEB (Chemical Text Embedding Benchmark) is a community-constructed benchmark for evaluating text embeddings in the chemical domain. We use its retrieval subset, which requires matching chemical queries to relevant documents sourced from PubChem compound descriptions, ChEMBL bioactivity records, and chemical literature abstracts. The retrieval subset contains approximately 5k queries with a corpus of 50k candidate documents. Importantly, although our chemistry training data (ChemData700K) also draws from PubChem and ChEMBL, ChemTEB was independently curated with its own query generation pipeline and document selection criteria, and we verified zero overlap at the document level between ChemTEB's retrieval corpus and our training split.

NFCorpus (Boteva et al., 2016). NFCorpus is a biomedical information retrieval dataset included in MTEB, containing 3,244 nutrition and fitness-related queries matched against 169k PubMed abstracts. The queries were derived from the NutritionFacts.org website and represent real user information needs about dietary health. This benchmark tests cross-topic generalization: while our medical training data (MedQuAD + PubMedQA) focuses on clinical questions about diseases, drugs, and medical procedures, NFCorpus targets consumer health queries about nutrition and fitness—a complementary but distinct subdomain.

Baseline	Data	Objective	Attn.	Pooling
BGE	General	CL	Bi.	[CLS]
GTE	General	CL	Bi.	Mean
LLM2Vec	General	CL	Bi.	Mean
ChemLLM	Domain	PT	Causal	Last
HuaTuo	Domain	PT	Causal	Last
SFT+CL	Domain (200k)	GL→CL	Causal	Last Z
LBR	Domain (200k)	IB-GL→CL	Causal	Last Z

Table 8: **Baseline training protocol.** "General" = trained on general data (off-the-shelf); "Domain (200k)" = trained on identical domain corpora (150k GL + 50k CL). All domain-adapted methods share the same data splits and hyperparameter search budget.

CodeSearchNet (Husain et al., 2019). CodeSearchNet is a large-scale code retrieval benchmark from MTEB, where natural language documentation queries are matched against code function implementations across six programming languages (Python, Java, Go, PHP, JavaScript, Ruby). The dataset contains approximately 10k queries and 100k code snippets. This is entirely independent of our code training data: KodCode contains algorithm problems with solutions, and OpenMath contains mathematical reasoning—neither includes the documentation-to-code matching task that CodeSearchNet evaluates.

Results and Analysis. Table reports the results on external benchmarks. General-purpose contrastive models (BGE, GTE, LLM2Vec) perform reasonably on CodeSearchNet but show limited domain specialization on ChemTEB-Retrieval (NDCG@10: 47–52). The SFT+CL baseline achieves the highest baseline score on ChemTEB (55.4) but *degrades* on NFCorpus (33.6 vs. 36–39 for off-the-shelf models), confirming that naive domain SFT can overfit at the cost of broader generalization. Our LBR model achieves the best average score (47.1), outperforming LLM2Vec (45.6) and SFT+CL (41.8), while exhibiting more balanced cross-domain transfer—improving on ChemTEB without sacrificing NFCorpus or CodeSearchNet performance.

A.6 Baseline Training Protocols

Table 8 summarizes the training protocol for each baseline method compared in our experiments. For fair comparison, all domain-adapted methods (SFT+CL, LBR) use identical data splits and hyperparameter search budgets.

General-Purpose Embedders (BGE, GTE, LLM2Vec). Pre-existing models evaluated *off-the-shelf* using released checkpoints. These were trained on general data and we aim to measure whether general-purpose embedders transfer to vertical domains without additional domain adaptation.

Domain-Adapted LLMs (ChemLLM, HuaTuo). Pretrained language models continually trained on domain-specific corpora, such as ChemLLM (Yao et al., 2024) for chemistry and HuaTuo (Wang et al., 2023) for the medical domain.

SFT+CL. Uses the *exact same* generative data and targets as LBR but *without* the bottleneck constraint. This baseline consistently underperforms even CL-only methods (e.g., 64.4 vs. 79.3 Avg in Table 1), demonstrating that naive domain SFT before CL actually *hurts* embedding quality. The dramatic improvement of LBR over SFT+CL (+23.4 Avg points on the same backbone and data) isolates the contribution of the bottleneck mechanism itself.

B Experimental Setup Details

B.1 Evaluation Protocol

To rigorously assess the quality of the learned representations, we adapt the **MTEB (Massive Text Embedding Benchmark)** framework for our specific domain requirements.

Embedding Conversion via llm2vec. Since our base models are decoder-only generative architectures, we utilize the **llm2vec** methodology to transform them into effective text encoders. This allows us to extract dense vector representations from the model checkpoints (both GL and IB-GL) for downstream discrimination tasks.

Custom Domain Retrieval. Instead of relying solely on generic open-domain benchmarks, we construct custom Information Retrieval (IR) tasks using the held-out evaluation splits of our three vertical domains (Medical, Code, Chemistry).

