Knowledge Distillation Based on Large Video Retrieval Models

Rui Yuhan Qiao Shihan

12310520

12310437

December 23, 2025

Outline

Background

- Safety validation for autonomous driving depends on fast, accurate video-text retrieval across ever-growing fleets.
- Multimodal large language models (MLLMs) provide rich semantic reasoning but are difficult to deploy on embedded GPU/ASIC platforms with strict power and memory budgets.
- Bridging this compute-performance gap requires systematic lightweighting strategies that preserve retrieval fidelity while shrinking latency and model size.

Motivation

- **Significance:** Reliable retrieval accelerates data curation, rare-event mining, and closed-loop safety audits for intelligent vehicles.
- **Advantages:** Combining knowledge distillation (TeachCLIP) with post-training quantization (VAST) offers complementary savings in FLOPs, memory, and training time.
- **Gap:** Existing works seldom co-design teacher feature caching, student architecture tweaks, and INT4 compression within one pipeline tailored for autonomous driving datasets.
- **Limitations:** Distillation inherits teacher biases and quantization introduces accuracy loss, motivating careful analysis of trade-offs and future refinement.

Research Objectives

1. Engineer a TeachCLIP-based distillation workflow with offline feature extraction to balance throughput and fine-grained alignment.
2. Explore student/teacher architectural variants (ConvNeXt students, InternVideo2.5 teachers) to understand their practical benefits and constraints.
3. Quantize the VAST foundation model via MS-Swift GPTQ to INT4 precision, characterizing compression-efficiency versus retrieval accuracy for deployment.

Introduction

Efficient video retrieval is critical for autonomous driving safety validation and data mining, yet deploying powerful Multimodal Large Language Models (MLLMs) on resource-constrained vehicles faces a critical bottleneck: the high computational demand versus strict hardware limitations. This work focuses on **model lightweighting** strategies to bridge this gap.

We explore two complementary optimization pathways:

- **Knowledge Distillation (KD)**
- **Post-Training Quantization (PTQ)**

These efforts collectively demonstrate viable paths toward constructing lightweight, high-performance video retrieval systems suitable for on-vehicle deployment.

Introduction

- **Knowledge Distillation**: Leveraging the TeachCLIP framework, we distill fine-grained knowledge from heavy teacher models into lightweight student networks. We propose an "Offline Feature Extraction" strategy (improving training throughput by 9.7%) and investigate architectural optimizations using ConvNeXt encoders.
- **Post-Training Quantization**: We apply GPTQ-based quantization to the VAST foundation model, compressing to INT4 precision and achieving a 2.80× reduction in global model size (4.33× for the vision encoder).

Mainstream Knowledge Distillation Approaches I

KD transfers knowledge by having the student mimic the teacher's output distribution as soft labels, enabling effective model compression while maintaining performance.
Additionally, KD facilitates knowledge transfer from source tasks to target tasks with limited labeled data, making it particularly valuable for domain-specific applications such as autonomous driving video retrieval.

Mainstream Knowledge Distillation Approaches II

Knowledge distillation methods can be categorized based on the source of knowledge being distilled. The three primary categories are:

- **logit-based distillation**, which transfers final predictions using softmax with temperature to create soft labels;
- **feature-based distillation**, which transfers intermediate layer representations and attention maps, providing richer information than logits alone;
- **similarity-based distillation**, which transfers structural knowledge through pairwise similarities between features, channels, or instances.

Mainstream Knowledge Distillation Approaches III

Additionally, distillation can be organized by training schemes:

- **offline distillation**: pre-trained teacher with frozen weights
- **online distillation**: simultaneous training
- **self-distillation**
- **attention-based distillation**
- **contrastive distillation**
- **cross-modal distillation**

DLDKD I

DLDKD (Dual Learning with Dynamic Knowledge Distillation): a dual-stream teacher-student framework that transfers knowledge from a large teacher model to a lightweight student model through multi-level distillation.

