VEDJE: Video-Efficient Discriminative Joint Encoder for Scalable Video-Text Retrieval

Shahaf Wagner [*1] **Gabriele Serussi** [*1] **Dan Ben Ami** [1] **Tomer Galanti** [2] **Chaim Baskin** [1]

Abstract

We recast scalable second-stage video-text retrieval as a *cache-design* problem and present VEDJE, a cached joint reranker for video. The difficulty is that pairwise verification needs temporal evidence, but rerankers either re-encode frames per query, which is too slow at index scale, or compress each video into a single global summary, which discards the temporal cues verification depends on. VEDJE keeps the cache *video-like*: a frozen visual backbone runs once per video and writes an ordered, frame-local cache that a 33M cross-encoder reads at query time, supported by a residual first-stage score prior and a training-only future-delta loss $\mathcal{L}_{\text{delta}}$ that biases tight caches toward frame-to-frame change. On MSR-VTT, MSVD, DiDeMo, and ActivityNet, VEDJE lifts R@1 over the matched first-stage retriever in both retrieval directions on all four datasets, reaching T2V R@1 of 56.5 on MSR-VTT and 56.8 on MSVD with the VideoCLIP-backed variant. VEDJE thereby keeps pairwise verification on the candidate path while moving its visual cost entirely to indexing time, where it is paid once per video and amortized across all future queries.

1. Introduction

Video search at scale runs in two stages: a first stage retrieves plausible candidates over the full index, and a second stage verifies whether each candidate matches the query. Verification is where video structure carries the signal (which frames appear, how objects move, when a state changes, in what order events unfold) so it requires inspecting query and candidate together.

Each existing approach satisfies only one of these stages well. Dual encoders (Luo et al., 2022; Ma et al., 2022; Wang et al., 2024a;b) index videos and queries independently and retrieve candidates cheaply, but their score cannot inspect query and video evidence together. Joint rerankers, mature for image-text retrieval (Li et al., 2022; 2023), restore that inspection at a cost that does not transfer cleanly to video: the visual encoder runs again per query, and query-time encoding, cached state, and pairwise scoring all grow with the number of sampled frames. MLLM rerankers (Ko et al., 2025; Lee et al., 2025) push pairwise reasoning further still, but their multi-billion-parameter backbones make per-query video encoding expensive at the candidate-pool sizes typically used in two-stage retrieval.

One way to keep verification deployable is to remove the visual encoder from the query path. The image cached reranker EDJE (Taraday et al., 2026) already shows how for static images: encode each image once offline, store a small set of compressed tokens, and rerank online with a lightweight joint encoder. Lifting this recipe to video runs into two distinct problems. A single global clip summary collapses exactly the temporal evidence that distinguishes video verification from image retrieval; per-frame caches that preserve everything are too large to deploy at index scale. Scalable video reranking is therefore a *cache-design problem*: the stored representation must be small enough to ship alongside an index of millions of videos, yet structured enough to remain *video-like* when read by a compact joint encoder.

VEDJE treats the cache as the central design object. A frozen visual encoder runs once per video and writes an ordered, frame-local cache: each of T sampled frames is compressed independently to M learned tokens, and the cached representation is the temporal concatenation $Z(v) = [Z_1; \ldots; Z_T]$. At query time, a 33M-parameter cross-encoder reranks first-stage candidates over $Z(v)$ and the query text, with the first-stage score injected as a residual prior; no visual encoder runs online. Cache size per video (the product of T and M) becomes the deployment knob, fixing both per-video storage and online sequence length. To prevent a tight cache from collapsing toward a static summary, a training-only auxiliary objective $\mathcal{L}_{\text{delta}}$ biases tokens at time t to predict patch-level changes between X_t and X_{t+h}, so the cache retains motion, state transitions, and

[*]Equal contribution [1]INSIGHT Lab, Ben Gurion University of the Negev, Israel [2]Texas A&M University. Correspondence to: Shahaf Wagner <shahafwg@post.bgu.ac.il>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

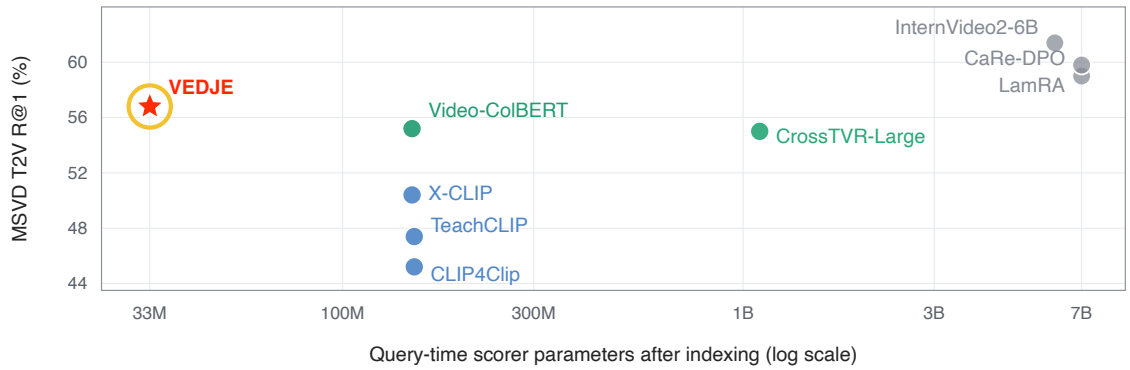

Figure 1. MSVD T2V cost-accuracy positioning. VEDJE keeps a 33M query-time scorer while retaining cached pairwise reranking.

event order.

Evaluated on MSR-VTT (Xu et al., 2016), MSVD (Chen & Dolan, 2011), DiDeMo (Hendricks et al., 2017), and ActivityNet (Krishna et al., 2017) (Figure 1), the VideoCLIP-backed VEDJE$_{\text{VCLIP}}$ (Wang et al., 2024a) lifts R@1 in both directions on all four datasets, reaching 56.5 percentage points on MSR-VTT and 56.8 on MSVD; compressing to one cached token per sampled frame nearly preserves these gains (MSR-VTT T2V R@1: 54.6 $\rightarrow$ 54.4) while shrinking per-video storage from 49 to roughly 12 kB. Ablations attribute most of the gain to temporal structure, with $\mathcal{L}_{\text{delta}}$ adding targeted improvements when caches are tight or visual dynamics are high.

Contributions. We make three contributions.

1. We recast scalable second-stage video reranking as a cache-design problem and identify why the image cached-rerank recipe (Taraday et al., 2026) does not transfer: a single global clip summary collapses temporal evidence, and storing per-frame patch tokens is too large at index scale. The cache must be small and *video-like*.
2. We instantiate VEDJE: a frame-local compressor that produces an ordered, temporally addressable cache; a residual first-stage score prior that adds the coarse ranking back into the joint encoder; and a training-only future-delta loss $\mathcal{L}_{\text{delta}}$ that biases tight caches to retain frame-to-frame change.
3. We isolate each component on MSR-VTT, MSVD, DiDeMo, and ActivityNet. One cached token per sampled frame ($\sim$12 kB/video at BF16) nearly preserves the 64-token R@1; ordered frame-local structure carries most of the gain; $\mathcal{L}_{\text{delta}}$ adds targeted improvements when caches are tight or visual dynamics are high.

2. Method

VEDJE replaces query-time video encoding with a *cache-design problem*: the visual backbone runs once per video, offline, and writes a compact representation that a small joint encoder later reads at query time, alongside the query text and the scalar first-stage score. The cache must be small enough to ship at index scale and structured enough for pairwise verification of actions, state changes, and event order. Figure 2 groups the three components that the rest of the section develops: (A) the offline cache, (B) the training-only supervision that keeps the cache video-like, and (C) the online reranker that reads it.

2.1. Setup and Notation

A first-stage retriever returns, for each text query q, a candidate set $\mathcal{C}(q) = \{v_1, \dots, v_K\}$ together with a coarse score $\rho(q, v) \in \mathbb{R}$. Each candidate is a video sampled into T frames, $v = \{f_t\}_{t=1}^{T}$. A frozen visual encoder g_ϕ maps each frame f_t to patch features $X_t = g_\phi(f_t) \in \mathbb{R}^{P \times d_v}$, where P is the number of patches and d_v is the visual feature dimension. VEDJE writes a cache $Z(v) \in \mathbb{R}^{TM \times d_\ell}$ once per video, storing M tokens per frame. At query time, a reranker produces a score $s_\theta(q, v) \in \mathbb{R}$ from q, $Z(v)$, and $\rho(q, v)$, and sorting $\mathcal{C}(q)$ by s_θ gives the second-stage ranking. The product TM controls both per-video storage and online sequence length, and is the single deployment knob optimized in the rest of this section.

2.2. A Frame-Local Cache that Stays Video-Like

The cache $Z(v)$ must satisfy two opposing constraints. It must be small enough to ship per video at index scale, since the visual backbone g_ϕ cannot run again at query time; and it must remain video-like, so a compact joint encoder can verify actions, state changes, and event order. A single global clip vector minimizes the first constraint and discards the second; a full per-frame patch dump does the reverse.

