Natural Language Queries in Egocentric Videos

Teaching Assistants:
Simone Alberto Peirone, Gabriele Goletto

Link to the slides: https://bit.ly/4aaWCK6
Link to this document: https://bit.ly/49UOmhK

📌 IMPORTANT: For assistance during the development of this project, join the specific slack channel. We will organize weekly meetings until the end of the course to answer your questions and doubts.

CHANGES:

- **23/04/2024**: the network should be trained using egovlp and omnivore features only. We removed the slowfast features to reduce training times. Added an indication to use version 1 of the annotations/features.
- **20/05/2024**: new version of the EgoVLP features with a flat directory structure (https://drive.google.com/file/d/1U318S34jw3uNnsURJ1T40YwsSuK5_-RJ/view?usp=share_link)

Project Overview

Figure 1: Some frames of egocentric videos taken from Ego4D [4], a massive-scale egocentric dataset of daily life activity

Egocentric Vision captures human interactions from a privileged viewpoint, thanks to cameras mounted directly on the head of the user performing the actions. The release of large-scale datasets, such as EPIC-KITCHENS [Damen et al] and Ego4D [Grauman et al], has encouraged the research community to explore different opportunities for learning from egocentric videos, from the more traditional action

recognition and anticipation tasks, to the use of natural language queries for video understanding.
A typical way for humans to understand what is happening in a video is to ask themselves questions under the form of natural language queries, like "*Where was object <x> last seen?*" or "*Which objects did the user interact with?*". Thus, the goal becomes to identify the segment of the video that allows the query to be answered. The use of natural language allows greater flexibility in the kinds of queries that can be made.

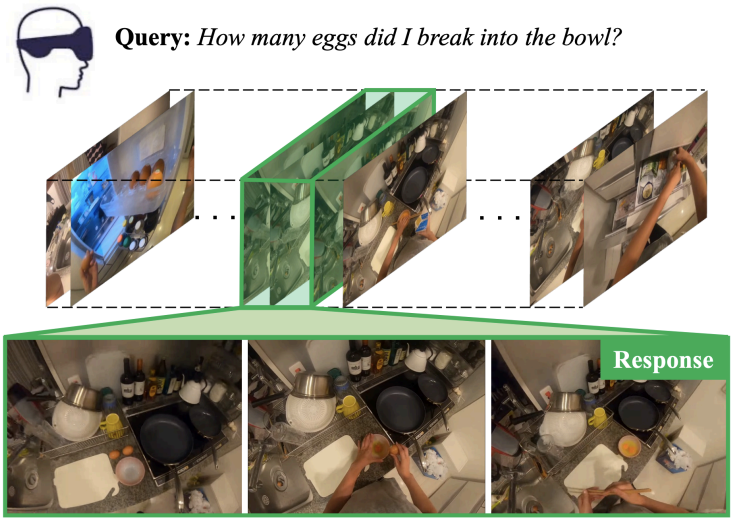


Figure 2: An example of a query on the content of the video, expressed in natural language.

The NLQ benchmark

The Ego4D NLQ benchmark [Grauman et al] is defined as a temporal segment prediction task in which, given a video clip and a query expressed in natural language, the model is asked to predict the temporal segment where the answer is visible or deducible. The queries can be related to objects, places, people, and performed activities. They are defined based on a set of 13 templates, as shown in the figure below.

Category	Template
Objects	Where is object X before / after event Y? Where is object X? What did I put in X? How many X's? (quantity question) What X did I Y? In what location did I see object X ? What X is Y? State of an object Where is my object X?
Place	Where did I put X?
People	Who did I interact with when I did activity X? Who did I talk to in location X? When did I interact with person with role X?

Figure 3: Query templates from the Ego4D NLQ benchmark.

The task can be addressed as a video segment matching problem, i.e. given a query we would like to find the best matching segment [Gao et al], or as a regression problem that directly outputs the boundaries of the relevant video segment [Zhang et al].
NLQ comes with two main challenges. First of all, videos may be of arbitrary length which poses the question of how to build models that can reason over long temporal ranges. The second challenge is related to the integration of the video and text modalities. Performance is evaluated in terms of top-k recall at a certain temporal Intersection over Union (tIoU) threshold.