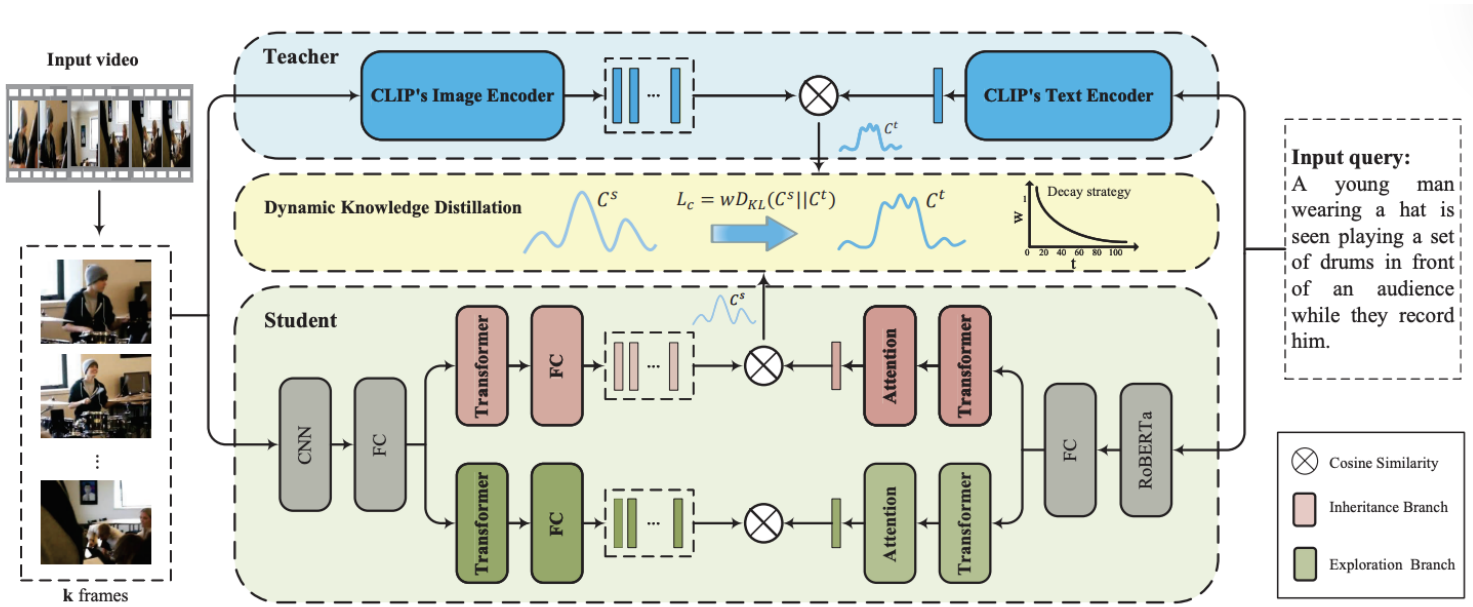


Figure: The framework of DLDKD.

DLDKD II

DLDKD performs distillation at multiple levels:

- feature-level distillation transfers intermediate representations
- logit-level distillation aligns output distributions
- attention-level distillation preserves attention patterns

DLDKD III

Loss Functions

The total loss function is defined as:

$$\mathcal{L}_{total} = \lambda_I \mathcal{L}_I + \lambda_E \mathcal{L}_E + \mathcal{L}_{task} \tag{1}$$

- λ_I: inheritance branch loss weight
- λ_E: exploration branch loss weight

Inheritance Student Branch Loss ($\mathcal{L}_I$): The loss combines feature-level, logit-level, and attention-level distillation:

$$\mathcal{L}_I = \alpha_f \mathcal{L}_{feat} + \alpha_l \mathcal{L}_{logit} + \alpha_a \mathcal{L}_{attn} \tag{2}$$

- α_f: feature-level distillation loss weight
- α_l: logit-level distillation loss weight
- α_a: attention-level distillation loss weight

DLDKD IV

- $\mathcal{L}_{feat}$: feature-level distillation loss(aligns intermediate representations between teacher and student)

$$\mathcal{L}_{feat} = \frac{1}{N} \sum_{i=1}^{N} \| F_t^{(i)} - F_s^{(i)} \|_2^2 \tag{3}$$