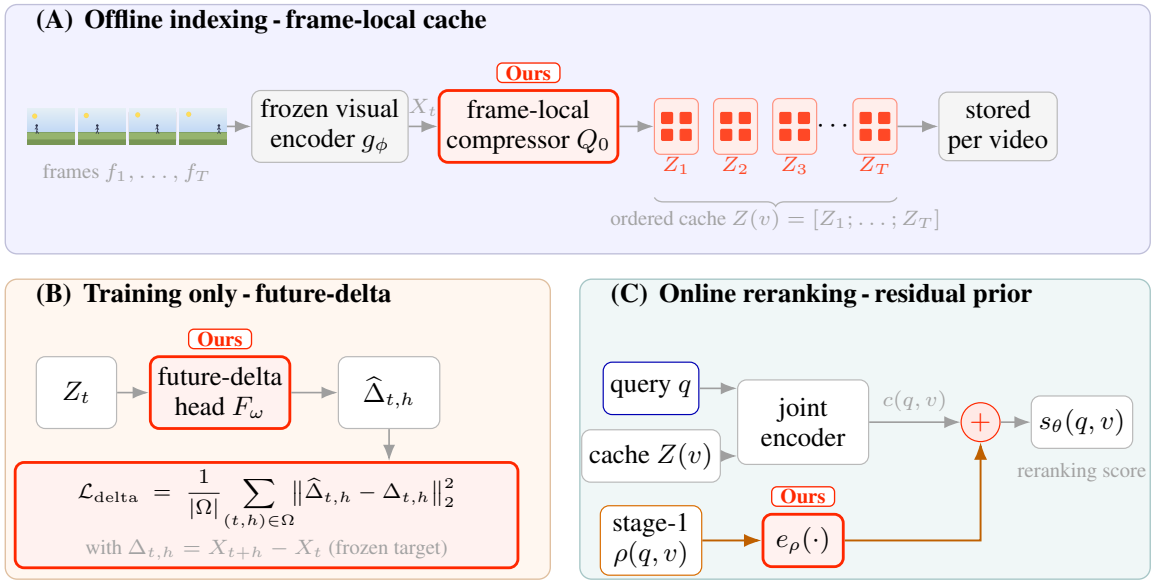

Figure 2. VEDJE pipeline; the three "Ours"-tagged modules (frame-local compressor, future-delta head, residual stage-1 prior) are our contributions. **(A) Offline indexing:** per-frame compression yields the ordered cache $Z(v) = [Z_1; \ldots; Z_T]$. **(B) Training only:** a future-delta head F_ω audits Z_t via the auxiliary loss $\mathcal{L}_{\text{delta}}$ and is discarded after training. **(C) Online reranking:** a 33M joint encoder rescores $(q, Z(v))$, and the stage-1 score $\rho(q, v)$ is injected as a residual prior through e_ρ. No visual encoder runs at query time.

VEDJE resolves the tension by decoupling *coverage* from *order*: each frame is compressed independently to a small token bank, and the bank is then placed at its temporal position.

A frame-local module with M learned cache queries $Q_0 \in \mathbb{R}^{M\times d_\ell}$ attends only to the patch tokens of the current frame,

$$K_t = X_t W_K, \qquad V_t = X_t W_V,$$
$$Z_t = \mathrm{FFN}(\mathrm{MHA}(Q_0, K_t, V_t)) \in \mathbb{R}^{M\times d_\ell}. \tag{1}$$

with $W_K, W_V \in \mathbb{R}^{d_v\times d_\ell}$. The stored cache is the temporal concatenation

$$Z(v) = [Z_1; \ldots; Z_T] \in \mathbb{R}^{TM\times d_\ell}. \tag{2}$$

This construction has two consequences. Per-frame compression bounds storage by $T \times M$ cached tokens and is independent of the patch count P of g_ϕ. Ordered concatenation keeps the cache video-like even when M is small: discarding patch tokens still leaves the fact that the remaining evidence arrived in a particular order. The cache-structure ablation in Section 3.4 contrasts this design with image-only and unordered multi-frame caches under matched cache size.

2.3. Future-Delta Supervision

The cache is query independent: once $Z(v)$ is written, every future query about v (about objects, actions, or temporal relations) reads the same tokens. When M is small, the compressor faces a static-on-average shortcut: store appearance that is informative for the easiest cases and let the rest go. The discarded information is exactly what distinguishes "before," "after," and "doing" from generic appearance.

$\mathcal{L}_{\text{delta}}$ disrupts that shortcut by auditing the cache for action cues that are already visible at frame t. A single frame typically carries hints of what is about to happen (a winding-up arm, a foot lifting off the ground, an opening door) and these hints are what action-aware retrieval needs at query time. The objective is not to memorize the future inside the cache; it is to certify, through training, that Z_t keeps enough of the present frame to predict how its frozen patch features will change a few steps later. A lightweight, training-only predictor F_ω does exactly that,

$$\widehat{\Delta}_t = F_\omega(Z_t) \in \mathbb{R}^{|\mathcal{H}|\times P\times d_v},$$
$$\Delta_{t,h} = X_{t+h} - X_t \in \mathbb{R}^{P\times d_v}. \tag{3}$$

The target is a delta rather than the absolute future feature: an absolute target can be matched by reconstructing static content shared with neighbouring frames, leaving action cues optional, whereas a delta target rewards exactly what changes between t and $t + h$. The predictor outputs one prediction per horizon and patch, so the loss is spatially localized: the cache is pushed to preserve *where* in the frame the action-implying signal lives (a hand, a foot, an object boundary) not a single pooled motion statistic.

The loss averages the squared ℓ_2 distance to the frozen target over the valid time-horizon pairs $\Omega = \{(t, h) \mid 1 \le t \le$

$T,\ h \in \mathcal{H},\ t+h \le T\}$,

$$\mathcal{L}_{\text{delta}} = \frac{1}{|\Omega|} \sum_{(t,h)\in\Omega} \left\| \widehat{\Delta}_{t,h} - \Delta_{t,h} \right\|_2^2, \tag{4}$$

where $\|\cdot\|_2^2$ is the per-element-averaged squared ℓ_2 norm over patches and the visual feature dimension; the frozen g_ϕ makes $\Delta_{t,h}$ a fixed regression target, so $\mathcal{L}_{\text{delta}}$ regularizes Z_t rather than the visual backbone. After training, F_ω is discarded and online reranking pays no auxiliary cost: $\mathcal{L}_{\text{delta}}$ changes *what* the cache stores rather than *how* it is read at deployment, with the strongest predicted effect when the cache is tight or the video has high adjacent-frame change (Section 3.5).

2.4. Cached Joint Reranking with a Residual Prior

The reranker is the only component on the query path: a compact transformer encoder that tokenizes q, concatenates the text tokens with the cached video tokens $Z(v)$, and produces a pooled pairwise representation $c(q,v) \in \mathbb{R}^d$. Its own learned positional embedding covers the full input $[\text{tokens}(q);\ Z(v)]$, so the temporal layout fixed at indexing time (Eq. 2) is read by self-attention rather than treated as a bag of tokens. Because $c(q,v)$ is computed after the query and the cache see each other, it carries the joint signal that a dual-encoder dot product cannot recover.

The first-stage score $\rho(q,v)$ carries different information. It contains no new visual evidence, and a monotone function of ρ alone cannot recover candidates that the first stage has already ordered poorly. It does, however, encode the coarse ranking that produced the candidate list; discarding it forces the joint encoder to learn an absolute ranking from a small training set. VEDJE therefore injects ρ as a residual prior, so the joint encoder learns corrections *around* the existing ordering,

$$\begin{aligned} \tilde{c}(q,v) &= c(q,v) + e_\rho\big(\rho(q,v)\big), \\ s_\theta(q,v) &= w^\top \tilde{c}(q,v) + b. \end{aligned} \tag{5}$$

where $e_\rho : \mathbb{R} \to \mathbb{R}^d$ is a small learned MLP that lifts the scalar prior into the joint representation space, and w, b are the final scoring parameters. The two signals play distinct roles as cached video evidence supplies pairwise verification, and the prior supplies coarse residual guidance, and Section 3.6 tests them separately.

2.5. Training Objective and Deployment Ledger

Training has to do two things at once: make the joint reranker accurate, and keep the online path cheap. VEDJE couples four objectives so that each component is shaped by a loss whose work matches its role at deployment,

$$\mathcal{L} = \mathcal{L}_{\text{vtm}} + \mathcal{L}_{\text{vtc}} + \mathcal{L}_{\text{mlm}} + \mathcal{L}_{\text{delta}}. \tag{6}$$

All four terms enter at equal weight: each is a standard objective for its head, and the loss-stack ablation in Appendix A.13 reads each contribution at this fixed weighting. The matching loss $\mathcal{L}_{\text{vtm}}$ supervises the pairwise score $s_\theta(q,v)$ directly, ranking matched pairs above in-batch negatives. The contrastive loss $\mathcal{L}_{\text{vtc}}$ aligns the joint encoder's text representation with the first-stage retriever's video-text embedding space: a linear head projects the joint encoder's text CLS into that space, and symmetric in-batch InfoNCE pulls each projected text vector toward the frozen first-stage video embedding of its matching video. The first-stage score $\rho(q,v)$ is itself a similarity in this same space, so the residual addition in Eq. 5 combines $c(q,v)$ and $e_\rho(\rho)$ in compatible geometries rather than as independently learned signals. The masked-language loss $\mathcal{L}_{\text{mlm}}$ predicts masked caption tokens from the surrounding text and the cached video tokens, forcing the joint encoder to read both modalities. $\mathcal{L}_{\text{delta}}$ reshapes the cache rather than the reranker: by requiring Z_t to predict near-future patch deltas (Eq. 4), it audits the offline tokens for action cues the other three objectives do not directly supervise.

