# From Moments to Meanings: Egocentric Temporal Localization and Answer Generation with VSLNet and Video-LLaVA on Ego4D

**Arian Mohammadi**[1] **Hassine El Ghazel**[2] **Hadi Abdallah**[3] **Ali Ayoub**[4]
{s346278[1], s346265[2], s339398[3], s339204[4]}@studenti.polito.it

## Abstract

*Egocentric vision focuses on understanding videos captured from a first-person perspective, posing unique challenges such as rapid camera motion and a restricted field of view. The Ego4D dataset provides a comprehensive benchmark for this domain through the Natural Language Queries (NLQ) task, which requires retrieving relevant video segments based on textual queries.*

*In this work, we evaluate the Video Span Localizing Network (VSLNet) and its simplified variant, VSLBase, under various configurations of video and text feature extractors—including Omnivore and EgoVLP for visual features, and GloVe and BERT for language embeddings. To extend egocentric video understanding beyond temporal localization, we propose a two-stage pipeline that combines segment retrieval with Video-LLaVA for natural language answer generation, enabling both fine-grained localization and semantic reasoning. We then evaluate how well Video-LLaVA performs compared to other multimodal natural language models.*

***GitHub repository:*** *EgoCentricVisionPolito.*

## 1. Introduction

Egocentric videos, captured from a first-person perspective, introduce significant challenges for video understanding, including rapid motion, occlusions, and a restricted field of view. These characteristics demand models capable of fine-grained temporal reasoning and robust language-visual alignment [1]. A central task in this domain is Natural Language Queries (NLQ) [6], which involves retrieving the temporal segment in a video that best matches a given natural language description. Solving NLQ requires precise temporal localization and deep cross-modal understanding between video and language modalities.

In this work, we benchmark two span-based temporal localization models: VSLBase, a feature similarity baseline, and VSLNet [18], which enhances query-conditioned attention through a Query-Guided Highlighting (QGH) mechanism. We evaluate these models on the Ego4D NLQ benchmark [6], comparing different visual feature extractors—Omnivore [5] and EgoVLP [10]—and language embeddings—GloVe [14] and BERT [3]. In addition to standard localization, we extend the NLQ pipeline by generating natural language answers based on the retrieved video segments. This extension bridges video retrieval with video question answering, aligning with recent trends in video-language.

## 2. Related Work

Natural Language Video Localization (NLVL) aims to retrieve temporal segments in videos that correspond to natural language queries. Early efforts, such as ActivityNet Captions [9] and Charades-STA [15], introduced joint video-text embeddings with attention mechanisms, establishing foundational approaches. Video representations have since advanced with models like Omnivore [5], trained on diverse visual modalities, and EgoVLP [10], optimized for egocentric video-language tasks. Language modeling has likewise evolved—from static embeddings like GloVe [14] to contextualized encoders such as BERT [3]. These representations feed into temporal localization models that align queries with relevant video moments. Modern approaches have moved beyond simple similarity measures toward query-aware mechanisms. VSLNet [18], for example, introduced Query-Guided Highlighting (QGH) to improve the alignment of text and video through early cross-modal interaction. Moreover, recent studies have advanced temporal localization by generating natural language responses based on extracted video segments. These methods enhance semantic understanding by requiring models to reason across visual and textual inputs. Systems like Video-ChatGPT [13], Video-ChatCaptioner [19], and Video-LLaVA [11] use large language models alongside video encoders to facilitate open-ended video question answering.