- $\mathcal{L}_{logit}$: logit-level distillation loss(KL divergence to align output distributions)

$$\mathcal{L}_{logit} = \mathrm{KL}(\mathrm{softmax}(z_t/\tau) \| \mathrm{softmax}(z_s/\tau)) \tag{4}$$

- $\mathcal{L}_{attn}$: attention-level distillation loss(preserves the teacher's attention patterns)

$$\mathcal{L}_{attn} = \frac{1}{L} \sum_{l=1}^{L} \| A_t^{(l)} - A_s^{(l)} \|_F^2 \tag{5}$$

DLDKD V

Exploration Student Branch Loss ($\mathcal{L}_E$): The exploration branch focuses on task-specific learning to capture domain-specific patterns.

$$\mathcal{L}_E = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp(\text{sim}(v_i, t_i^+)/\tau_E)}{\sum_{j=1}^{B}\exp(\text{sim}(v_i, t_j)/\tau_E)} \tag{6}$$

- B: batch size
- v_i: video embedding
- t_i^+: positive text query
- τ_E: temperature parameter for the exploration branch

Task-Specific Loss ($\mathcal{L}_{task}$): The task loss ensures that the final model performs well on the video-text retrieval task.

$$\mathcal{L}_{task} = \mathcal{L}_{retrieval} = -\log P(v^+|t) - \log P(t^+|v) \tag{7}$$

Dynamic Weight Adjustment: The weights λ_I and λ_E are dynamically adjusted during training based on the performance of each branch.

$$\lambda_I^{(t)} = \lambda_I^{(0)} \cdot \exp(-\gamma_I \cdot t), \quad \lambda_E^{(t)} = \lambda_E^{(0)} \cdot (1 + \gamma_E \cdot t) \tag{8}$$

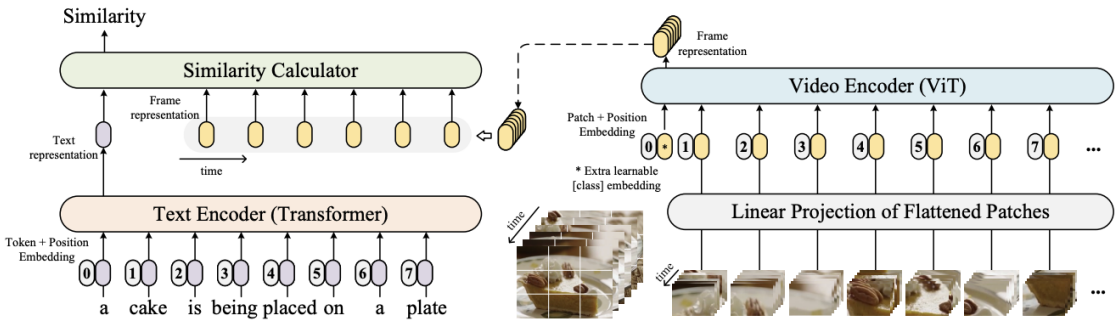
Feature Pre-Extraction Strategy in DLDKD

- DLDKD employs feature pre-extraction to accelerate training: teacher model features are pre-computed and cached for all training samples before distillation training
- We will incorporate this feature pre-extraction strategy to shorten the training time in TeachCLIP.

TeachCLIP Inspiration I

Existing VTR Models:

- **Lightweight Global Feature Models (e.g., CLIP4Clip):**
  - Computationally efficient.
  - Lack frame-level details → suboptimal accuracy.
- **Heavy Fine-grained Models (e.g., X-CLIP):**
  - Leverage frame-level interactions → high performance.
  - High computational costs.