After training, the future-delta predictor F_ω is discarded; the candidate path runs only the query text encoder, the 33M joint reranker, and the scalar prior e_ρ. Algorithm 1 in Appendix A.1 writes out indexing and reranking. Section 3 evaluates this cost regime: per-video storage controlled by TM, a 33M online scorer, no query-time visual feature extraction.

3. Experiments

The central claim is that an ordered, frame-local token cache supports joint pairwise verification with a deployment cost determined by TM and the 33M online reranker, with no query-time visual encoder. Five experiments isolate the components of this claim by changing one factor at a time: the cost-accuracy operating regime (Section 3.2), the cache budget (Section 3.3), the cache structure (Section 3.4), the future-delta auxiliary loss (Section 3.5), and the first-stage prior (Section 3.6). Appendices A.7 and A.13 report candidate-pool sensitivity and the complementary training-loss ablation.

3.1. Experimental Setup

VEDJE is evaluated on MSR-VTT (Xu et al., 2016), MSVD (Chen & Dolan, 2011), DiDeMo (Hendricks et al., 2017), and ActivityNet (Krishna et al., 2017), using the standard splits and Recall@K in both retrieval directions: text-to-video (T2V) and video-to-text (V2T). All recall values and absolute recall changes are reported in percentage points throughout. A first-stage retriever returns the top-K candidates (default K=20); VEDJE reranks them from cached video tokens. Unless otherwise stated, abla-

tions change one factor relative to the 64-token BF16 cache with the VideoPrism prior (Zhao et al., 2024). A variant is written VEDJE$_{S1}$-, where the subscript names the stage-1 candidate generator - VideoPrism (VP) (Zhao et al., 2024), VideoCLIP-XL (VCLIP) (Wang et al., 2024a), or Perception Encoder (PE) (Bolya et al., 2025). Parenthesized values next to VEDJE R@1 entries report the absolute gain over the matched stage-1 retriever. The joint reranker is fixed to MiniLM-L12-uncased (Wang et al., 2020), a compact 12-layer language model with ∼33M parameters, in all experiments. It reads the cached video tokens alongside the query text; no online visual encoder runs at query time, and the cached tokens occupy 12-49 kB per video.

3.2. Q1: Where Does VEDJE Sit in Cost-Accuracy Space?

Comparing absolute Recall@K across systems conflates accuracy with the cost regime that produces it. Three regimes appear in the recent literature: dual encoders that score cached embeddings (Luo et al., 2022; Ma et al., 2022; Tian et al., 2024), late-interaction methods that cache richer unimodal tokens (Reddy et al., 2025), and heavy rerankers that score sampled frames or captions online (Dai et al., 2025; Liu et al., 2025; Lee et al., 2025). Figure 1 groups methods by query-time scorer size; VEDJE sits in a fourth regime, cached joint reranking, where pairwise verification remains after the visual encoder leaves the query path. Table 1 quantifies accuracy within and across these regimes on MSR-VTT; LamRA values are from the CaRe-DPO reproduction (Lee et al., 2025).

Table 1 is the main positioning result. VEDJE$_{VP}$ matches CrossTVR-Large R@1 with roughly 33× fewer online parameters and no query-time frame extraction; VEDJE$_{VCLIP}$ reaches 56.5, the strongest cached result in the table. LamRA and CaRe-DPO are higher in absolute R@1, but they score sampled frames or captions online with 7B-parameter MLLMs, so they occupy a different deployment regime.

Across datasets, the same operating point improves its own first-stage retriever rather than merely swapping in a stronger backbone: VEDJE$_{VCLIP}$ improves every reported dataset/direction, with the largest lift on DiDeMo T2V (+8.9), and VEDJE$_{VP}$ improves MSR-VTT, MSVD, and ActivityNet in both directions. Full R@1/R@5/R@10 tables, including matched stage-1 baselines and all external baselines, appear in Appendix A.6.

Cache size and online-parameter counts are proxies for cost, and a deployable claim needs direct measurement. Table 2 reports indexing time, per-query reranking time, and peak GPU memory on a single NVIDIA L40S for the default VEDJE$_{VP}$ (Zhao et al., 2024) and the most aggressive VEDJE$_{VP}$-16 budget.

Table 1. Deployment regimes on MSR-VTT T2V retrieval. "Cached" means no query-time visual feature extraction. "Online params" counts trainable candidate-path parameters after indexing; VEDJE gains are absolute R@1 over the matched stage-1 retriever.

Method	Scoring / evidence	Cached	Online params	R@1
Embedding and late-interaction retrieval				
CLIP4Clip (Luo et al., 2022)	single-vector / video embedding	✓	151M	44.5
X-CLIP (Ma et al., 2022)	multi-frame / frame-video embeddings	✓	149M	49.3
Video-ColBERT (Reddy et al., 2025)	late interaction / late tokens	✓	149M	51.5
Heavy online video reranking or scoring				
CrossTVR-Large (Dai et al., 2025)	pairwise reranking / 12 sampled frames	✗	1.1B	54.0
LamRA (Liu et al., 2025)	MLLM scoring / sampled frames	✗	7B	59.7
CaRe-DPO (Lee et al., 2025)	MLLM scoring / frames + captions	✗	7B	64.1
Compact cached video reranking				
VEDJE$_{VP}$ (Zhao et al., 2024)	**pairwise reranking / video tokens**	✓	**33M**	**54.6** ±0.1 (+4.5)
VEDJE$_{VCLIP}$ (Wang et al., 2024a)	**pairwise reranking / video tokens**	✓	**33M**	**56.5** ±0.2 (+6.4)
VEDJE$_{PE}$ (Bolya et al., 2025)	**pairwise reranking / video tokens**	✓	**33M**	**53.9** ±0.2 (+6.3)

Table 2. Cost profile of VEDJE on a single NVIDIA L40S, MSR-VTT 1ka, K=100 candidates, batch 1, FP16, with caches already resident on device. "Compress" is the offline per-video cost of writing the cache; "Cross-encoder" is the online per-query cost for all K candidates. Lower is better in all columns.

Pipeline	Compress ms/video	Rerank ms/query	Peak GPU GB
VEDJE$_{VP}$, 4 tok/frame	0.80	3.03	0.99
VEDJE$_{VP}$-16, 1 tok/frame	0.78	1.73	0.78

Three readings stand out. The cache budget halves the online cost (from 3.03 to 1.73 ms per query at K=100) without changing offline compression, which sits near 0.8 ms per video at both budgets and is essentially insensitive to M. Peak GPU stays under 1 GB throughout, so the online path runs comfortably on commodity inference hardware, and the offline cost is paid once per video and amortized across all future queries. Recall is stable across $K \in \{10, 20, 50, 100, 200, 500\}$ (App. A.7).

3.3. Q2: How Small Can the Cache Become?

Two knobs control cache size: tokens per video and bits per token. Table 3 sweeps tokens on MSR-VTT with the rest of the pipeline held fixed; the precision sweep at 16 tokens (FP8, FP4) is in Appendix A.8.

Token count dominates compression. One cached token per sampled frame (16 tokens per video) preserves T2V R@1

Table 3. Token-budget sweep on MSR-VTT 1ka for the full VEDJE model with $\mathcal{L}_{\text{delta}}$, BF16 cache. One token per sampled frame nearly preserves T2V R@1 while cutting cache size by roughly $4\times$.

Tokens/ video	Tokens/ frame	Cache/ video	T2V R@1	V2T R@1	Δ T2V vs. 64
64	4	49 kB	**54.6** ±0.1	**54.6** ±0.1	–
16	**1**	∼**12 kB**	**54.4** ±0.0	**53.1** ±0.0	**-0.2**

Table 4. Cache-structure ablation on MSR-VTT 1ka. All rows share the same candidate generator, stage-1 prior, 33M reranker, and 64-token BF16 cache.

Variant	Temporal Order	Future-Delta	T2V	V2T
EDJE middle frame	✗	✗	29.6 ±0.4	31.4 ±0.6
FrameSet-EDJE mean	✗	✗	51.7 ±0.2	51.6 ±0.2
FrameSet-EDJE attn.	✗	✗	52.0 ±0.2	51.6 ±0.2
VEDJE w/o $\mathcal{L}_{\text{delta}}$	✓	✗	54.0 ±0.1	53.6 ±0.1
Full VEDJE	✓	✓	**54.6** ±0.1	**54.6** ±0.1

within 0.2 (54.4 vs. 54.6) and shrinks BF16 cache from 49 to ∼12 kB; V2T R@1 drops asymmetrically from 54.6 to 53.1, so the 16-token point favours T2V. Precision is benign: FP8 leaves R@1 unchanged and FP4 loses 0.4 T2V R@1 (Appendix A.8). At one million videos, the 16-token BF16 setting therefore stores roughly 12 GB and FP4 roughly 3 GB, both small enough that the bottleneck at index scale is cache *design* rather than cache *capacity*.

3.4. Q3: Which Cache Structure Carries the Gain?

Cache size does not specify cache *structure*. Table 4 isolates this factor by holding the candidate pool, stage-1 prior, reranker, and 64-token budget fixed while changing how the cache is organized: image-only, unordered multi-frame, ordered frame-local, and ordered frame-local with future-delta supervision.

The ladder isolates the design choice. Single-frame image caching is not a usable video baseline; unordered multi-frame coverage rises to about 52/52; ordered frame-local tokens recover most of the remaining gap; and $\mathcal{L}_{\text{delta}}$ supplies the final lift to 54.6/54.6. Cache *structure* therefore carries most of the gain at fixed budget; loss-stack contributions are in Appendix A.13.