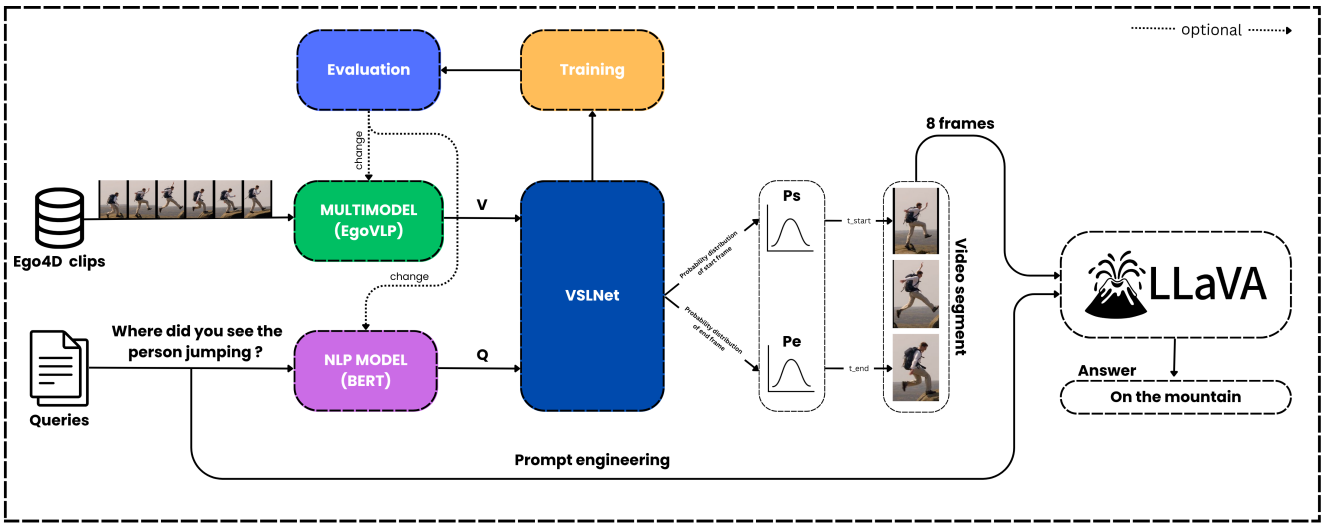


Figure 1. Overview of the pipeline combining VSLNet and LLaVA for temporal localization and visual answer generation on Ego4D clips

# 3. Methodology

## 3.1. Span Localization Architecture

### 3.1.1 NLQ Task Definition

The Natural Language Queries (NLQ) task focuses on localizing temporal segments in videos that best match a given natural language query. Formally, given a video $v$ and a query $q$, the objective is to predict a segment defined by start and end times $[t_s, t_e]$ that maximizes the overlap with the ground-truth segment. We follow the Ego4D NLQ benchmark [6] for data and evaluation protocols.

### 3.1.2 Text Query Representations

We evaluate two types of pretrained text embeddings for encoding natural language queries. **GloVe 840B.300d** [14] provides static 300-dimensional word vectors trained on 840 billion tokens, representing queries via token-level averages without contextual nuance. In contrast, **BERT** [3] produces contextualized embeddings using Transformer-based attention, capturing richer syntactic and semantic information. This comparison allows us to assess the impact of static versus contextual language representations on egocentric video localization performance.

### 3.1.3 Pre-Extracted Visual Features

We use official pre-extracted video embeddings from the Ego4D dataset [6]. Our experiments utilize two types of visual features: **Omnivore FP16 (v1)**, **EgoVLP FP16**.

**Omnivore FP16** features are extracted using the Omnivore model [5], which incorporates a Swin-L backbone and a video head pretrained on Kinetics-400 [7] and ImageNet-1K [2]. This model yields high-quality general-purpose video representations.

**EgoVLP FP16** [10], on the other hand, is a contrastive video-language model trained directly on Ego4D data to align egocentric video clips with their textual descriptions. As training EgoVLP from scratch requires roughly 1,536 GPU hours on NVIDIA A100s, we follow standard practice by leveraging its publicly available pre-extracted features, which are particularly well-suited for egocentric understanding.

All video inputs are segmented into 32-frame windows with a stride of 16 frames, following the official NLQ feature extraction protocol. Feature dimensionalities differ across models (256 for EgoVLP and 1536 for Omnivore), and the `VIDEO_FEATURE_DIM` parameter is set accordingly during training to ensure proper tensor alignment. These features are used as input to both VSLBase and VSLNet architectures in our pipeline, supporting consistent and efficient evaluation without full end-to-end training.

### 3.1.4 Localization Models

We benchmark two span-based temporal localization models to assess the impact of query-conditioned attention.

**VSLBase** follows a span-based temporal localization framework [18], where video and query representations are processed independently through modality-specific encoders. These representations are projected into a common embedding space, where a cross-modal similarity matrix is computed. The model then applies span prediction using two parallel feed-forward layers to estimate the start and end positions of the target segment. VSLBase is relatively simple and does not incorporate any interaction between video frames and the query during encoding.

**VSLNet** builds upon VSLBase by introducing the Query-Guided Highlighting (QGH) module [18], which enhances temporal localization by injecting query awareness

into the video representation early in the network. QGH employs multi-head attention to highlight frames that are semantically aligned with the query, allowing for finer-grained reasoning across time.

**Key Differences:** While VSLBase processes modalities in isolation and fuses them only during span prediction, VSLNet introduces early cross-modal interaction via QGH, enabling more precise alignment between the query and relevant video content.

#### 3.1.5 Training Details

To ensure a fair and consistent comparison, all models in our experiments were trained using the same hyperparameters and procedures. We evaluated eight model variants by combining two architectures (VSLBase and VSLNet), two types of visual features (Omnivore and EgoVLP), and two text encoders (BERT and GloVe 840B.300d), resulting in $2 \times 2 \times 2 = 8$ configurations.