TeachCLIP Inspiration II

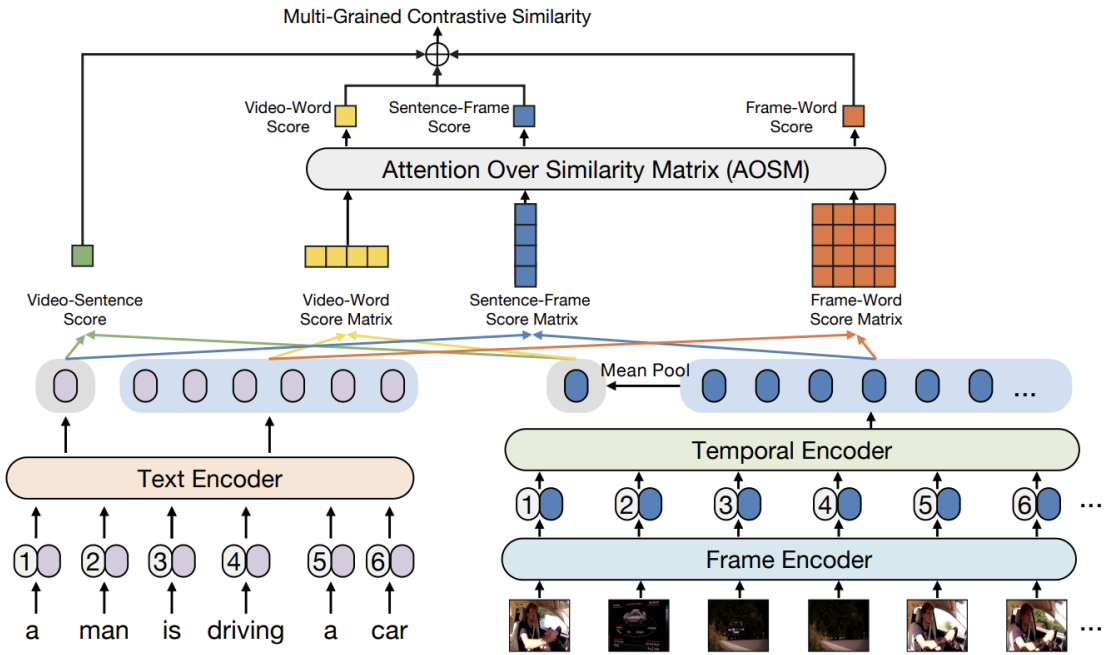


Figure: Comparison of CLIP4Clip and X-CLIP frameworks.

TeachCLIP Inspiration III

In terms of vision and text encoding, both CLIP4Clip and X-CLIP share nearly identical bottom-level encoding mechanisms (Frame $\rightarrow$ Patch $\rightarrow$ ViT, Word $\rightarrow$ Embedding $\rightarrow$ Transformer). The primary distinction lies in how they utilize these features in the upper layers. TeachCLIP aims to bridge this gap by distilling the fine-grained alignment capability of heavy models into a lightweight student model.