3.5. Q4: Where Does Future-Delta Supervision Help?

The cache-design view predicts that $\mathcal{L}_{\text{delta}}$ should matter most when caches are tight or videos are visually dynamic. Table 5 confirms the tight-cache case: at 16 cached tokens, $\mathcal{L}_{\text{delta}}$ adds +1.9 T2V R@1, while gains remain positive at 64 tokens.

Tight caches. At 16 cached tokens, $\mathcal{L}_{\text{delta}}$ adds +1.9 R@1 on MSR-VTT T2V, with a smaller V2T gain; at 64 tokens, gains stay positive in both directions. Tighter caches lean harder on the temporal signal $\mathcal{L}_{\text{delta}}$ preserves.

Table 5. Effect of $\mathcal{L}_{\text{delta}}$ under cache budgets. Values are R@1 percentage points.

Tok/ vid	T2V R@1			V2T R@1		
	w/o	with	Δ	w/o	with	Δ
16	52.5 ±0.0	**54.4** ±0.0	**+1.9**	52.6 ±0.0	**53.1** ±0.0	**+0.5**
64	54.0 ±0.1	**54.6** ±0.1	+0.6	53.6 ±0.1	**54.6** ±0.1	+1.0

Table 6. Stage-1 prior controls. Full VEDJE needs both cached evidence and the prior.

Variant	T2V R@1	V2T R@1
Stage 1 only	50.1 ±0.0	49.8 ±0.0
Joint only	43.8 ±0.3	41.6 ±0.3
Full VEDJE	**54.6** ±0.1	**54.6** ±0.1

High visual dynamics. If $\mathcal{L}_{\text{delta}}$ preserves temporal evidence, its benefit should concentrate in high-dynamics videos. Stratifying retrieval by frozen VideoPrism (Zhao et al., 2024) adjacent-frame change reveals exactly that: the high-dynamics T2V gain is +2.8 R@1, with no comparable low-dynamics or V2T counterpart (Table 12, Appendix A.10).

Cache probe. A query-directed MaxSim probe shows that $\mathcal{L}_{\text{delta}}$ increases verb and temporal-word alignment, and masking the selected cached tokens causes a much larger score drop with $\mathcal{L}_{\text{delta}}$ (Appendix A.11).

3.6. Q5: What Does the Stage-1 Prior Contribute?

The first-stage score $\rho(q, v)$ is not new visual evidence; any monotone function of ρ only re-scales rankings. Table 6 contrasts prior-only calibration, cached tokens alone, and the residual combination. Prior-only calibration matches stage 1 exactly, so no monotone function of ρ alone explains the lift. Joint tokens without the prior lose 6.3 T2V and 8.2 V2T R@1, because a small-pool reranker cannot reconstruct the ordering stage 1 already supplies. Cached evidence is therefore a residual correction around the prior, not a substitute for it.

4. Conclusion

VEDJE recasts second-stage video reranking as a cache-design problem: encode each video once into an ordered, frame-local cache, then let a 33M joint reranker read that cache with a residual stage-1 prior. The experiments show that cache structure dominates cache capacity, the stage-1 prior and cached evidence are non-redundant, and $\mathcal{L}_{\text{delta}}$ helps most when caches are tight or videos are dynamic. Across MSR-VTT, MSVD, DiDeMo, and ActivityNet, VEDJE lifts R@1 over its matched first-stage retriever while keeping reranking at ∼3 ms/query and under 1 GB of GPU memory.

Limitations and future work. VEDJE trades query-time compute for a query-blind cache, so query-adaptive reads or lightweight cache expansion are natural next steps. Finally,

its cache is tied to a frozen visual backbone, making re-indexing necessary after backbone updates, and the first and second stages are not yet trained jointly.

References

Bolya, D., Huang, P.-Y., Sun, P., Cho, J. H., Madotto, A., Wei, C., Ma, T., Zhi, J., Rajasegaran, J., Rasheed, H. A., et al. Perception encoder: The best visual embeddings are not at the output of the network. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/pdf?id=INqBOmwIpG.

Chen, D. and Dolan, W. Collecting highly parallel data for paraphrase evaluation. In Lin, D., Matsumoto, Y., and Mihalcea, R. (eds.), *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 190–200, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1020/.

Dai, Z., Cheng, K., Shao, F., Dong, Z., and Zhu, S. Text–video retrieval re-ranking via multi-grained cross attention and frozen image encoders. *Pattern Recognition*, 159:111099, 2025. doi: 10.1016/j.patcog.2024.111099. URL https://doi.org/10.1016/j.patcog.2024.111099.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, June 2019. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Hendricks, L. A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., and Russell, B. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 5804–5813, October 2017. doi: 10.1109/iccv.2017.618. URL https://doi.org/10.1109/iccv.2017.618.

Ko, D., Lee, J. S., Choi, M., Meng, Z., and Kim, H. J. Bidirectional likelihood estimation with multi-modal large language models for text-video retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22263–22273, October 2025. URL https://doi.org/10.1109/iccv51701.2025.02067.

Krishna, R., Hata, K., Ren, F., Fei-Fei, L., and Niebles, J. C. Dense-captioning events in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 706–715, October 2017. doi: 10.1109/iccv.2017.83. URL https://doi.org/10.1109/iccv.2017.83.

Lee, J. S., Ko, B., Cho, J., Lee, H., Byun, J., and Kim, H. J. Captioning for text-video retrieval via dual-group direct preference optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 16022–16039, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.869. URL https://aclanthology.org/2025.findings-emnlp.869/.

Li, J., Li, D., Xiong, C., and Hoi, S. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 12888–12900. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/li22n.html.

Li, J., Li, D., Savarese, S., and Hoi, S. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19730–19742. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/li23q.html.

Liu, Y., Zhang, Y., Cai, J., Jiang, X., Hu, Y., Yao, J., Wang, Y., and Xie, W. LamRA: Large multimodal model as your advanced retrieval assistant. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4015–4025, June 2025. doi: 10.1109/cvpr52734.2025.00380. URL https://doi.org/10.1109/cvpr52734.2025.00380.

Luo, H., Ji, L., Zhong, M., Chen, Y., Lei, W., Duan, N., and Li, T. CLIP4Clip: An empirical study of CLIP for end to end video clip retrieval and captioning. *Neurocomputing*, 508:293–304, 2022. doi: 10.1016/j.neucom.2022.07.028. URL https://doi.org/10.1016/j.neucom.2022.07.028.

Ma, Y., Xu, G., Sun, X., Yan, M., Zhang, J., and Ji, R. X-CLIP: End-to-end multi-grained contrastive learning for video-text retrieval. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 638–647, 2022. doi: 10.1145/3503161.3547910. URL https://doi.org/10.1145/3503161.3547910.

Reddy, A., Martin, A., Yang, E., Yates, A., Sanders, K., Murray, K., Kriz, R., de Melo, C. M., Van Durme, B., and Chellappa, R. Video-ColBERT: Contextualized late interaction for text-to-video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19691–19701, June 2025. doi: 10.1109/cvpr52734.2025.01834. URL https://doi.org/10.1109/cvpr52734.2025.01834.

Taraday, M. K., Wagner, S., and Baskin, C. Efficient discriminative joint encoders for large scale vision-language reranking. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/pdf?id=UXtTBAyqVB.

Tian, K., Zhao, R., Xin, Z., Lan, B., and Li, X. Holistic features are almost sufficient for text-to-video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17138–17147, June 2024. doi: 10.1109/cvpr52733.2024.01622. URL https://doi.org/10.1109/cvpr52733.2024.01622.

Wang, J., Wang, C., Huang, K., Huang, J., and Jin, L. VideoCLIP-XL: Advancing long description understanding for video CLIP models. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16061–16075, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.898. URL https://aclanthology.org/2024.emnlp-main.898/.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, Y., Li, K., Li, X., Yu, J., He, Y., Chen, G., Pei, B., Zheng, R., Wang, Z., Shi, Y., Jiang, T., Li, S., Xu, J., Zhang, H., Huang, Y., Qiao, Y., Wang, Y., and Wang, L. InternVideo2: Scaling foundation models for multimodal video understanding. In *Computer Vision – ECCV 2024*, pp. 396–416. Springer Nature Switzerland, 2024b. doi: 10.1007/978-3-031-73013-9_23. URL https://doi.org/10.1007/978-3-031-73013-9_23.

Xu, J., Mei, T., Yao, T., and Rui, Y. MSR-VTT: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5288–5296, June 2016. doi: 10.1109/cvpr.2016.571. URL https://doi.org/10.1109/cvpr.2016.571.

Zhao, L., Gundavarapu, N. B., Yuan, L., Zhou, H., Yan, S., Sun, J. J., Friedman, L., Qian, R., Weyand, T., Zhao, Y., Hornung, R., Schroff, F., Yang, M.-H., Ross, D. A., Wang, H., Adam, H., Sirotenko, M., Liu, T., and Gong, B. VideoPrism: A foundational visual encoder for video understanding. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 60785–60811. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhao24f.html.

A. Additional Experimental Details

This appendix documents the protocol behind the cached-reranking results and reports the secondary controls that complement the main experiments. Sections A.1-A.5 fix the deployment-relevant details that remain in the appendix (algorithm, architecture, training, cache quantization, compute, and code release); Sections A.6-A.15 provide the higher-rank cutoffs, the token-cache probe, the loss-stack ablation, the future-delta horizon sweep, and the reranker-backbone control.