All models were trained for 10 epochs with a batch size of 32. The internal hidden size and maximum position length were both set to 128. Optimization was performed using Adam [8] with an initial learning rate of 0.0025, which decayed linearly during training. Dropout [16] with a rate of 0.2 was applied to all layers to mitigate overfitting. Three checkpoints were saved per epoch, resulting in 30 checkpoints per model configuration. The dimensionality of video features was 256 for EgoVLP and 1536 for Omnivore. This parameter was passed to the model through an environment variable to ensure correct tensor alignment. Note that this setting is distinct from the internal hidden dimension, which remained fixed at 128 across all runs.

Table 1. Fixed hyperparameters for all 8 model configurations.

| Parameter | Value |
|---|---|
| Number of epochs | 10 |
| Batch size | 32 |
| Hidden dimension (`--dim`) | 128 |
| Max position length | 128 |
| Initial learning rate | 0.0025 |
| Optimizer | Adam |
| Dropout rate | 0.2 |
| Checkpoints per epoch | 3 |
| Feature dimension (EgoVLP) | 256 |
| Feature dimension (Omnivore) | 1536 |

As shown in Table 1, the training setup was consistent across all experiments, ensuring that any variation in results could be attributed to differences in model architecture, visual features, or text representations.

**Baseline Comparison.** To evaluate the effectiveness of our model variants, we also compare our best-performing configuration against the official Ego4D baseline, which uses VSLNet with SlowFast [4] features as reported by Grauman et al. [6]. This allows us to assess the impact of stronger video-language representations (EgoVLP, Omnivore) and contextual language embeddings (BERT) on NLQ performance.

#### 3.1.6 Metrics Used for Evaluation

**Evaluation Metrics.** We adopt the evaluation metrics defined in the Ego4D NLQ benchmark [6, 10], focusing on temporal grounding performance across different thresholds and ranks. Our metrics include:

- **Rank@$k$@$\mu$** (e.g., Rank@1@0.5): Percentage of queries for which at least one of the top-$k$ predicted segments overlaps with the ground-truth segment at IoU $\geq \mu$. We report:

  **Ranks:** Rank@1, Rank@3, Rank@5

  **IoU thresholds:** 0.3, 0.5, 0.01

- **Mean IoU (mIoU):** Average IoU between top-1 predicted segment and ground truth across the dataset. Unlike thresholded recall, mIoU reflects alignment quality without cutoff.

This comprehensive evaluation enables nuanced analysis under both strict and lenient localization conditions and across multiple retrieved candidates.

### 3.2. Textual Answer Generation from Localized video Segments

Our modular pipeline generates semantic responses by pairing a natural language query with 8 sampled frames from localized video segments. This query-frame input is fed to various VLM's, primarily **Video-LLaVA**, comparing its segment-conditioned answer generation against commercial models like **Gemini 1.5/2.5** and **ChatGPT-4o** under identical conditions. Outputs are evaluated against ground truth using ROUGE, BLEU, and BERTScore (Figure 2).

#### 3.2.1 Input format preparation

For textual answer generation, we selected 200 high-quality VSLNET EgoVLP BERT segment predictions (chosen for superior performance). Using a 6-second tolerance against Ego4D NLQ validation ground truth, we identified the best localized segments. We then extracted and segmented corresponding video clips via Ego4D CLI and ffmpeg, labeling files with query data. To assess our generation of textual responses, we manually annotated answers for the 200 selected NLQ samples using the official Ego4D visualizer website. By accessing the `video_uid`, we reviewed the relevant temporal segments identified by the VSLNet prediction. This allowed us to formulate concise answers, which were recorded in a shared annotation file.

