Tem

Sure! Here's the full English translation of the content you provided:

Models Used in Evaluation

Function	Default Model	Parameter Count	Remarks
ASR	Distil-Whisper	—	Run Whisper model locally to extract audio transcripts
VLM	MiniCPM-V-2_6-int4	—	Load MiniCPM model locally (.bin weights included in the project)
LLM - Reasoning	Qwen2.5_VL_7B	7B	Default uses OpenAI API; can be switched to local inference via Ollama
LLM - Query	qwen2.5_0.5B	0.5B	—
Text Embedding	inf-retriever-v1-1.5b	1.5B	—
Visual Embedding	ImageBind	—	Run locally to generate clip-level embeddings

---

Video Workflow

VideoRAG technology combines RAG and LVLM techniques. Its process is divided into two main steps:

1. Video Retrieval

- **Goal:** Retrieve a set of relevant videos $V = \{V_1, V_2, ..., V\}$ from a large video corpus C based on a user query q.

- **Method:**
  - Encode each video (frames + transcripts) and the query `q` using a LVLM to get embeddings `fvideo` and `fquery`.
  - Use a similarity function (e.g., cosine similarity) to measure similarity between `fvideo` and `fquery`.
  - Select the top-k most relevant videos based on similarity scores.

2. **Video-Augmented Response Generation**

- **Steps:**
  a. Concatenate each video's frames and text info into pairs: $[V_1, t_1], [V_2, t_2], \ldots,$ $[V\ , t\ ]$.
  b. Add the user query `q` to the input: $[V_1, t_1, \ldots, V\ , t\ , q]$.
  c. Feed the multimodal input into the LVLM for processing and response generation.
- **Goal:** Leverage both visual and textual information from the videos to enhance answer accuracy, richness, and contextual understanding.

Current Research Status

Currently, there are **no open-source, end-to-end large models** for VideoRAG. Research uses a **modular approach**, combining:

- **Retriever**: CLIP, InternVideo, BLIP-2, Faiss/HNSWlib
- **Generator**: Video-LLaVA, GPT-4, MiniCPM-V, LLaMA, ChatGLM

Evaluation Methods

1. **Win-Rate Comparison** (uses OpenAI API)

Compare the win rates of responses from VideoRAG with other RAG-based baselines.

2. **Quantitative Comparison** (uses OpenAI API)

Enhance win-rate comparison by assigning a **5-point score** for long-context video understanding. Use NaiveRAG's answer as a baseline for scoring.

Scoring Prompt Format:

- Q: [User Question]
- Retrieved Video(s): [Subtitle snippets or screenshots]

- Answer: [Generated Answer]

Rate the answer quality from 1 to 10 based on:

- Relevance to the question
- Specificity and detail (does it actually use the video context?)
- Factual correctness

Then briefly explain your rating.

3. Module-level Evaluation

- **Retrieval Metrics:** Recall@K, Precision@K, MRR
- **Answer Metrics:** ROUGE-L, BLEU, METEOR, BERTScore
  - Require **ground truth Video/Answers**, which can come from public datasets or be self-created using structured query generation.

Example Workflow for Query/Answer Dataset Construction:

- Sample from 500 videos
- Use GPT-4 + video transcripts to generate **open-ended, video-relevant** questions (e.g., "How do you test if a solar panel is working?")
- Produce 1000 queries for retrieval and QA tasks