Recall@K is reported in percentage points throughout, rounded to one decimal. The matched-reranking tables report the stage-1 R@1 and the reranked R@1 in the same row whenever both numbers are available, so the gain columns measure changes over the same candidate generator rather than comparisons across retrieval systems. Body tables that contain $\pm$std entries (Tables 1, 3, 4, 5, and 6) report mean R@1 with the 1-σ standard deviation across 3 training seeds; appendix tables without $\pm$std are single-run results unless their caption states otherwise. The token-cache probe in Section A.11 is a diagnostic for cached-token alignment and score faithfulness, not a claim of ground-truth temporal localization.

A.1. VEDJE Indexing and Reranking Algorithm

Algorithm 1 writes out the offline indexing and online reranking steps used in Section 2. Indexing runs g_ϕ and the frame-local compressor once per video and writes $Z(v)$ to disk. Reranking runs the query text encoder, the joint encoder, and the prior embedding e_ρ over the candidate list returned by the first-stage retriever; the visual encoder g_ϕ and the future-delta predictor F_ω never run at query time.

Algorithm 1 VEDJE: offline indexing and online reranking.

Require: frozen g_ϕ; trained compressor, joint encoder, prior embedding e_ρ; first-stage retriever
Offline indexing (per video v):
sample frames $\{f_t\}_{t=1}^{T}$ and compute $X_t = g_\phi(f_t)$ {visual backbone, once per video}
$Z_t \leftarrow \mathrm{FFN}(\mathrm{MHA}(Q_0,\, X_tW_K,\, X_tW_V))$ {Eq. 1}
store ordered cache $Z(v) = [Z_1; \ldots; Z_T]$ {Eq. 2}
Online reranking (per query q):
$\mathcal{C}(q),\ \rho(\cdot, q) \leftarrow$ first-stage retriever over the index
for each candidate $v \in \mathcal{C}(q)$ **do**
  $c(q, v) \leftarrow$ joint encoder over $[\mathrm{tokens}(q);\ Z(v)]$
  $s_\theta(q, v) \leftarrow w^\top\big(c(q, v) + e_\rho(\rho(q, v))\big) + b$ {Eq. 5}
end for
return $\mathcal{C}(q)$ sorted by s_θ {no g_ϕ on the query path}

A.2. Architecture Details

This subsection fixes every component referenced in Section 2 to a concrete operator with explicit dimensions; symbol use matches the main text.

Visual backbone g_ϕ. VideoPrism-Base (Zhao et al., 2024), used at 288×288 resolution with 16 frames per video. The HuggingFace checkpoint is `MHRDYN7/videoprism-base-f16r288`. The backbone is frozen at every stage: gradients do not flow into its weights, and it runs once per video at indexing time. Each frame produces P=256 patch tokens with vision dimension d_v=768, so $X_t \in \mathbb{R}^{P\times d_v}$.

Frame-local compressor. A 2-layer transformer-decoder stack operates in the joint-encoder dimension d_ℓ=384. For each frame, M learned query tokens $Q_0 \in \mathbb{R}^{M\times d_\ell}$ cross-attend to projected patches $K_t, V_t = X_tW_K, X_tW_V$ via a single shared linear projection W_K=$W_V \in \mathbb{R}^{d_v\times d_\ell}$ (`kv_proj`). The two decoder layers use 8 attention heads, FFN width $4d_\ell$=1536, GELU activation, and dropout 0.1. The output of the second layer is the per-frame cache $Z_t \in \mathbb{R}^{M\times d_\ell}$. Concatenating frames in temporal order yields $Z(v)$=$[Z_1; \ldots; Z_T] \in \mathbb{R}^{TM\times d_\ell}$. The default TM=64 uses T=16, M=4. Compact configurations reduce M while keeping T=16 (e.g., TM=16 uses M=1).

Joint reranker (online). `microsoft/MiniLM-L12-H384-uncased` (Wang et al., 2020): 12 transformer blocks, hidden size d_ℓ=384, 12 attention heads, ∼33M parameters, with the standard BERT (Devlin et al., 2019) uncased WordPiece

tokenizer (max text length 64 tokens, including `[CLS]` and `[SEP]`). Cached video tokens $Z(v)$ are concatenated with the text input embeddings as additional input positions.

Score-encoder e_ρ. The scalar stage-1 score $\rho(q, v)$ is mapped to $\mathbb{R}^{d_\ell}$ by a 2-layer MLP $\mathrm{Linear}(1, 64) \to \mathrm{GELU} \to \mathrm{Linear}(64, d_\ell)$.

Score head. The compact form $s_\theta(q, v)=w^\top \tilde{c}(q, v)+b$ in Eq. 5 is implemented in code as a 2-layer MLP $\mathrm{Linear}(d_\ell, d_\ell) \to \mathrm{GELU} \to \mathrm{Linear}(d_\ell, 1)$. The nonlinearity is absorbed into w in the main paper for narrative compactness; this choice is parameter-light ($\sim$0.15M extra parameters) and does not change the deployment-cost claims.

Future-delta predictor F_ω (training only). A 2-layer transformer-decoder stack with 8 attention heads, FFN width $4d_\ell$=1536, GELU activation, dropout 0.1, and LayerNorm on the output. The default horizon set is the single horizon $\mathcal{H}=\{3\}$ (Section A.14 reports $h \in \{2, 3, 4\}$). Queries are *fixed* sinusoidal vectors of dimension d_ℓ indexed by horizon and patch position (h, p) with $h \in \mathcal{H}$ and $p \in \{0, \ldots, P-1\}$ (P=256); the first half of d_ℓ encodes h and the second half encodes p via standard sinusoidal positional encoding. The decoder cross-attends to a single frame's compressed tokens $Z_t \in \mathbb{R}^{M \times d_\ell}$, and a final linear layer projects each output to $\mathbb{R}^{d_v}=\mathbb{R}^{768}$ to produce $\widehat{\Delta}_t \in \mathbb{R}^{|\mathcal{H}| \times P \times d_v}$. The predictor is discarded at deployment, so its parameter count does not affect the 33M online cost.

Cache layout and storage. The per-video cache $Z(v)$ is stored as an $\mathrm{TM} \times d_\ell$ tensor with no auxiliary metadata. At BF16 precision the on-disk size is $\mathrm{TM} \cdot d_\ell \cdot 2$ bytes.

A.3. Training Protocol

Training is a single stage that jointly optimizes the four objectives in Eq. 6: the matching loss $\mathcal{L}_{\mathrm{vtm}}$, the contrastive loss $\mathcal{L}_{\mathrm{vtc}}$, the masked-language loss $\mathcal{L}_{\mathrm{mlm}}$, and the future-delta auxiliary $\mathcal{L}_{\mathrm{delta}}$.

Trainable and frozen parameters. The visual backbone g_ϕ is frozen throughout: gradients do not flow into its weights, and it is never re-run after indexing. The frame-local compressor (Eq. 1), the joint encoder, the score-encoder MLP e_ρ (Eq. 5), the score head, the VTC text-projection head, the MLM head, and the future-delta predictor F_ω are jointly trained. After training, the future-delta predictor and the auxiliary heads are discarded. The remaining trained modules sit on different sides of the indexing/query split: the frame-local compressor runs once per video at indexing time and writes the cache $Z(v)$ to disk, while only the joint encoder, e_ρ, and the score head run online at query time on the cached tokens.

Optimization.

- Optimizer: AdamW with default β=$(0.9, 0.999)$ and weight decay 0.02.
- Schedule: 400 warmup steps from 10^{-6} to 3×10^{-4}, then per-epoch step decay with multiplicative factor 0.9.
- Batch size: 64 query-video pairs per optimizer step.
- Epochs: 4.

Loss definitions used in training.

- *VTM* ($\mathcal{L}_{\mathrm{vtm}}$): softmax cross-entropy over the candidate scores $s_\theta(q, v_i)$ at temperature τ_{vtm}=1.0, with the positive as the target.
- *VTC* ($\mathcal{L}_{\mathrm{vtc}}$): symmetric InfoNCE between the text CLS embedding (linearly projected to d_{clip}=768 and L2-normalized) and the L2-normalized VideoPrism-LvT-B (Zhao et al., 2024) video features, at logit scale 20.0.
- *MLM* ($\mathcal{L}_{\mathrm{mlm}}$): standard masked-language modeling on the text tokens, computed jointly with the cached video tokens in context (i.e., the encoder reads both modalities).
- $\mathcal{L}_{\mathrm{delta}}$: squared ℓ_2 regression as in Eq. 4, mean-squared error between the predicted patch-level delta $\widehat{\Delta}_{t,h}$ and the frozen-backbone target $\Delta_{t,h}=X_{t+h}-X_t$, averaged over patches and the visual feature dimension and then over the set of valid future pairs $\Omega = \{(t, h) : 1 \le t,\ t+h \le T,\ h \in \mathcal{H}\}$. The default horizon set is the single horizon $\mathcal{H}=\{3\}$; Section A.14 reports $h \in \{2, 3, 4\}$.

Table 7. Full text-to-video retrieval accuracy, with Recall@K in percentage points. VEDJE variant subscripts identify the first-stage candidate generator; stage-1-only rows are omitted here and shown in Table 9. PE cross-dataset rows are omitted because the underlying stage-1 features for those datasets are not available in this release.