TeachCLIP Framework I

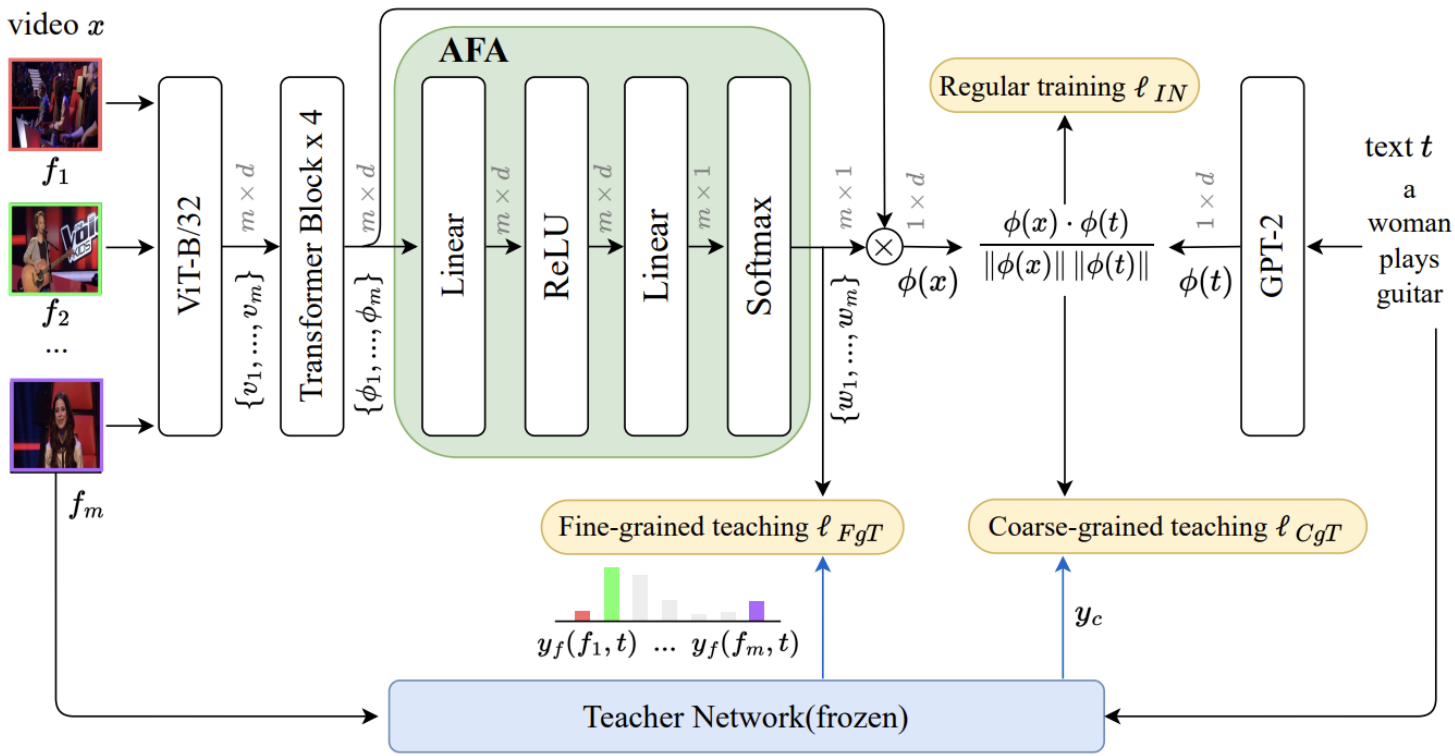


Figure: The framework of TeachCLIP.

TeachCLIP Framework II

Student Model: The student model is built upon CLIP4Clip. It samples the video into frames, processes each frame through the CLIP ViT to obtain frame features, and enhances them via a temporal Transformer. A key innovation is replacing the original mean pooling with Attentional Frame-Feature Aggregation (AFA). AFA generates frame weights $\{w_i\}$ to compute a weighted sum of frame features:

$$\phi(x) = \sum_{i=1}^{m} w_i \cdot \phi_i \tag{9}$$

The AFA module consists of a lightweight structure (Linear $\rightarrow$ ReLU $\rightarrow$ Linear $\rightarrow$ Softmax) and introduces negligible parameters.

TeachCLIP Framework III

Teacher Model: The teacher model provides two types of supervision:

- **Video-level soft labels (y_c):** The teacher provides the distribution of video-text correlations to guide the student's ranking.
- **Frame-level soft labels (y_f):** The teacher provides the distribution of frame-text correlations to guide the student's frame weight allocation in AFA.

Learning Objectives: The training involves three losses:

- *Frame-level Distillation (ℓ_{FgT}):* Optimizes AFA weights to match teacher's frame-text similarity.

$$\ell_{FgT} = -\frac{1}{b}\sum_{i=1}^{b}\sum_{k=1}^{m} y_f(f_{i,k}, t_i)\,\log w_{i,k} \tag{10}$$

TeachCLIP Framework IV

- *Video-level Distillation (ℓ_{CgT}):* Aligns video-text similarity matrices using Pearson distance.

$$\begin{aligned}\ell_{CgT} = &\frac{1}{b}\sum_{i} d_p(\sigma(B_{i,.}), \sigma(y_c(v_i, \cdot))) \\ &+\frac{1}{b}\sum_{j} d_p(\sigma(B_{.,j}), \sigma(y_c(\cdot, t_j)))\end{aligned} \tag{11}$$

- *Contrastive Learning (ℓ_{IN}):* Standard InfoNCE loss.

$$\ell_{IN} = \frac{1}{2}\left[\ell_{NCE}^{v\to t} + \ell_{NCE}^{t\to v}\right] \tag{12}$$

The total loss is $\ell = \ell_{CgT} + \ell_{FgT} + \ell_{IN}$.