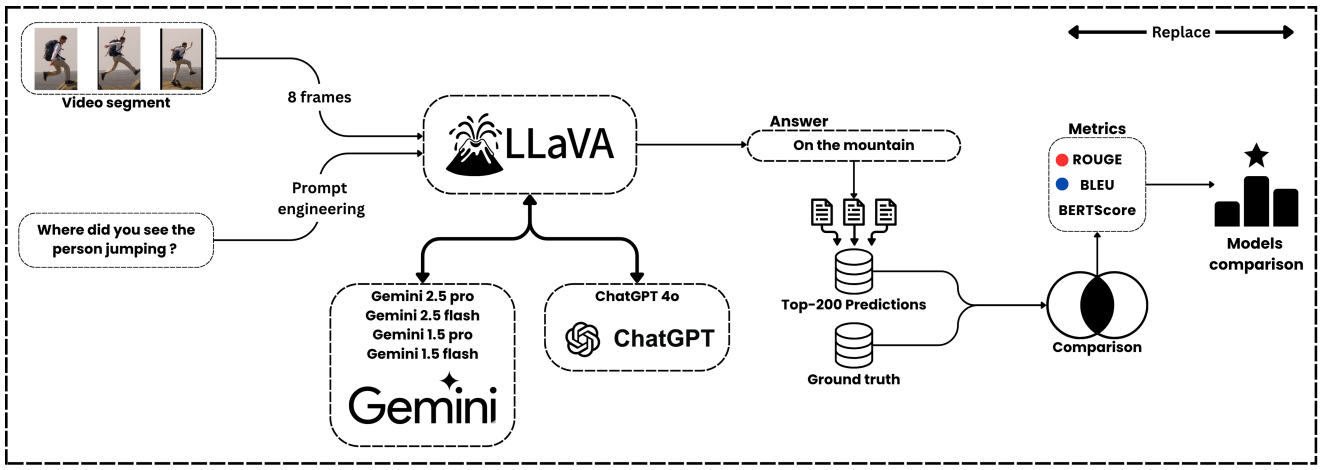


Figure 2. Evaluation pipeline for egocentric answer generation. Each 8-frame video segment and query is used to prompt various vision-language models, including Video-LLaVA, Gemini (1.5/2.5, Pro/Flash), and ChatGPT-4o. Generated answers are compared to ground truth using ROUGE, BLEU, and BERTScore.

We then linked each answer to its corresponding data point by combining three elements: the path to the extracted segment (`video_path`), the NLQ query (`question`), and the annotated response (`ground_truth`). For model's input preparation, we extracted **eight evenly spaced frames** from each localized video segment and paired them with **a prompt derived from the query.** The choice of using eight frames is not arbitrary; it directly follows the default configuration of the Video-LLaVA framework [12], which is the model we had chosen to work with later. This ensures compatibility with the model's vision encoder while providing sufficient temporal coverage of the segment.

#### 3.2.2 A vision language model, Video-LLaVA

Building on this, we used **Video-LLaVA-7B**, a cutting-edge vision-language model that can handle both video and text input in a unified approach. Video-LLaVA builds on the LLaVA (Large Language and Vision Assistant) framework by incorporating temporal video features and aligning them with language prompts. Unlike standard image-language models, Video-LLaVA is designed to **understand sequential visual information**—such as motion, scene changes, and temporal context—making it especially effective for tasks like temporal localization, video captioning, and answering questions. We selected Video-LLaVA for its strengths in multimodal understanding and answer generation. It integrates visual and textual inputs into a shared embedding space, enabling effective **cross-modal reasoning**. Its strong **zero-shot capabilities** stem from its pretrained LLM backbone, which eliminates the need for task-specific fine-tuning. Moreover, it supports **free-form natural language responses**, offering greater flexibility than classification-based approaches. Video-LLaVA encodes frames with a vision encoder (e.g., EVA-ViT), aligns them in the language space, and generates answers using an LLM decoder **(e.g., LLaMA-7B [17]).**

The exact configuration of the model, quantization setup, and inference settings are summarized in Table 2. Answer generation was performed in **25 batches**, with special attention to short video segments and efficient GPU memory usage. The resulting outputs include the generated answers, ground truth labels, and associated metadata for evaluation.

Table 2. Configuration details of the `Video-LLaVA-7B` model used in our experiments.

| Parameter | Value |
|---|---|
| Model Name | `Video-LLaVA-7B-hf` |
| Language Backbone | `LLaMA-7B` |
| Video Encoder | `EVA-ViT` |
| Quantization | 4-bit NF4 (`bnb_4bit_quant_type="nf4"`) |
| Compute Dtype | `torch.float16` |
| Quantization Library | `BitsAndBytes` |
| Device Mapping | `device_map="auto"` |
| Max Token Length | 3000 tokens[1] |
| Input Modalities | 8 Video frames + Natural language query |
| Processor | `VideoLlavaProcessor` |

#### 3.2.3 Metrics of answer evaluation

We used **ROUGE**, **BLEU**, and **BERTScore** to assess the quality of generated answers, as each metric captures different aspects of textual and semantic fidelity. **ROUGE-1** and **ROUGE-2** evaluate unigram and bigram recall, while **ROUGE-L** and **ROUGE-Lsum** capture sentence-level flu-

[1]We used 3000 tokens because, when comparing with Gemini 2.5 Pro/flash, it failed to return answers until the token limit was raised to approximately 3000. For fairness, we used the same configuration across all models.

ency via the longest common subsequence. **BLEU-1** to **BLEU-4** were computed using `corpus_bleu` from NLTK with smoothing (method 4), which helps stabilize scores for short, open-ended outputs. We used progressive n-gram weight configurations, summarized in Table 3, where higher-order BLEU scores require longer phrase matches and thus act as stricter measures of syntactic alignment.