	MSR-VTT			MSVD			DiDeMo			ActivityNet		
Method	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**
Reported retrieval baselines												
CLIP4Clip (Luo et al., 2022)	44.5	71.4	81.6	45.2	75.5	–	42.8	68.5	79.2	40.5	72.4	–
X-CLIP (Ma et al., 2022)	49.3	75.8	84.8	50.4	80.6	–	47.8	79.3	–	46.2	75.5	–
TeachCLIP (Tian et al., 2024)	45.2	72.3	82.3	47.4	77.3	85.5	–	–	–	42.2	72.7	85.2
Video-ColBERT (Reddy et al., 2025)	51.5	76.3	85.5	55.2	82.9	89.4	51.7	76.1	84.8	45.8	76.3	86.7
Online rerankers and MLLM scorers												
CrossTVR-Large (Dai et al., 2025)	54.0	77.5	85.3	55.0	81.9	–	55.0	77.6	–	–	–	–
LamRA (Liu et al., 2025)	59.7	81.4	87.2	59.0	–	–	83.5	94.8	96.2	76.0	92.8	96.3
CaRe-DPO (Lee et al., 2025)	64.1	83.8	88.8	59.8	–	–	85.1	95.0	96.2	79.2	93.6	96.5
Cached joint reranking												
VEDJE$_{\mathbf{VP}}$ (Zhao et al., 2024)	**54.6**	**76.9**	**85.4**	56.7	83.7	89.8	48.2	72.9	80.3	50.6	77.8	87.7
VEDJE$_{\mathbf{VCLIP}}$ (Wang et al., 2024a)	56.5	81.8	88.0	56.8	83.0	90.1	56.6	79.8	86.8	51.6	79.9	89.3
VEDJE$_{\mathbf{PE}}$ (Bolya et al., 2025)	53.9	77.3	85.0	–	–	–	–	–	–	–	–	–

A.4. Cache Quantization

The default cache precision is BF16. The FP8-E4M3 and FP4-E2M1 rows in Section 3.3 and Table 10 are produced by quantizing the trained, written-out cache tensors and dequantizing back to FP32 before the joint encoder reads them. Reranker weights and the score-encoder e_ρ remain at training precision; only the cache pays the precision cost. The kB-per-video figures are the on-disk sizes of the quantized cache (FP4 stores half a byte per element, BF16 two bytes per element, both before any container overhead).

A.5. Compute and Code Release

Hardware. Training and indexing run on a single 48 GB NVIDIA L40S GPU. A single training run takes roughly 16-24 wall-clock hours at the default schedule (4 epochs, batch size 64); the full set of trained configurations reported in the paper (the seed reruns of Tables 1, 3, 4, 5, and 6 together with the appendix loss-stack, horizon-sweep, reranker-backbone, and absolute-feature ablations) total approximately 850-900 GPU-hours, i.e. ∼35-40 L40S-days; this figure does not include preliminary or failed experiments that did not make it into the paper.

Code release. An anonymized snapshot of the code is included in the supplementary material.

License and asset notes. The pretrained VideoPrism-Base and VideoPrism-LvT-B (Zhao et al., 2024) weights are released by Google DeepMind under the Apache License 2.0 (`https://github.com/google-deepmind/videoprism`) and accessed through the public HuggingFace mirrors `MHRDYN7/videoprism-base-f16r288` and `MHRDYN7/videoprism-lvt-base-f16r288`, which carry no additional license declaration of their own. The MiniLM (Wang et al., 2020) checkpoint `microsoft/MiniLM-L12-H384-uncased` is distributed under the MIT license. MSR-VTT, MSVD, DiDeMo, and ActivityNet are used under their original release terms.

A.6. Full Retrieval Results

Tables 7 and 8 report Recall@K for text-to-video and video-to-text retrieval separately, in percentage points. The variant convention follows the main text: VEDJE variants are written VEDJE$_{\mathrm{S1}}$-N, with the subscript naming the first-stage candidate generator and N naming cached tokens (omitted in print when N=64). Parenthesized values next to VEDJE R@1 entries denote the absolute improvement over the matched stage-1 retriever. Table 9 then isolates the matched first-stage comparison and reports the same gain explicitly.

A.7. Sensitivity to the Candidate-Pool Size K

The default candidate pool K=20 used elsewhere in the paper matches the smaller MSR-VTT 1ka test set. We sweep $K \in \{10, 20, 50, 100, 200, 500\}$ on MSR-VTT to check that this choice is not load-bearing, fixing the first-stage retriever to

Table 8. Full video-to-text retrieval accuracy, with Recall@K in percentage points. VEDJE variant subscripts identify the first-stage candidate generator; stage-1-only rows are omitted here and shown in Table 9. PE cross-dataset rows are omitted because the underlying stage-1 features for those datasets are not available in this release.

	MSR-VTT			MSVD			DiDeMo			ActivityNet		
Method	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**
Reported retrieval baselines												
CLIP4Clip (Luo et al., 2022)	43.1	70.5	81.2	62.0	87.3	–	42.5	70.6	80.2	42.6	73.4	–
X-CLIP (Ma et al., 2022)	48.9	76.8	84.5	66.8	90.4	–	47.8	76.8	–	46.4	75.9	–
TeachCLIP (Tian et al., 2024)	–	–	–	–	–	–	–	–	–	–	–	–
Video-ColBERT (Reddy et al., 2025)	–	–	–	–	–	–	–	–	–	–	–	–
Online rerankers and MLLM scorers												
CrossTVR-Large (Dai et al., 2025)	51.3	78.3	85.4	74.3	95.5	–	51.3	76.7	–	–	–	–
LamRA (Liu et al., 2025)	60.7	82.3	89.0	85.5	–	–	79.4	94.8	96.6	68.7	90.1	95.3
CaRe-DPO (Lee et al., 2025)	63.8	83.0	87.3	85.9	–	–	82.5	95.2	96.3	74.4	92.4	96.3
Cached joint reranking												
VEDJE$_{\mathbf{VP}}$ (Zhao et al., 2024)	**54.6**	**79.3**	**86.7**	83.7	96.0	98.7	47.9	73.5	81.5	48.3	76.8	86.6
VEDJE$_{\mathbf{VCLIP}}$ (Wang et al., 2024a)	57.1	81.6	88.7	76.9	93.8	96.4	54.8	80.8	86.4	49.9	79.1	89.5
VEDJE$_{\mathbf{PE}}$ (Bolya et al., 2025)	53.9	78.2	85.9	–	–	–	–	–	–	–	–	–

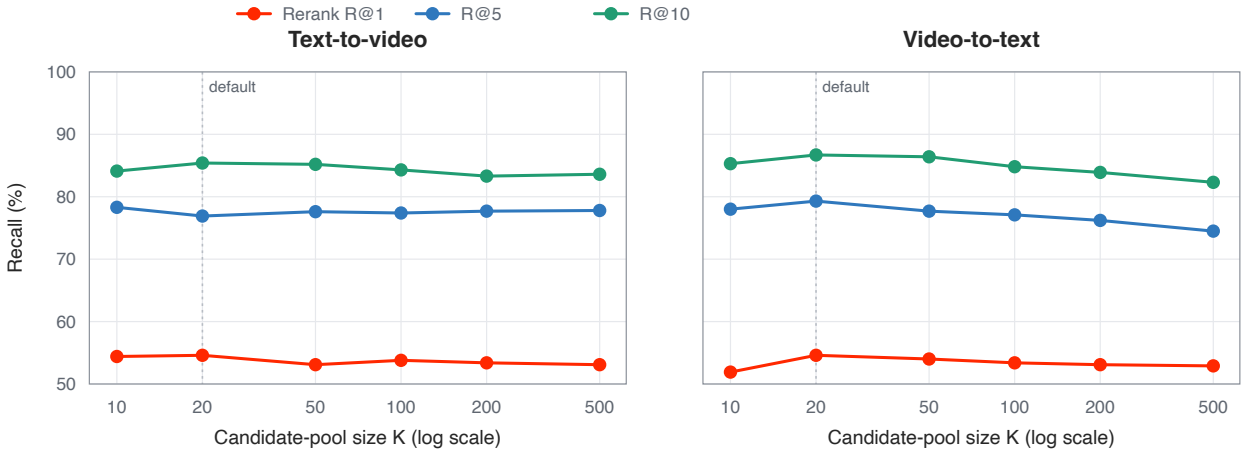

Figure 3. MSR-VTT 1ka candidate-pool sensitivity of VEDJE$_{\mathrm{VP}}$ (Zhao et al., 2024). Curves report reranked recall; the vertical guide marks the default $K{=}20$.

VideoPrism (Zhao et al., 2024) and the reranker to the default VEDJE configuration. Reranked recall is stable across the entire range (Figure 3): R@1, R@5, and R@10 are essentially flat in both retrieval directions, with no saturation at small K (additional candidates would still help) and no dilution at large K (irrelevant candidates do not crowd out the relevant ones).

A.8. 16-Token Precision Details

Table 10 reports Recall@K for the 16-token precision rows in Section 3.3, with all values in percentage points.

The higher-rank cutoffs reinforce the R@1 reading. FP8 leaves T2V and V2T values effectively unchanged from the BF16 reference, and FP4 keeps R@5 and R@10 within 0.1 percentage point on T2V while matching or slightly improving the V2T R@1 reference. At this cache budget, precision is therefore not the limiting factor.