TeachCLIP vs X-CLIP Parameters and FLOPs Comparison

Table 1 presents the comparison of parameters and FLOPs between TeachCLIP and the teacher model (X-CLIP).

Table: Comparison of Parameters and FLOPs

Model	**Para**	**FLOPs**	**Description**
TeachCLIP	$\approx$ 200M	53.65G (12 frames)	Student model, based on CLIP4Clip + AFA
X-CLIP (Teacher)	$\approx$ 220M	145G (8 frames) / 287G (16 frames)	Teacher model, multi-grained contrastive similarity

Feature Pre-computation Acceleration

Strategy: Offline Feature Extraction

- **Motivation:** Eliminate expensive video encoding overhead during training.
- **Method:**
  - **Offline:** Pre-compute frame features using Teacher (X-CLIP) and cache as $(\mathbf{V}, \mathbf{m})$.
  - **Online:** Student loads features directly, bypassing raw pixel re-encoding.
- **Outcome:** Enhances training throughput and reproducibility.

Table: Training Efficiency with Precomputed Teacher Features

Setting	**Time/step (s)**	**Speedup**
Online Extraction (Baseline)	1.91	–
Offline Pre-computation	**1.74**	**+9.7%**

On Student Model I

Objective: Replace ViT with ConvNeXt (CNN) to leverage inductive biases (locality, translation invariance) using OpenCLIP.

Exploration on COCO Dataset (ConvNeXtV2-Tiny)

- **Finding:** Full fine-tuning leads to overfitting.
- **Optimal Strategy:** "Controlled Gradual Unfreezing" (Text unfreeze @ ep 3, Visual @ ep 5).
- **Result:** Best trade-off achieved (I2T R@1: 9.55%).

Table: Comparison of ConvNeXtV2-Tiny Strategies on COCO

Strategy	**I2T R@1**	**Observation**
No Freeze	8.12%	Overfitting, worst performance
Unfreeze 3-5(Gradual)	**9.55%**	**Best balance**
Change Text Tower + Unfreeze	9.07%	Intermediate performance

On Student Model II

Implementation on TeachCLIP with MSRVTT Dataset

- **Setup:** Integrated ConvNeXt-Base encoder into TeachCLIP.
- **Result:** Direct migration yielded suboptimal results (R@1 $\approx$ 2.6–3.4%) compared to ViT baselines.
- **Conclusion:** Simple substitution is insufficient. Effective adaptation requires specialized temporal modeling (e.g., 'absattn') to capture video dynamics.

Table: Preliminary Results of ConvNeXt-based Student on MSRVTT

Setting	**Text-to-Video R@1**
MSRVTT-9k (10 epochs)	3.4%
MSRVTT-7k (10 epochs)	2.5%
MSRVTT-7k (16 epochs, Locked)	2.6%

On Teacher Model I

To utilize the computationally expensive InternVideo2.5 as a teacher without the overhead of online inference, we implemented an "Offline Feature Extraction - Online Loading" scheme:

- **1. Offline Feature Pre-computation:** To avoid expensive repeated inference, we pre-compute and store InternVideo2.5 features (global, text, frame-level) as a local knowledge base.
- **2. Data Pipeline Index Alignment:** We implemented unique Sample IDs to strictly align shuffled student inputs with the corresponding offline teacher features.
- **3. Training Logic Adaptation:** The training loop was refactored to skip teacher inference, instead utilizing an 'Offline Feature Loader' to feed pre-computed similarity matrices and weights directly into the distillation loss.

On Teacher Model II

Results on MSRVTT

- **Evaluation:** Student models distilled from different teachers on MSRVTT.
- **Observation:** InternVideo2.5 is a stronger teacher, and its student (TeachCLIP w/ InternVideo2.5) outperforms the X-CLIP baseline.
- **Improvement:** Achieved gains in R@1 (+1.0%), R@5 (+1.5%), and R@10 (+1.7%).
- **Conclusion:** Verifies that stronger teachers can transfer better representations to students.