Table 3. BLEU n-gram weights

| Metric | Weights (n-gram) |
|---|---|
| BLEU-1 | (1.0, 0, 0, 0) |
| BLEU-2 | (0.5, 0.5, 0, 0) |
| BLEU-3 | (0.33, 0.33, 0.33, 0) |
| BLEU-4 | (0.25, 0.25, 0.25, 0.25) |

**BERTScore** measures semantic similarity using contextual embeddings from a pre-trained transformer (e.g., BERT or RoBERTa), and reports **Precision**, **Recall**, and **F1** scores based on token-level meaning. This combination of lexical overlap and deep semantic matching enables robust evaluation of generated responses in the open-ended NLQ task.

#### 3.2.4 Lightweight prompt engineering

After preparing the input format **(8 frames + prompt)** for **Video-LLaVA-7B**, we evaluated the quality of the generated responses with testing on two prompting approaches to help Video-LLaVA produce concise responses. The first involved a static general prompt, while the second adjusted the prompt according to the question type:

**Common Prompt Structure**

**SYSTEM:** You are an assistant that provides answers based on visual summaries of video segments. Use only the information visible in the video. Be precise and respond in 5 words or fewer.
**CONTEXT:** /video/ (Visual summary of the video is shown here)
**USER:** Question: /natural language query/

**Adaptive Prompt Extension (Example)**

**TASK INSTRUCTION:** This is a color detection task. Identify the visible color of the mentioned object. **(Changes depending on the question template)**

We used this lightweight prompting setup to enable a controlled comparison between fixed and task-adaptive prompts without requiring model fine-tuning.

#### 3.2.5 Comparison of Video-LLaVA with Commercial Models

Given that **Video-LLaVA** is an open-source and freely available model, we specifically compare it to leading commercial counterparts, including systems like **Gemini 1.5**, **2.5 (Pro/Flash)** by Google, and **GPT-4o** by OpenAI, which excel in image-conditioned prompting for vision-language tasks. The goal is to evaluate how well Video-LLaVA performs relative to these proprietary systems within the context of the **Natural Language Queries (NLQ)** task on the **Ego4D** dataset. Our comparison emphasizes two main aspects: (1) **performance**, based on the accuracy and relevance of generated answers measured using standard metrics (see previous subsection); and (2) **accessibility**, considering the availability and cost of each model as of June 2025. This analysis highlights both the strengths and trade-offs of adopting an open-source vision-language model like Video-LLaVA over commercial multimodal LLMs.

#### 3.2.6 Setup of comparison settings

We compared **Video-LLaVA-7B** against these MLLMs for the Ego4D NLQ task using the same maximum token length: 3000, the same adaptive prompt, and practically the same visual input: Unlike Video-LLaVA, which directly utilizes 8 frame embeddings as separate inputs, models like GPT-4o and Gemini require the visual context to be encoded into a single image. We arranged the 8 selected frames into a $2 \times 4$ grid using the Python Imaging Library (PIL).

## 4. Experiments

### 4.1. Temporal localization models for the Natural Language Query (NLQ) task

#### 4.1.1 Ego4D NLQ Dataset Overview

We present a statistical summary of the Ego4D Natural Language Queries (NLQ) dataset in Table 4. This table outlines key descriptive statistics of the query durations, clip durations, and their relative sizes. The *relative query size* is computed as the ratio between the query duration and the full clip duration, representing how much of the clip is relevant to a query. The dataset comprises thousands of egocentric video clips, each paired with natural language queries and their corresponding temporal intervals. Notably, the distribution of query durations is highly skewed toward shorter segments, as reflected by high positive skewness and kurtosis values. This skewness is also mirrored in the relative query sizes, suggesting that the majority of queries refer to short moments, often near the beginning of each clip [6].

Table 4. Descriptive statistics of query size, clip size, and relative query size from the Ego4D NLQ dataset [6].

| | Query Size | Clip Size | Relative Query Size |
|---|---|---|---|
| Count | 11296.00 | 11296.00 | 11296.000 |
| Mean | 9.67 | 522.68 | 0.020 |
| Std | 22.83 | 197.64 | 0.046 |
| Kurtosis | 114.20 | 7.38 | 126.586 |
| Skewness | 8.53 | 2.95 | 8.877 |
| Min | 0.00 | 207.17 | 0.000 |
| Max | 480.00 | 1200.07 | 1.000 |

Table 5. NLQ query templates categorized by type and frequency.

| Category | Template | Freq. |
|---|---|---|
| Objects | Where is object X before / after event Y? | 1760 |
| | Where is object X? | 1455 |
| | What did I put in X? | 1264 |
| | How many X's? (quantity question) | 967 |
| | What X did I Y? | 878 |
| | In what location did I see object X? | 922 |
| | What X is Y? | 853 |
| | State of an object | 569 |
| | Where is my object X? | 261 |
| Place | Where did I put X? | 1676 |
| People | Who did I interact with when I did activity X? | 299 |
| | Who did I talk to in location X? | 224 |
| | When did I interact with person with role X? | 103 |

Each query in the dataset follows one of 13 predefined templates, used to annotate temporal segments where relevant actions or objects occur. However, these templates are unevenly distributed. As shown in Table 5, object-centric queries (e.g., "Where is the cup placed?") are much more frequent than those involving people or actions. This imbalance may bias model predictions toward overrepresented templates [6]. Additional visualizations and analyses—such as template histograms and query-type breakdowns—are available in the codebase linked in the abstract.