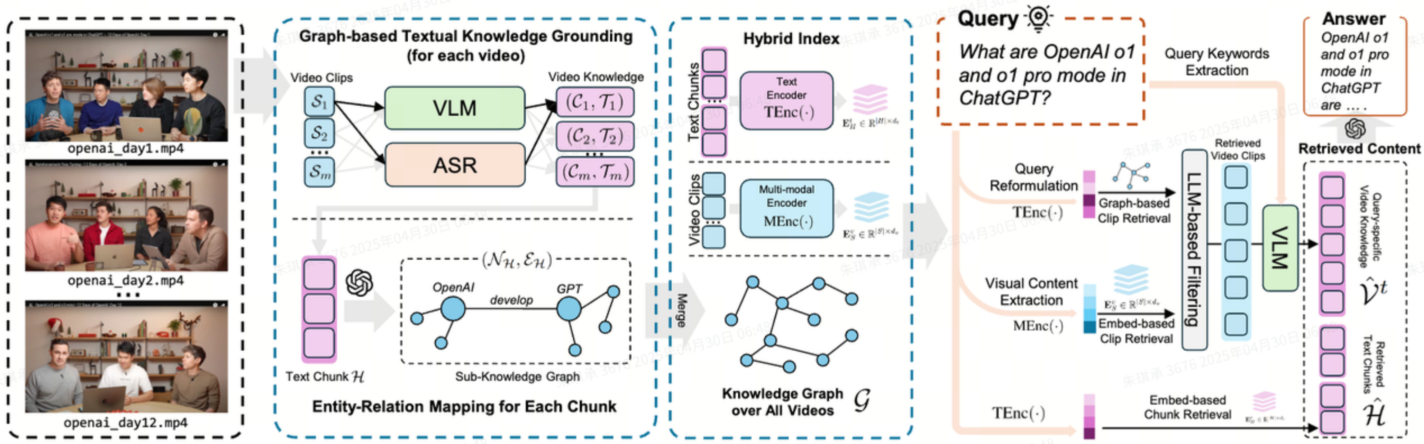
4. Custom Evaluation Metrics

- **Relevancy Score (RS)**
- **Correctness Score (CS)**
  (From paper: https://huggingface.co/papers/2501.03995)

Datasets

Name	Size	Link	Summary
LongerVideos	160+ videos, 600+ QA queries	[GitHub - HKUDS/VideoRAG]	Public videos (lectures, documentaries), long-context, open-ended QA
TVQA	—	[TVQA Corpus]	—
Howto100M	18.4M	[HuggingFace]	Large-scale instructional video dataset
Cinepile	—	[HuggingFace]	Movie-related multimodal QA dataset

Paper Summary



(a) Videos as Knowledge Base (b) Multi-Modal Video Knowledge Indexing (c) Multi-Modal Retrieval Paradigm

Phase 1: Indexing

This phase performs 3 tasks using Distil-Whisper and MiniCPM-V:

1. **Text Embedding**
   Using text chunks with `text-embedding-3-small`
2. **Multimodal Embedding**
   Using video clips with **ImageBind**
3. **Knowledge Graph Construction**
   Use text chunks to extract entities and update descriptions with **GPT-4o-mini**

$$\mathcal{G} = (\mathcal{N}, \mathcal{E}) = \bigcup_{\mathcal{H} \in \{\mathcal{V}_1^t, \ldots, \mathcal{V}_n^t\}} (\mathcal{N}_{\mathcal{H}}, \mathcal{E}_{\mathcal{H}}),$$

Phase 2: Retrieval

Two paths for retrieval:

1. **Knowledge Graph-Based**
   - Query Reformulation (GPT-4o-mini) → Entity Matching → Chunk Selection → Clip Extraction
2. **Query Reformulation-Based**
   - Scene Info Extraction from Query (GPT-4o-mini) → Query Embedding (VLM) → Similarity → Chunk → Clip

Both are followed by:
LLMs-based Video Clip Filtering

$$\{\hat{\mathcal{S}} \mid (\hat{\mathcal{S}} \in \{\mathcal{S}\}_q^t \cap \{\mathcal{S}\}_q^v) \land \text{LLMs-Judge}(\mathcal{V}_{\hat{\mathcal{S}}}^t) = 1\},$$

Phase 3: Answering

Use LLM to generate a response, based on:

1. Extracted key terms
2. Relevant text embeddings (from similar chunks)
3. Selected video clips with frames → captions and transcripts (from Phase 1 again)