Table: Comparison of Different Teachers on MSRVTT

Model	**R@1**	**R@5**	**R@10**
X-CLIP	49.3	75.8	84.8
TeachCLIP (Teacher: X-CLIP)	46.8	74.9	82.9
InternVideo2.5 (Teacher)	55.9	78.3	85.1
TeachCLIP (Teacher: InternVideo2.5)	47.8	76.4	84.6

Post-Training Quantization of VAST

We apply post-training quantization methods (MS-Swift) to compress the VAST model for efficient video-text retrieval.

MS-Swift Quantization Methodology:

- **Framework:** Implements GPTQ-based quantization (INT4 precision).
- **Technique:** Block-wise sequential quantization with Hessian-based optimization to minimize reconstruction error.

Key Configuration:

- **Precision:** INT4.
- **Parameters:** Group size 128, symmetric quantization, sequential quantization (preserves inter-layer dependencies).
- **Selective Quantization:** Applied only to *vision encoder blocks*, preserving full precision for text/audio branches.

VAST Architecture for Video-Text Retrieval I

VAST (Vision-Audio-Subtitle-Text) is an omni-modality foundation model. The VAST framework architecture is illustrated in Figure 4.

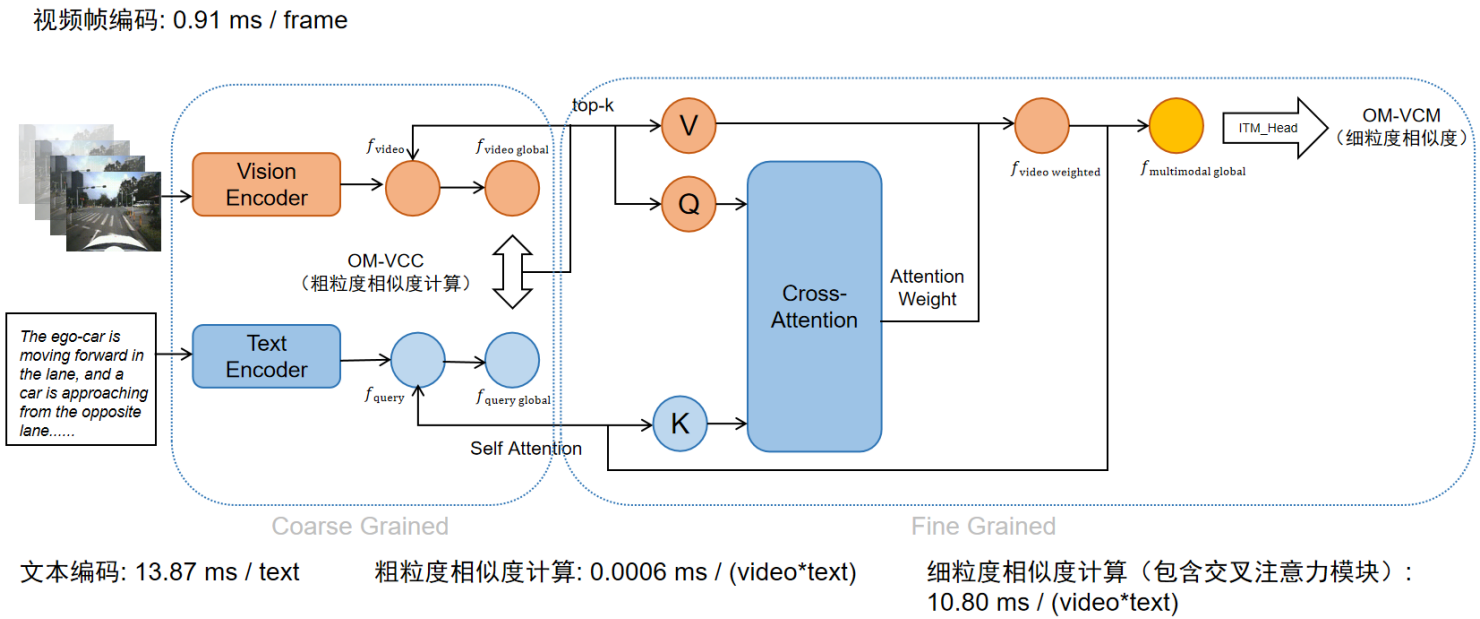


Figure: The framework of VAST.