- **Task Setup:** We formulate the evaluation as a retrieval task where the questions (Q) serve as queries and the corresponding ground-truth answers (A) serve as the documents to be retrieved.

- **Metrics:** We report standard retrieval metrics such as **NDCG@10** (Normalized Discounted Cumulative Gain), which measures the model's ability to rank the correct domain-

Hyperparameter	Value
Stage 1: IB-GL (SFT variant)	
Backbone	Qwen2(.5)-1.5B / Llama3.2-1B
Learning rate	2e-5
Batch size	128
Max seq. length	512 (X) + 512 (Y)
Bottleneck tokens $\|Z\|$	1–4 (adapts to R)
Compression ratio R	500 (default)
Training steps	3,000
Warmup ratio	0.03
Optimizer	AdamW (β_1=0.9, β_2=0.95)
Weight decay	0.01
Stage 2: Contrastive Learning	
Learning rate	1e-5
Batch size	256
Temperature τ	0.02
Negatives	In-batch (255 per query)
Training steps	1,000
Pooling	Last Z token, final layer
Attention mask	Causal (preserved)
Data budget per domain	
Stage 1 (GL/IB-GL)	150k samples
Stage 2 (CL)	50k samples
Evaluation (held-out)	20k samples

Table 9: **Training hyperparameters.** All experiments use identical configurations unless otherwise noted.

specific answers highest among the candidates.

B.2 Implementation Details

Table 9 reports the full training configuration for reproducibility.

Attention Mask Specification. Our bottleneck attention mask $\mathbf{M} \in \{0,1\}^{(|X|+|Z|+|Y|)\times(|X|+|Z|+|Y|)}$ follows three rules: (1) tokens in X attend causally within X; (2) tokens in Z attend to all of X and causally within Z; (3) tokens in Y attend to Z and causally within Y, but *cannot* attend to X (the "information cut-off"). This is illustrated in Figure 2 and enforced via a block-structured mask applied before the softmax in every attention layer.

Bottleneck Token Initialization. Bottleneck tokens Z are initialized as copies of the model's end-of-turn token embedding. They are fully trainable and shared across all attention layers (i.e., the same $|Z|$ positions receive gradients from all layers).

Pooling. The sequence embedding $\mathbf{v}$ is the hidden state of the *last* bottleneck token at the final transformer layer, L2-normalized before computing cosine similarity.

Method	Retrieval	Training Time
Standard SFT	45.98	1.0×
IB-GL (SFT)	80.60	1.03×
Standard PT	43.25	1.0×
IB-GL (PT)	81.36	3.9×

Table 10: Performance and efficiency comparison between SFT and PT paradigms. IB-GL (SFT) achieves superior retrieval performance with negligible overhead, while IB-GL (PT) offers a scalable solution at higher computational cost.

B.3 Training Efficiency Analysis

We analyze the computational overhead of IB-GL under both SFT and PT paradigms, comparing against their respective baselines.

IB-GL with SFT. Our IB-GL approach under the SFT paradigm incurs minimal computational overhead compared to standard SFT, with the training time increasing by only 3%. This efficiency stems from our architectural design: we introduce no additional modules, merely inserting a small number of bottleneck tokens and modifying the attention mask through parallelized matrix operations. Consequently, the computational complexity remains nearly identical to standard SFT.

IB-GL with PT. The PT paradigm exhibits higher computational costs due to the reconstruction task. Since the input sequence must be reconstructed, the sequence length doubles, resulting in approximately $4\times$ training time due to the quadratic attention complexity.

SFT vs. PT. Table 10 compares the retrieval performance and training efficiency of both paradigms. SFT achieves slightly better retrieval performance than PT, as the carefully curated query-answer pairs enable more efficient learning. The PT paradigm, while less efficient, provides a viable alternative when supervised data is unavailable, trading computational cost for data scalability.

In practice, we recommend SFT when labeled data is available for its superior efficiency. PT serves as an alternative for scenarios with abundant unlabeled corpora but limited supervision.

C Proofs

C.1 Proof of Proposition 1

Proof. In a Transformer, the hidden state $\mathbf{h}_i^{(\ell)}$ at position i and layer ℓ is computed as $\mathbf{h}_i^{(\ell)} =$

$f^{(\ell)}(\mathbf{h}_i^{(\ell-1)}, \{\mathbf{h}_j^{(\ell-1)} : \mathbf{M}_{i,j} = 1\})$. For any target position t, the mask enforces $\mathbf{M}_{t,j} = 0$ for all $j \in X$. Therefore $\mathbf{h}_t^{(\ell)}$ depends only on $\{Z, Y_{\leq t}\}$ at every layer, and the autoregressive output $q_\theta(y_t \mid Z, Y_{<t})$ is independent of X given Z. Applying the chain rule: $q_\theta(Y \mid X, Z) = \prod_t q_\theta(y_t \mid Z, Y_{<t}) = q_\theta(Y \mid Z)$. □