Project steps

1. IMPORTANT: to access the Ego4d dataset you need to sign the license as described here. It may take a few days to obtain the credentials so make sure to complete this step as soon as possible.
2. Read all the materials provided in order to get familiar with the egocentric vision and natural language queries literature.
3. Get yourself familiar with the Ego4D Natural Language Queries (NLQ) annotations and try to explore the dataset through plots. For example, you may look at the template distribution, the average duration of the annotated clips and the corresponding query/answer temporal segments, or how queries are distributed across different scenarios.
4. Download the pre-extracted Omnivore and EgoVLP features and train VSLBase and VSLNet on the Ego4D Natural Language Queries benchmark and compare the results obtained with different features.
5. Implement an extension (see below).

6. Optional: submit your results to the official Ego4D NLQ challenge 🏆. More details here. Leaderboard closes on May 31, 2024.

Step 2: Literature review

Before starting the project, you need to get familiar with the egocentric vision and video understanding literature. We provide a short list of documents that we consider essential for the development of this project. You do not need to fully understand each paper at this stage, but make sure to grasp the main concepts.

Step 3: Dataset and annotations

Get familiar with the Ego4D (sign the license and get access to the dataset) and its Natural Language Queries (NLQ) annotations ("`nlq_{split}.json`"). The benchmark is composed of a set of ~19k annotated queries from around 227 hours of videos. In this initial step, you should analyze the annotations, for example plotting the queries distribution according to their template, their average duration of the input clips, the distribution of the answer segments along the videos. Additionally, you can use the `ego4d.json` annotations to get the scenarios, for example *cooking* or *commuting*, associated with each video and see how these correlate with the NLQ annotations.
NOTE: you should use version 1 of the annotations and features (check the `--version` flag of the ego4d cli)

Example of NLQ annotation:


```
{
 "video_uid":"d250521e-5197-44aa-8baa-2f42b24444d2",
 "clips":[
   {
     "clip_uid":"fae92e70-88aa-4b77-b41a-5879b74c804c",
     "video_start_sec":0.0210286,
     "video_end_sec":480.0210286,
     "video_start_frame":1,
     "video_end_frame":14401,
     "clip_start_sec":0,
     "clip_end_sec":480.0,
     "clip_start_frame":0,
     "clip_end_frame":14400,
     "source_clip_uid":"51e04dae-3ad0-48c1-b94b-c3ba0edaa99e",
     "annotations":[
       {
         "language_queries":[
           {
             "clip_start_sec":0.0,
             "clip_end_sec":43.6657,
             "video_start_sec":0.0210286,
             "video_end_sec":43.6867286,
             "video_start_frame":1,
```

```
            "video_end_frame":1311,
            "template":"Objects: How many X's? (quantity question)",
            "query":"How many frying pans can i see on the shelf?",
            "slot_x":"frying pans",
            "verb_x":"[verb_not_applicable]",
            "slot_y":"i See on the shelf",
            "verb_y":"see",
            "raw_tags":[
              "Objects: How many X's? (quantity question)",
              "How many frying pans can i see on the shelf?",
              "frying pans",
              "[verb_not_applicable]",
              "i See on the shelf",
              "see"
            ]
          }
        ]
      }
    ]
  }
 ]
}
```


ATTENTION: in ego4d videos are divided into shorter segments called clips. Annotations provide the start and end timestamps of each sample with respect to both the clips ("`clip_start_time`" and "`clip_end_time`") as well as with respect to the full videos ("`video_start_time`" and "`video_end_time`").

Step 4: Training VSLNet

VSLNet is a video-language architecture designed for the NLQ task [Zhang et al]. The paper proposes two variants of this architecture: VSLBase and VSLNet.

VSLBase is composed of two branches that extract visual and textual features from the input video and the corresponding textual query (blue and red blocks in the figure below), followed by a shared encoder block that maps the features of both modalities in a shared features space (yellow blocks). Then, the Context-Query Attention (CQA) module (violet block) computes the similarity between the visual and textual features using an attention-like mechanism. Finally, two unidirectional LSTMs are used to regress the temporal boundaries of the answer (green block).