#### 4.1.2 Model Performance Across Configurations

We evaluate all eight model variants by systematically combining video features (EgoVLP or Omnivore), text encoders (BERT or GloVe), and model architectures (VSLBase or VSLNet). Performance is assessed using Rank@$k$@IoU=0.3/0.5 and mean IoU (mIoU), as reported in Tables 6–9.

*1) Omnivore.* Using Omnivore FP16 features, VSLNet consistently outperforms VSLBase, demonstrating the benefit of its Query-Guided Highlighting (QGH) module. In both models, BERT leads to better results than GloVe across all metrics (Tables 8, 9).

*2) EgoVLP.* With EgoVLP FP16 features, a similar performance pattern is observed: models using BERT embeddings consistently outperform those using GloVe, as shown in Tables 6 and 7. The highest overall performance is achieved by the **VSLNet (EgoVLP + BERT)** configuration, which attains an mIoU of 6.65, Rank1@0.5 of 5.16, and Rank3@0.5 of 9.09.

Table 6. EgoVLP + VSLBase performance with BERT vs GloVe (last checkpoint)

| Embedding | Rank1@0.3 | Rank1@0.5 | Rank3@0.5 | mIoU |
|---|---|---|---|---|
| **BERT** | **6.12** | **3.87** | **6.50** | **4.98** |
| GloVe | 4.78 | 2.97 | 5.21 | 3.71 |

Table 7. EgoVLP + VSLNet performance with BERT vs GloVe (last checkpoint)

| Embedding | Rank1@0.3 | Rank1@0.5 | Rank3@0.5 | mIoU |
|---|---|---|---|---|
| **BERT** | **8.57** | **5.16** | **9.09** | **6.65** |
| GloVe | 5.24 | 3.28 | 6.04 | 4.32 |

Table 8. Omnivore + VSLBase performance with BERT vs GloVe (last checkpoint)

| Embedding | Rank1@0.3 | Rank1@0.5 | Rank3@0.5 | mIoU |
|---|---|---|---|---|
| **BERT** | **5.50** | **3.33** | **6.09** | **4.65** |
| GloVe | 3.51 | 1.81 | 3.77 | 3.05 |

Table 9. Omnivore + VSLNet performance with BERT vs GloVe (last checkpoint)

| Embedding | Rank1@0.3 | Rank1@0.5 | Rank3@0.5 | mIoU |
|---|---|---|---|---|
| **BERT** | **6.43** | **3.74** | **6.38** | **4.96** |
| GloVe | 4.21 | 2.27 | 4.49 | 3.52 |

Overall, our best-performing model—**VSLNet (EgoVLP + BERT)**—achieved the highest accuracy, highlighting the effectiveness of combining domain-aligned visual encoders with advanced temporal and language modeling.

#### 4.1.3 Comparison with Official Ego4D Baseline

We compare our best model—VSLNet with EgoVLP visual features and BERT embeddings—with the official Ego4D baseline [6], which uses VSLNet with SlowFast [4] features and BERT.

As shown in Table 10, our model outperforms the baseline across all retrieval metrics. Rank@1@0.3 improves from 5.47 to 8.57, and Rank@5@0.5 from 6.57 to 11.20. These gains hold for both test and validation sets, demonstrating the effectiveness of EgoVLP's domain-specific features and BERT's contextualized embeddings for egocentric video localization.

Table 10. Comparison of our best models vs. official Ego4D baseline.

| Model | R@1@0.3 | R@5@0.3 | R@1@0.5 | R@5@0.5 | mIoU |
|---|---|---|---|---|---|
| VSLNet (EgoVLP + BERT) | **8.57** | **16.78** | **5.16** | **11.20** | **6.65** |
| VSLNet (Omnivore + BERT) | 6.43 | 12.67 | 3.74 | 8.03 | 4.96 |
| VSLNet (SlowFast + BERT Test) | 5.47 | 11.21 | 2.80 | 6.57 | – |
| VSLNet (SlowFast + BERT Val) | 5.45 | 10.74 | 3.12 | 6.63 | – |

## 4.2. Textual Answer Generation from VSLNet Segments

### 4.2.1 Lightweight prompt engineering results

As shown in Table 12, The adaptive prompt, featuring task-specific instructions (such as counting, color, or location queries), results in significant improvements in lexical alignment metrics (BLEU and ROUGE) and enhances semantic fidelity (BERTScore F1 and Recall). Notably, a minor decline in **BERTScore Precision** is observed, possibly due to the model producing longer or more exploratory responses that better match the reference in meaning (Recall), although they slightly diverge in token-level precision. These findings highlight the effectiveness of lightweight prompt engineering in facilitating open-ended response generation for vision-language tasks.