C.2 Proof of Proposition 2

Proposition 2 (Generative Loss as Variational Lower Bound)**.** *Let $q_\theta(Y \mid Z) = \prod_{t=1}^{|Y|} q_\theta(y_t \mid Z, Y_{<t})$ be the autoregressive model distribution. The standard next-token prediction loss:*

$$\mathcal{L}_{\text{gen}} = -\mathbb{E}_{p(X,Y)}\left[\sum_{t=1}^{|Y|} \log q_\theta(y_t \mid Z, Y_{<t})\right] \quad (7)$$

provides a variational lower bound on $I(Z;Y)$:

$$I(Z;Y) \geq H(Y) - \mathcal{L}_{\text{gen}} \quad (8)$$

Minimizing $\mathcal{L}_{\text{gen}}$ is therefore equivalent to maximizing a lower bound on the IB prediction term $I(Z;Y)$.

Proof. By definition, $I(Z;Y) = H(Y) - H(Y \mid Z)$. Introducing the variational distribution $q_\theta(Y \mid Z)$ and applying the non-negativity of the KL divergence:

$$\begin{aligned} H(Y \mid Z) &= -\mathbb{E}_{p(Z,Y)}[\log p(Y \mid Z)] \\ &\leq -\mathbb{E}_{p(Z,Y)}[\log q_\theta(Y \mid Z)] \quad (9) \end{aligned}$$

where Eq. (9) follows from $D_{\text{KL}}(p(Y|Z) \| q_\theta(Y|Z)) \geq 0$, i.e., $\mathbb{E}_p[\log p] \geq \mathbb{E}_p[\log q_\theta]$. Substituting into $I(Z;Y) = H(Y) - H(Y|Z)$:

$$I(Z;Y) \geq H(Y) + \mathbb{E}_{p(Z,Y)}[\log q_\theta(Y \mid Z)] = H(Y) - \mathcal{L}_{\text{gen}} \quad (10)$$

Since $H(Y)$ is constant w.r.t. θ, $\min_\theta \mathcal{L}_{\text{gen}} \Leftrightarrow \max_\theta I_{\text{VLB}}(Z;Y)$. □

D Information Bottleneck

Figure 4 provides a visual summary of the IB principle as applied in our framework. In LBR, X corresponds to the input sequence (query or document), Z to the bottleneck tokens, and Y to the target sequence. The attention mask (Eq. 2) enforces the Markov chain $X \to Z \to Y$, ensuring that Z captures only the minimal sufficient statistics of X needed for predicting Y. The compression ratio R controls the capacity of Z, trading off between retaining task-relevant information and discarding irrelevant noise.

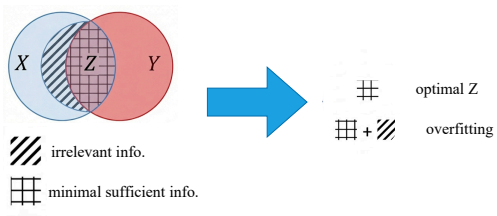

Figure 4: **Illustration of the Information Bottleneck (IB) principle.** The representation Z aims to mediate the trade-off between compressing the input X and predicting the target Y. The *minimal sufficient statistics* (grid area) represent the optimal embedding that captures only task-relevant semantics. Including *irrelevant information* (diagonal lines) leads to overfitting, where the model memorizes input noise rather than generalizing to the target domain.

D.1 IB Perspective on Two-Stage Training

The sequential structure of LBR naturally decomposes into two phases of the IB optimization:

1. **Stage 1** ($\max I(Z;Y)$ under capacity constraint): The generative objective forces Z to retain all information from X that is necessary for target reconstruction, while the architectural bottleneck limits channel capacity. This produces representations that are *maximally informative about domain-specific content* while being compact.
2. **Stage 2** ($\min I_{\text{irr}}(X;Z)$ while preserving task-relevant $I(S;Z)$): Contrastive learning further refines Z for the downstream retrieval task. Given positive pairs (q, p^+) that share semantic content S but differ in surface form, contrastive training encourages Z to be invariant to surface variation—effectively discarding task-irrelevant information while preserving semantic similarity structure.

Sequential rather than joint optimization is critical: the generative objective encourages retention of fine-grained, token-level information (high $I(X;Z)$), while the contrastive objective encourages coarser semantic invariance. Optimizing both simultaneously creates gradient conflict, as confirmed empirically in Table 2 where naive SFT+CL degrades performance. The two-stage approach resolves this by first establishing a rich, informative Z (Stage 1), then refining it for the specific task (Stage 2).

E Use of AI Assistants

In this work, we used Claude Code (Anthropic) as an AI programming assistant to help with code implementation, experiment management scripts, and proofreading of manuscript language. Specifically, the tool was employed to: (1) assist in writing and debugging training and evaluation code, (2) manage experiment configurations and result collection, and (3) polish English grammar and phrasing in the manuscript draft. All scientific contributions—including research ideas, method design, experimental design, result interpretation, and quality control of the final paper—are solely the responsibility of the human authors. The authors carefully reviewed and validated all AI-assisted outputs to ensure correctness and academic integrity.