- **Dual Encoder Architecture:**

VAST Architecture for Video-Text Retrieval II

- **Video Encoder:** Processes frames through a ViT architecture with temporal modeling.
- **Text Encoder:** Based on transformer architecture.

- **Learning Mechanism:** Learns cross-modal alignment in a shared embedding space, using contrastive learning to maximize similarity between matching video-text pairs.
- **Motivation for Optimization:**
  - The VAST model achieves strong performance after fine-tuning on domain-specific data.
  - Given the model's size, lightweight optimization is required for real-time applications.
  - We apply post-training quantization to obtain smaller model weights while maintaining competitive performance.

Post-Training Quantization Experimental Process (1/2)

Quantization Setup: The quantization is performed on the fine-tuned VAST model using MS-Swift's GPTQ implementation. We configure the quantization to target INT4 precision with the following key parameters:

- Group size of 128 for grouped quantization.
- Symmetric quantization (sym=true).
- Sequential quantization (`true_sequential`=true) to maintain layer dependencies.

The quantization is selectively applied only to the vision encoder blocks (`vision_encoder.visual.blocks`), preserving the full precision of other components including the text encoder and audio branches.

Calibration Data: We use 200 calibration samples in JSONL format, each containing text queries and corresponding image frame sequences. These samples are forward-passed to collect activation statistics (Hessian matrices) for guiding the optimization.

Post-Training Quantization Experimental Process (2/2)

Quantization Process: The quantization proceeds block-by-block through 40 vision encoder blocks. For each block, four linear modules are quantized sequentially:

- Attention QKV projection (`attn.qkv`)
- Attention output projection (`attn.proj`)
- MLP layers (`mlp.fc1`, `mlp.fc2`)

The process involves collecting activations, computing Hessian matrices for weight sensitivity, performing grouped quantization (converting FP16 to INT4), and applying error compensation. Weights are packed (4 INT4 values into 1 INT16) for storage.

The quantized VAST model maintains competitive retrieval performance with significantly reduced model size and memory footprint, enabling more efficient deployment for real-time video-text retrieval applications.

Quantization Results and Comparison

Table 6 and Table 7 compare the model statistics and retrieval performance before and after quantization.

Table: Model Statistics (FP16 vs INT4)

Metric	**FP16**	**INT4**	**Ratio**
File Size	5.20G	1.86G	2.80×
Params	1.40B	0.52B	2.67×
Vis. Enc.	1.14B	0.26B	4.33×
Weights	5.33G	1.99G	2.67×

Table: Suscape Performance

Metric	**FP16**	**INT4**
R@1	64.4	37.3
R@5	88.7	55.9
R@10	97.2	65.2
Avg	83.4	52.8

The quantization results demonstrate effective compression: the vision encoder achieved **4.33**× compression, closely matching the theoretical limit. The overall model size was reduced by **2.80**×.
Expected performance degradation is observed (R@1 dropped from 64.4% to 37.3%), indicating a trade-off between model size and accuracy.

Conclusion

Summary of Contributions

- **Knowledge Distillation (TeachCLIP):**
  - Optimized training with offline feature extraction (**9.7% speedup**).
  - Explored ConvNeXt student and InternVideo2.5 teacher; identified need for specialized temporal modeling.
- **Model Quantization (VAST):**
  - Applied GPTQ INT4 quantization using MS-Swift.
  - Achieved **2.80× size reduction** (5.20 GB → 1.86 GB) with 4.33× vision encoder compression.
  - Addressed trade-off between edge deployment feasibility and accuracy drop.

Future Work

1. Refine distillation for InternVideo2.5 (better alignment strategies).
2. Explore recovery strategies like Quantization-Aware Training (QAT) to minimize accuracy degradation.

Thank You!

Q & A