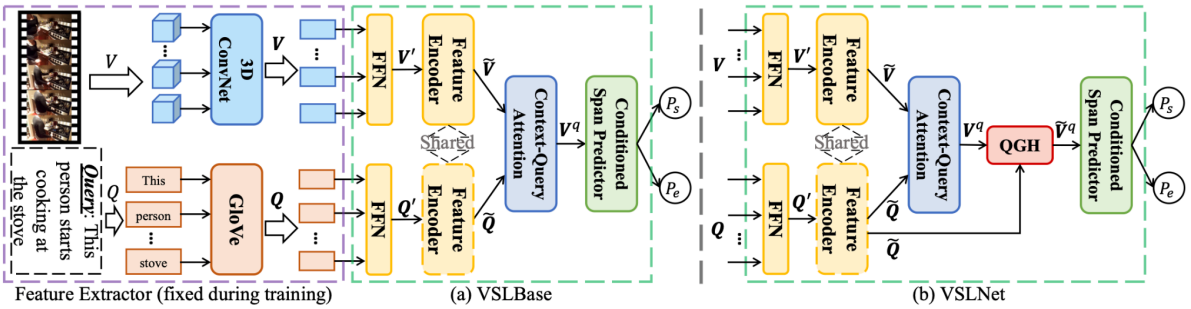


Figure 4: Architecture of the VSLBase and VSLNet models.

VSLNet extends the base architecture by adding an additional module, the Query Guided Highlighter, that encourages the modules to better focus on subtle differences between video frames.

Training from frozen features

Training models end-to-end on video datasets is often impractical due to the large computational budgets required. As an example, training EgoVLP [Lin et al] requires ~1536 hours on Nvidia A100 GPUs. A common strategy to circumvent this problem is to leverage pre-trained models and fine-tune them for the specific downstream task we want to solve. Among these pre-trained models, we focus on Omnivore [Girdhar er at] and EgoVLP [Lin et al]. EgoVLP is a video-language contrastive model pre-trained on a subset of Ego4D using the weak-supervision of free-form textual narrations.

Substeps

1. You are provided with pre-extracted features (for the official ones extracted with Omnivore follow here downloading omnivore_video_swinl_fp16 features, for the EgoVLP ones download from here) for the subset of videos in the NLQ benchmark. These features were extracted using Omnivore [Girdhar et al] and EgoVLP [Lin et al] using temporal windows of 32 frames and temporal stride 16, in line with the *official* Ego4D pre-extracted features.
2. Start from the official quick-start Google Colab notebook for the NLQ task and train a VSLBase and a VSLNet model using Omnivore and EgoVLP features.
3. Compare the obtained results with the official baseline results provided in the Ego4D paper and trained on the SlowFast features (table below).
4. Implement one variation of the VSLNet architecture and compare the performance with the baseline. Some examples include:
    a. Replace the text encoder (GloVe) with a different encoder, e.g. BERT.
    b. Use different (*non shared*) encoders for the two modalities (video and text).
    c. Try the 2D-TAN [Zhang et al] model also implemented in the official GitHub repository for NLQ episodic memory. Compare the results with

the ones obtained using VSLNet (with the different features). What are the differences among the two models?

	Baseline	IoU=0.3 (%) r@1	IoU=0.3 (%) r@5	IoU=0.5 (%) r@1	IoU=0.5 (%) r@5
Val	2D-TAN [236]	5.04	**12.89**	2.02	5.88
	VSLNet [235]	**5.45**	10.74	**3.12**	**6.63**
Test	2D-TAN [236]	**5.80**	**13.90**	2.34	5.96
	−visual	2.29	6.77	1.32	3.46
	−text	3.46	10.13	1.78	4.38
	VSLNet [235]	5.47	11.21	**2.80**	**6.57**
	−visual	1.80	5.44	0.90	2.45
	−text	3.05	7.39	1.45	4.12

Table 1: Results on the validation and test splits for the Ego4D NLQ baseline.

Step 4: Extensions

Implement one of the following extensions. You may also propose a different extension of the project, but you MUST discuss your ideas in advance with the teaching assistants.

Proposal 1: Automatic Queries Generation using LLMs

Although the NLQ benchmark is quite extensive (~227 hours), it covers a limited portion of the full Ego4D dataset (~3600 hours). However, all the videos from the dataset are annotated with textual narrations describing what the user is doing and their corresponding timestamps. We could use these narrations to generate NLQ-like question-answer pairs and build an augmented version of the NLQ dataset. This new dataset could be used to pre-train VSLNet, before fine-tuning it to the real NLQ data, providing more variegate query examples.
The goal of this extension is to use a Large Language Model (e.g. llama or Gemma) to generate query-answer pairs from a set of randomly sampled consecutive narrations. The ground truth for the answer is the extreme timestamps corresponding to the narrations.