Table 12. Impact of prompt engineering on evaluation metrics

| Metric | Variant | Basic Prompt | Adaptive Prompt |
|---|---|---|---|
| BLEU | BLEU-1 | 0.1204 | **0.2558** |
| | BLEU-2 | 0.0221 | **0.1607** |
| | BLEU-3 | 0.0087 | **0.0901** |
| | BLEU-4 | 0.0043 | **0.0333** |
| ROUGE | ROUGE-1 | 0.1899 | **0.2825** |
| | ROUGE-2 | 0.0075 | **0.1282** |
| | ROUGE-L | 0.1900 | **0.2808** |
| | ROUGE-Lsum | 0.1922 | **0.2801** |
| BERTScore | Precision | **0.9006** | 0.8964 |
| | Recall | 0.8620 | **0.8851** |
| | F1 Score | 0.8804 | **0.8904** |

### 4.2.2 Results of comparing Video-LLaVA with Commercial Vision-Language Model: Performance and Accessibility

When evaluating the language models presented, a clear trade-off emerges between state-of-the-art performance and accessibility. This comparison is framed by analyzing the evaluation metrics from Table 11 alongside the cost and deployment model outlined in Table 13.

From the analysis of Table 11, it is evident that **G2.5P** (Gemini 2.5 Pro) and **G1.5P** (Gemini 1.5 Pro) achieve the highest scores across the majority of the lexical and semantic evaluation metrics. Specifically, **G1.5P** obtains the top scores in **BLEU-3 (0.1214)** and **BLEU-4 (0.0524)** and **G2.5P** has a better **BLEU-1 (0.3256)** and **BLEU-2 (0.2058)**. For the ROUGE metrics, **G1.5P** leads with the best **R-1 (0.3449)**, **R-L (0.3439)**, and **R-Lsum (0.3456)** scores, while **G2.5P** has the highest **R-2 (0.1442)**. In terms of semantic similarity measured by BERTScore, the open-source model **Video-LLaVA** demonstrates exceptional performance. It records the highest **Precision (0.8964)**, and a very competitive **F1 Score (0.8904)**. While its lexical scores (BLEU and ROUGE) are generally lower than the top-performing Gemini models, its semantic understanding capabilities are clearly state-of-the-art. Notably, its **F1 Score** surpasses that of GPT-4o, Gemini 1.5 Flash, and Gemini 2.5 Flash.

Table 13. API costs per 1M tokens (June 18, 2025; prompts $\leq$ 200k) from Google Gemini/OpenAI ChatGPT

| Model | API Provider | Input (USD) | Output (USD) |
|---|---|---|---|
| Gemini 1.5 Flash | Google Vertex AI | 0.075 | 0.30 |
| Gemini 1.5 Pro | Google Vertex AI | 1.25 | 5.00 |
| Gemini 2.5 Flash | Google Vertex AI | 0.30 | 2.50 |
| Gemini 2.5 Pro | Google Vertex AI | 1.25 | **10.00** |
| GPT-4o | OpenAI API | **2.50** | **10.00** |

Table 13 provides a cost breakdown per one million tokens for the different models. A significant advantage of **Video-LLaVA** is its open-source nature, making it completely **free** to use. This presents a stark contrast to the commercial models. For the Gemini series, the cost varies. **Gemini 1.5 Flash** is the most affordable at $0.075 for input and $0.30 for output. **Gemini 1.5 Pro** and **Gemini 2.5 Pro** have incrementally higher costs, with the latter costing $1.25 for input and $10 for output. **GPT-4o** is the most expensive model listed, with an input cost of $2.50 and an output cost of $10.00 per million tokens.