A.9. 16-Token Future-Delta Details

Table 11 gives Recall@K for the MSR-VTT 16-token future-delta comparison from Section 3.5, in percentage points. At one cached token per sampled frame, $\mathcal{L}_{\mathrm{delta}}$ improves every reported cutoff in both retrieval directions, with the largest gain at T2V R@1.

The 16-token results extend the R@1 claim in the main paper to higher cutoffs. The top-rank T2V gain is $+1.9$ percentage

Table 9. Matched first-stage and cached-reranker retrieval, with Recall@K and gains in percentage points. Each panel fixes the candidate generator and the reranker; the VEDJE subscript names that fixed stage-1 generator. "Gain" is the absolute R@1 change from the stage-1 retriever to the matched cached reranker.

Dataset	Dir.	Stage-1 R@1	Reranked R@1	Gain	Reranked R@5	Reranked R@10
VideoPrism-LvT-B stage 1 → VEDJE$_{\mathrm{VP}}$ (Zhao et al., 2024)						
MSR-VTT	T2V	50.1	54.6	+4.5	76.9	85.4
MSR-VTT	V2T	49.8	54.6	+4.8	79.3	86.7
MSVD	T2V	55.6	56.7	+1.1	83.7	89.8
MSVD	V2T	83.4	83.7	+0.3	96.0	98.7
DiDeMo	T2V	47.1	48.2	+1.1	72.9	80.3
DiDeMo	V2T	47.3	47.9	+0.6	73.5	81.5
ActivityNet	T2V	48.8	50.6	+1.8	77.8	87.7
ActivityNet	V2T	47.9	48.3	+0.4	76.8	86.6
PE-Core-B stage 1 → VEDJE$_{\mathrm{PE}}$ (Bolya et al., 2025)						
MSR-VTT	T2V	47.6	53.9	+6.3	77.3	85.0
MSR-VTT	V2T	47.3	53.9	+6.6	78.2	85.9
VideoCLIP-XL stage 1 → VEDJE$_{\mathrm{VCLIP}}$ (Wang et al., 2024a)						
MSR-VTT	T2V	50.1	56.5	+6.4	81.8	88.0
MSR-VTT	V2T	49.9	57.1	+7.2	81.6	88.7
MSVD	T2V	51.9	56.8	+4.9	83.0	90.1
MSVD	V2T	76.7	76.9	+0.2	93.8	96.4
DiDeMo	T2V	47.7	56.6	+8.9	79.8	86.8
DiDeMo	V2T	47.9	54.8	+6.9	80.8	86.4
ActivityNet	T2V	46.4	51.6	+5.2	79.9	89.3
ActivityNet	V2T	48.1	49.9	+1.8	79.1	89.5

Table 10. MSR-VTT 1ka Recall@K for 16 cached tokens under different cache precisions, in percentage points. All rows use one cached token per sampled frame.

Precision	T2V R@1	T2V R@5	T2V R@10	V2T R@1	V2T R@5	V2T R@10
BF16 reference	**54.4**	**78.9**	**85.6**	53.1	**78.4**	**86.5**
FP8-E4M3	**54.4**	**78.9**	85.5	**53.2**	**78.4**	**86.5**
FP4-E2M1	54.0	78.8	85.5	**53.2**	**78.4**	**86.5**

points; improvements persist at $+0.8$ R@5 and $+1.1$ R@10. V2T gains are smaller but consistently positive, from $+0.5$ R@1 to $+0.8$ R@10.

A.10. Dynamics Stratification Protocol

The dynamics-stratified analysis in Section 3.5 (Table 12) groups test-split videos into Low, Medium, and High visual-dynamics bins. The bin labels are derived from the frozen VideoPrism (Zhao et al., 2024) backbone shared with the first-stage retriever, computed once per dataset on the test split. They never enter training, do not affect the trained VEDJE model or the cached tokens, and are read only at evaluation time by the analysis in Table 12.

Per-frame feature. For each test-split video, the same 16 uniformly sampled frames used at indexing and evaluation (Section A.2) are passed through the frozen VideoPrism (Zhao et al., 2024) backbone. The per-frame feature $u_t \in \mathbb{R}^{d_{\mathrm{clip}}}$ is the mean-pooled patch tokens at frame t, L2-normalized, taken from the same forward pass that produces the stage-1 retriever's video embedding.

Adjacent-frame change. Adjacent-frame change is the cosine distance between consecutive frame features,

$$d_t(v) \;=\; 1 \;-\; u_t^{\top} u_{t+1}, \qquad t = 1, \ldots, T-1, \tag{7}$$

with T=16. The per-video dynamics scalar is the mean over the $T-1 = 15$ adjacent pairs,

$$D(v) \;=\; \frac{1}{T-1} \sum_{t=1}^{T-1} d_t(v). \tag{8}$$

Table 11. MSR-VTT 1ka Recall@K for 16 cached tokens with and without $\mathcal{L}_{\text{delta}}$, in percentage points. All rows use one cached token per sampled frame.

Variant	T2V R@1	T2V R@5	T2V R@10	V2T R@1	V2T R@5	V2T R@10
16 tokens w/o $\mathcal{L}_{\text{delta}}$	52.5	78.1	84.5	52.6	77.8	85.7
16 tokens with $\mathcal{L}_{\text{delta}}$	**54.4**	**78.9**	**85.6**	**53.1**	**78.4**	**86.5**

Table 12. Dynamics-stratified effect of $\mathcal{L}_{\text{delta}}$. Recall and deltas (full − w/o $\mathcal{L}_{\text{delta}}$) are R@1 percentage points; bins use frozen VideoPrism (Zhao et al., 2024) adjacent-frame change. The largest gains concentrate in high-dynamics T2V on both datasets; low-dynamics rows are mixed in sign.

Dataset	Dir.	Low dynamics			Medium dynamics			High dynamics		
		w/o	Full	Δ	w/o	Full	Δ	w/o	Full	Δ
MSR-VTT	T2V	56.8	56.4	-0.4	53.2	53.8	+0.6	52.4	**55.2**	**+2.8**
MSR-VTT	V2T	54.8	58.4	+3.6	54.4	53.8	-0.6	51.2	52.0	+0.8
ActivityNet	T2V	43.4	42.8	-0.6	49.3	50.1	+0.8	48.9	**51.7**	**+2.8**
ActivityNet	V2T	41.7	42.6	+1.0	46.3	46.5	+0.2	44.3	45.9	+1.5

A larger $D(v)$ means that consecutive frames sit further apart in the frozen feature space, which is consistent with larger visible motion or scene change.

Bin assignment. Bins are computed once per dataset on the corresponding test split. Let $D_{25}^{(\mathcal{D})}$ and $D_{75}^{(\mathcal{D})}$ denote the 25th and 75th percentiles of $\{D(v)\}_{v\in\mathcal{D}_{\text{test}}}$ within dataset $\mathcal{D}$. Each video is then assigned to one of three bins,

$$\text{bin}(v) = \begin{cases} \text{Low} & \text{if } D(v) \le D_{25}^{(\mathcal{D})}, \\ \text{High} & \text{if } D(v) \ge D_{75}^{(\mathcal{D})}, \\ \text{Medium} & \text{otherwise.} \end{cases} \tag{9}$$

The Low and High bins therefore correspond to the bottom and top quartiles of the per-dataset dynamics distribution and the Medium bin to the middle 50%; the Low/High contrast in Table 12 is a comparison between the tails rather than a partition into thirds. Bins are recomputed independently for MSR-VTT and ActivityNet because the absolute scale of $D(v)$ depends on dataset-specific scene-change statistics, so a single global threshold would not be comparable across datasets.

What the bin labels apply to. Each (caption, video) test pair inherits the bin of its associated video. For T2V retrieval, R@1 within a bin is computed over the captions whose ground-truth video sits in that bin. For V2T retrieval, R@1 within a bin is computed over the query videos whose own bin label matches. The bins therefore stratify the same retrieval evaluation by the dynamics of the video evidence on either side of the query/gallery.

Stratified results. Table 12 reports the full Low / Medium / High split for both directions on MSR-VTT and ActivityNet. The body of the paper quotes the high-dynamics T2V row directly; the table also records that gains do not concentrate in the V2T direction and that low-dynamics rows are mixed in sign, consistent with $\mathcal{L}_{\text{delta}}$ acting on cues that only matter when adjacent frames change.

A.11. MSR-VTT Token-Cache Probe

The main experiments show *where* $\mathcal{L}_{\text{delta}}$ helps retrieval. This probe asks *what* changes inside the cached representation. The construction follows Video-ColBERT's query-directed MaxSim (Reddy et al., 2025): each query token selects its most relevant video evidence, without penalizing unrelated frames.

For each MSR-VTT query-video pair, let $J(q)$ be the set of content query tokens after removing special tokens, punctuation, and stopwords. Let e_j denote the normalized embedding of query token $j \in J(q)$ in the reranker text space, and let $z_{t,m}$ denote the normalized cached token for sampled frame t and within-frame slot m. The query-directed MaxSim alignment for token j is

$$a_j(q,v) = \max_{t,m} e_j^\top z_{t,m}, \qquad (t_j^\star, m_j^\star) = \arg\max_{t,m} e_j^\top z_{t,m}, \tag{10}$$

Table 13. MSR-VTT 1ka token-cache probe for 64 cached tokens. Alignment is the query-token MaxSim cosine similarity to cached frame-local tokens of the ground-truth video. Masking reports the reranker-score drop after replacing selected cached tokens; larger drops mean that the selected tokens are more important to the reranker score.