Example (using Gemma-2b-it):

Prompt:
Given the following narrations describing the actions of a person, generate a set of simple queries (one per line) that could be answered by looking at the video segments corresponding to these narrations:

```
the person waters the plants using a water gun
the person touches the plant with the left hand
the person waters the plants using a water gun
the person stares at the plant
the person touches the soil with the left hand
```

Response:

- 1. What is the person doing with the water gun?
- 2. Which hand is touching the plant?
- 3. What is the person doing while watering the plants?
- 4. What is the person looking at?
- 5. Which hand is touching the soil?

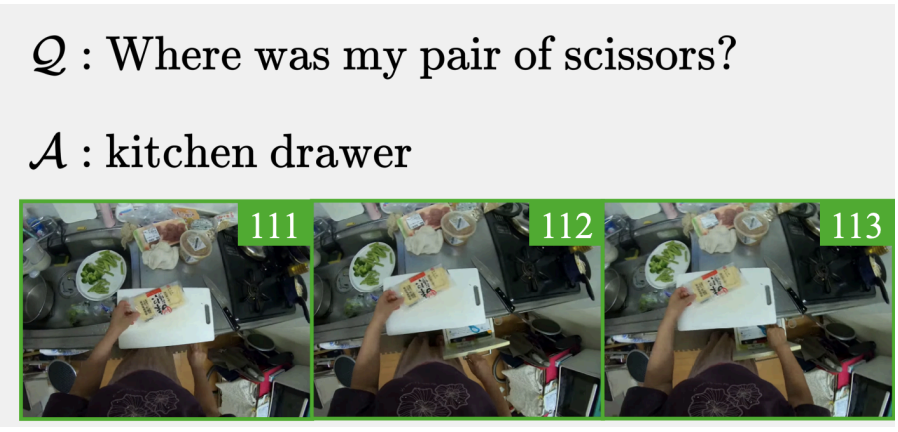
Proposal 2: From video interval to a textual answer

The NLQ task outputs the time interval during which the input query is answered, but it's essential to note that to have the actual answer a person should then watch that interval. Egocentric videos, which capture daily life activities, tend to be long and unstructured. This makes it challenging to directly extract textual answers from the entire video. Particularly, the high number of video tokens makes it computationally expensive for Video Language Models (VLMs) to process an entire recording for an answer which is present just in a subset of it. Additionally, aggregating information to make it more computationally feasible might lead to a loss of details.

This extension aims to address these challenges by leveraging the NLQ method as the first step in a video question answering pipeline. It involves identifying the relevant segments within long videos that contain useful information and then using these segments as input for a VLM to have textual answers.

Particularly the substeps are the following ones:

1. Select 50 NLQ queries where the correct video segment is retrieved by one of the models in step 3. Manually annotate the correct textual ground truth associated with each query.
2. Download the videos related to the previously retrieved video segments one by one, and extract the parts of interest using ffmpeg.
3. Adopt a videoQA model to feed the previously extracted parts along with the input query, requesting an answer. For example, Video-LLaVA [Lin et al] can operate on Google Colab within the given computational budget.
4. Select an interesting subset of qualitative results among those obtained and showcase them together with some quantitative metrics (e.g. ROUGE or BLEU)



References

1. Grauman, Kristen, et al. "Ego4d: Around the world in 3,000 hours of egocentric video." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.
2. Damen, Dima, et al. "Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100." *International Journal of Computer Vision* (2022): 1-23.
3. Gao, Jiyang, et al. "Tall: Temporal activity localization via language query." Proceedings of the IEEE international conference on computer vision. 2017.
4. Zhang, Hao, et al. "Span-based Localizing Network for Natural Language Video Localization." *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.
5. Feichtenhofer, Christoph, et al. "Slowfast networks for video recognition." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.
6. Lin, Kevin Qinghong, et al. "Egocentric video-language pretraining." *Advances in Neural Information Processing Systems* 35 (2022): 7575-7586.
7. Zhang, Songyang, et al. "Learning 2d temporal adjacent networks for moment localization with natural language." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. No. 07. 2020.
8. Girdhar, R., Singh, M., Ravi, N., Van Der Maaten, L., Joulin, A., & Misra, I. (2022). Omnivore: A single model for many visual modalities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 16102-16112).
9. Lin, B., Zhu, B., Ye, Y., Ning, M., Jin, P., & Yuan, L. (2023). Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*.