This cost efficiency, combined with its strong semantic performance, establishes **Video-LLaVA** as a highly valuable and economical alternative for vision-language tasks. It is particularly well-suited for research, development, and deployment scenarios where budget constraints or the need for **offline capabilities** are primary considerations. Moreover, While commercial APIs like Gemini and GPT-4o offer peak performance and operational simplicity for a direct monetary cost, they trade control and incur ongoing expenses. Conversely, the primary challenge of using a free, local model like Video-LLaVA lies in its significant non-monetary costs. This "problem of locality" shifts the burden to the user, demanding substantial investment in powerful hardware (e.g., high-end GPUs) and the technical expertise needed for setup, maintenance, and scaling. Therefore, the choice is not simply free versus paid, but a **strategic**

Table 11. Evaluation comparison of Video-LLaVA and commercial MLLMs on the Ego4D NLQ task.

| Metric | Variant | LLaVA | G1.5F | G1.5P | G2.5F | G2.5P | GPT-4o |
|---|---|---|---|---|---|---|---|
| BLEU | BLEU-1 | 0.2558 | 0.2684 | 0.3115 | 0.2889 | **0.3256** | 0.2785 |
| | BLEU-2 | 0.1607 | 0.1677 | 0.1992 | 0.1741 | **0.2058** | 0.1691 |
| | BLEU-3 | 0.0901 | 0.1023 | **0.1214** | 0.0995 | 0.1191 | 0.0955 |
| | BLEU-4 | 0.0333 | 0.0419 | **0.0524** | 0.0449 | 0.0507 | 0.0434 |
| ROUGE | R-1 | 0.2825 | 0.2852 | **0.3449** | 0.3097 | 0.3420 | 0.3073 |
| | R-2 | 0.1280 | 0.1047 | 0.1373 | 0.1065 | **0.1442** | 0.1047 |
| | R-L | 0.2799 | 0.2819 | **0.3439** | 0.3080 | 0.3359 | 0.3030 |
| | R-Lsum | 0.2800 | 0.2798 | **0.3456** | 0.3062 | 0.3374 | 0.3031 |
| BERTScore | Precision | **0.8964** | 0.8871 | 0.8906 | 0.8859 | 0.8938 | 0.8824 |
| | Recall | 0.8851 | 0.8923 | **0.8996** | 0.8910 | 0.8975 | 0.8952 |
| | F1 Score | 0.8904 | 0.8893 | 0.8946 | 0.8880 | **0.8952** | 0.8883 |

**decision** between paying for convenient, managed performance or investing in the hardware and skills required to run a powerful model independently.

## 5. Conclusion

To accurately pinpoint specific moments in egocentric videos, our research shows that a combination of powerful language models, specialized visual features, and precise temporal reasoning is crucial. BERT embeddings consistently outperformed GloVe, highlighting the advantage of understanding words in context. The VSLNet model's Query-Guided Highlighting (QGH) module also significantly improved temporal predictions by focusing attention based on the query. Furthermore, visual models trained with egocentric data, like EgoVLP, yielded better results than general-purpose models such as Omnivore. The top performance was achieved by integrating EgoVLP, VSLNet, and BERT, surpassing the Ego4D SlowFast baseline.

Regarding answer generation, while commercial models like Gemini 2.5 Pro are highly accurate, Video-LLaVA stands out as a strong open-source option. It offers comparable semantic accuracy without inference costs, making it ideal for academic and research projects where cost-effectiveness is important. For situations where small improvements in accuracy don't justify higher infrastructure and licensing expenses, Video-LLaVA is a practical and efficient alternative.

## References


[1] Dima et al. Damen. Rescaling egocentric vision: Epic-kitchens-100. *IJCV*, 2022.

[2] Jia et al. Deng. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[3] Jacob et al. Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

[4] Christoph et al. Feichtenhofer. Slowfast networks for video recognition. In *ICCV*, 2019.

[5] Rohit et al. Girdhar. Omnivore: A single model for many visual modalities. In *CVPR*, 2022.

[6] Kristen et al. Grauman. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, 2022.

[7] Will et al. Kay. The kinetics human action video dataset. In *CVPR*, 2017.

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[9] Ranjay et al. Krishna. Dense-captioning events in videos. In *ICCV*, 2017.

[10] Kevin et al. Lin. Egocentric video-language pretraining. In *NeurIPS*, 2022.

[11] Xiang et al. Lin. Video-llava: Learning united visual representation for large language models via video instruction tuning. *arXiv:2311.16452*, 2023.

[12] Haotian et al. Liu. Video-llava: Learning visual representations from instructional videos with multimodal llms. *arXiv:2403.16447*, 2024.

[13] Moaz et al. Maaz. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv:2306.05424*, 2023.

[14] Jeffrey et al. Pennington. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[15] Gunnar A et al. Sigurdsson. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016.

[16] Nitish et al. Srivastava. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[17] Hugo et al. Touvron. Llama: Open and efficient foundation language models, 2023.

[18] Hao et al. Zhang. Span-based localizing network for natural language video localization. In *ACL*, 2020.

[19] Zhiyang et al. Zhu. Video-chatcaptioner: Towards detailed video captioning via chat-like video question answering. *arXiv:2402.00743*, 2024.