Probe	**64 w/o $\mathcal{L}_{\text{delta}}$**	**64 with $\mathcal{L}_{\text{delta}}$**
Mean token-cache alignment	0.1549	**0.1722**
Verb alignment	0.1397	**0.1582**
Temporal-word alignment	0.0977	**0.1299**
Top-token masking score drop	0.0236	**0.1790**
Random masking score drop	-0.0091	0.0598
Low-attribution masking score drop	-0.0212	0.0413

and the pair-level alignment averages over content tokens,

$$A(q, v) = \frac{1}{|J(q)|} \sum_{j \in J(q)} a_j(q, v). \tag{11}$$

$A(q, v)$ is reported for the ground-truth video on MSR-VTT 1ka, both as a mean over all content tokens and split by coarse lexical type. The verb and temporal-word splits are the most relevant ones for $\mathcal{L}_{\text{delta}}$, since the auxiliary target predicts future feature changes rather than static appearance.

The alignment rows in Table 13 show that $\mathcal{L}_{\text{delta}}$ raises the cosine similarity between query tokens and cached frame-local tokens. The relative gain is largest for temporal words, and the verb gain exceeds the mean-token gain. The pattern matches the intended role of the auxiliary loss: future-delta supervision biases the cache toward action and change rather than uniformly raising similarities.

Alignment alone does not establish that the cached tokens carry the joint encoder's evidence: a token can sit close to a query embedding without changing the reranker's score. The masking probe addresses this. Throughout the probe, the query text, the first-stage prior $\rho(q, v)$, and the visual backbone g_ϕ stay frozen; only the cached visual tokens entering the joint encoder change. The cache is a $T \times M$ grid: T=16 sampled frames, M=4 slots per frame, TM=64 tokens in total. Each cached token carries a query-support score

$$r_{t,m}(q, v) = \max_{j \in J(q)} e_j^\top z_{t,m}, \tag{12}$$

the cosine similarity to its closest content query token. Three masks then zero K=4 cached positions: the *top-token* mask picks the four positions with the largest $r_{t,m}$; the *low-attribution* mask picks the four with the smallest; the *random* mask picks four positions under a deterministic seed keyed by the query-video pair. Let $\tilde{Z}$ denote the resulting masked cache. The reported score drop is

$$\Delta s(q, v) = s_\theta(q, v, Z, \rho(q, v)) - s_\theta(q, v, \tilde{Z}, \rho(q, v)). \tag{13}$$

Negative values indicate that masking slightly raised the score, which can occur for irrelevant or noisy cached tokens. Within a single model, the top-token mask drops the reranker score substantially more than the random or low-attribution masks, and the gap widens with $\mathcal{L}_{\text{delta}}$ (0.179 versus 0.024). The interpretation is faithfulness: the MaxSim-selected positions are not nearest neighbors in a detached embedding space; they are the cached evidence that the reranker actually uses.

A.12. Future-Delta Loss Variant: Absolute Future Features

The default $\mathcal{L}_{\text{delta}}$ regresses the patch-level *difference* $\Delta_{t,h}$=X_{t+h}−X_t. As an ablation we replace this target with the absolute future feature X_{t+h} itself, holding everything else fixed (predictor architecture, horizon set, loss weight, training schedule). The intent is to check whether the gain from $\mathcal{L}_{\text{delta}}$ comes specifically from the change-prediction structure or from any auxiliary that ties cached tokens to the frozen visual feature space.

The delta target gives a higher T2V and V2T R@1 than the absolute-future-feature target ($+1.8$ T2V, $+0.8$ V2T), with the higher-rank cutoffs within a narrow band of each other. This is consistent with the design argument in Section 3.5: an absolute target can be matched by reconstructing static content shared between X_t and X_{t+h}, leaving action cues optional, whereas the delta target rewards exactly what changes between t and $t + h$. The gap is small enough that the auxiliary itself, not the specific delta form, supplies most of the gain at higher K; the delta form is the choice that pulls R@1 in particular.

Table 14. Future-delta vs. absolute-future-feature targets on MSR-VTT 1ka. Both rows use the same predictor and training configuration; only the regression target changes. Absolute-feature rows are reported with target features L2-normalized into $[0, 1]$. Recall values in percentage points.

Target	**T2V R@1**	**T2V R@5**	**T2V R@10**	**V2T R@1**	**V2T R@5**	**V2T R@10**
Absolute future feature X_{t+h} (normalized)	52.8	**78.2**	85.3	53.8	78.6	86.5
Future delta $\Delta_{t,h} = X_{t+h} - X_t$ (default)	**54.6**	76.9	**85.4**	**54.6**	**79.3**	**86.7**

Table 15. VideoPrism (Zhao et al., 2024) loss-stack ablation on MSR-VTT 1ka, with Recall@K in percentage points. The stage-1 row reports the VideoPrism retriever before reranking. Reranker rows add the training objectives in stages; the full VEDJE row adds $\mathcal{L}_{\text{delta}}$ on top of VTM, MLM, and VTC.

Variant	**Active training objectives**	**T2V R@1**	**T2V R@5**	**T2V R@10**	**V2T R@1**	**V2T R@5**	**V2T R@10**
Stage-1 only (Zhao et al., 2024)	none; VideoPrism retriever only	50.1	–	–	49.8	–	–
VTM	$\mathcal{L}_{\text{vtm}}$	52.8	76.9	84.0	50.6	77.4	85.2
VTM+MLM	$\mathcal{L}_{\text{vtm}} + \mathcal{L}_{\text{mlm}}$	52.7	76.2	84.2	51.4	76.0	84.9
VTM+MLM+VTC	$\mathcal{L}_{\text{vtm}} + \mathcal{L}_{\text{mlm}} + \mathcal{L}_{\text{vtc}}$	54.0	**77.8**	85.3	53.6	77.9	**87.0**
Full VEDJE	$\mathcal{L}_{\text{vtm}} + \mathcal{L}_{\text{mlm}} + \mathcal{L}_{\text{vtc}} + \mathcal{L}_{\text{delta}}$	**54.6**	76.9	**85.4**	**54.6**	**79.3**	86.7

A.13. Training-Loss Stack Ablation

Table 15 attributes the cached-reranking gain to each training objective. The stage-1 row reports the unreordered VideoPrism (Zhao et al., 2024) retriever. The remaining rows use the same VideoPrism (Zhao et al., 2024) candidate generator and cached joint-reranking setup while turning the training losses on in stages.

The R@1 trajectory is monotone even though some higher-rank cutoffs are not. VTM supplies most of the initial reranking gain over the VideoPrism (Zhao et al., 2024) prior ($+2.7$ T2V R@1 percentage points relative to stage-1 only); MLM mainly helps V2T R@1; VTC is the largest step for T2V R@1 ($+1.3$ percentage points); and $\mathcal{L}_{\text{delta}}$ adds a final $+0.6$ T2V and $+1.0$ V2T R@1 percentage points. The total decomposes the $+4.5$ T2V and $+4.8$ V2T gains in Table 1 into the contributions of the four objectives.

A.14. Future-Delta Loss Horizon Sweep

The horizon sweep tests whether $\mathcal{L}_{\text{delta}}$ depends on a single prediction horizon. R@K varies within a narrow band across $h \in \{2, 3, 4\}$, and h=3 achieves the best R@1 in both retrieval directions.

No horizon dominates every R@K cutoff, but the sweep rules out a brittle objective that works only at one offset. The default h=3 setting is selected because it achieves the best R@1 in both retrieval directions; neighboring horizons remain within a narrow band. The result supports a modest interpretation: $\mathcal{L}_{\text{delta}}$ acts as a short-range temporal-change bias rather than a single-horizon tuning trick.

A.15. Reranker Text-Encoder Control

This control isolates the source of the cached-reranking gain: cache structure versus reranker capacity. Replacing the default 33M joint encoder with BERT-base (Devlin et al., 2019) (110M parameters) adds 77M online parameters, a $3.3\times$ increase, but yields small mixed changes on MSR-VTT 1ka.

The larger text encoder does not explain the cached-reranking gain. BERT-base (Devlin et al., 2019) improves T2V R@5 and R@10 by 0.6 and 0.1 percentage points but lowers R@1 by 1.7 (T2V) and 0.6 (V2T) percentage points despite using $3.3\times$ more online parameters. The bottleneck therefore lies elsewhere; the evidence points back to the cache structure, the temporal supervision, and the use of the first-stage prior.

Table 16. Future-delta horizon ablation on MSR-VTT 1ka, with Recall@K in percentage points. The objective is not fragile to a single horizon; h=3 achieves the best R@1 in both directions.

Horizon	T2V R@1	T2V R@5	T2V R@10	V2T R@1	V2T R@5	V2T R@10
$h = 2$	53.8	78.2	85.5	53.0	78.7	86.6
$h = 3$ **(default)**	**54.6**	76.9	85.4	**54.6**	**79.3**	86.7
$h = 4$	54.4	**78.7**	**85.6**	53.0	78.2	**86.9**

Table 17. MSR-VTT 1ka reranker-backbone control, with Recall@K in percentage points. The BERT-base (Devlin et al., 2019) control raises the online reranker size from 33M to 110M parameters.

Variant	Online params	T2V R@1	T2V R@5	T2V R@10	V2T R@1	V2T R@5	V2T R@10
Default VEDJE	**33M**	**54.6**	76.9	85.4	**54.6**	**79.3**	**86.7**
BERT-base text encoder (Devlin et al., 2019)	110M	52.9	**77.5**	**85.5**	54.0	77.9	86.5