Natural Language Queries in Egocentric Videos

Amirmohammad Goudarzi
DAUIN
Politecnico di Torino
Turin, Italy
amirmohammad.goudarzi@studenti.polito.it

Johan Lindell
DAUIN
Politecnico di Torino
Turin, Italy
johan.lindell@studenti.polito.it

Iiro Moisander
DAUIN
Politecnico di Torino
Turin, Italy
iiro.moisander@studenti.polito.it

***Abstract*—Egocentric vision is a rapidly growing field with significant applications in wearable devices and robotics. This project aims to enhance egocentric action recognition models. We trained two previously existing models using various hyperparameters, video features, and text encoders for a natural language video localization tasks. Additionally, we extended the models to generate natural language responses based on video segments and corresponding queries. Our results show improved model performance and provide insights into future research directions.**

I. Introduction

Egocentric vision is a subset of computer vision which deals with analyzing and processing videos recorded in a first-person point of view. This perspective makes this field highly valuable for applications in wearable devices, autonomous systems and assistive technologies.

Action recognition within egocentric videos is a crucial task that enables intelligent systems to understand and anticipate human actions. Effective action recognition can significantly enhance the functionality of applications such as augmented reality, personal assistants, and surveillance systems by providing contextual understanding and real-time responses. Despite significant advancements in computer vision, egocentric action recognition remains a challenging problem due to the complexity and variability of human actions observed from a first-person perspective.

The problem addressed in our work is understanding egocentric videos using natural language. We use the Ego4D NLQ benchmark [1], where the task is to train a model to predict the video segment containing the answer to a given natural language query. This problem can be approached as a classification problem to find the best matching segment [2], or as a regression problem [3] to determine the optimal time boundaries of the video.

In our work, we trained VSLBase and VSLNet models [3] on different sets of hyperparameters. Due to limited computational budget, we used pre-extracted video features from Omnivore [4] and EgoVLP [5] instead of actual videos for this task. Additionally, we experimented with two different text encoders, GloVe [6] and BERT [7]. We further extended our work to enable the model to output a natural language answer based on the corresponding video segment.

The code for our project is available at *https://github.com/johan-lindell/VSL-egocentric*.

II. Background material

Natural Language Video Localization. The goal of this task is to find the relevant segment of the video that semantically corresponds to the given query. NLVL was introduced by [2] and [8]. The early works approached this problem using a multi-modal architecture to capture the interactions between natural language and video and subsequently find the best matching video moment ([2], [8]–[14]). These solutions were highly sensitive to negative samples and required dense sampling of candidate moments, leading to inefficiency and limited flexibility. There have been approaches to overcome these drawbacks. Some recent works regress the timestamps of the target video segment directly ([15], [16]), while some others that treat this task as a sequence decision making problem and use reinforcement learning approaches ([17], [18]).

Span-Based Question Answering. The task of span-based question answering has been studied extensively in the past, primarily within the field of natural language processing [19]–[21]. We extended the work of [3], which sees the NLVL task in the span-based QA framework. In this approach, the input video is treated as a text, and the goal is to locate the temporal locations of the answer span to the given query.

Dataset and Benchmark. The release of large-scale datasets such as EPIC-KITCHENS [22] and Ego4D [1] has enabled researchers to explore the potential of egocentric vision. In this work we use the Ego4D dataset, which provides an extensive collection of egocentric videos captured from a first-person perspective. The dataset includes a diverse range of daily-life activities performed in different environments and scenarios (household, outdoor, workplace, leisure, etc.), offering a rich source of data for training and evaluating egocentric vision models [1]. Ego4D also offers several tasks for egocentric video understanding. We addressed the Episodic Memory - NLQ challenge, the goal of which is to localize the temporal window from the video where the answer to the question is evident [1].

III. Methodology

In this section we briefly present the VSLBase and VSLNet architectures [3] and then outline how we can expand the solution to output natural language predictions. As discussed in the introduction, multiple versions were trained, equipping the models with varying compositions of encoders and video

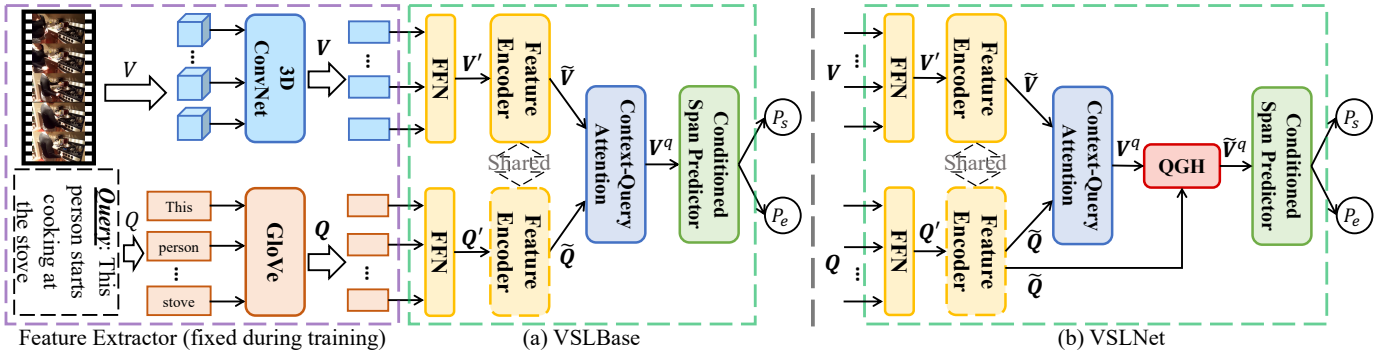

Fig. 1. An overview the architectures of VSLBase and VSLNet [3]. The feature extractor was not used, instead pre-extracted features were employed.

feature sets. In following subsections we will elaborate on the key components of the trained models and the extension. In the case of VSLBase and VSLNet especially, we aim to explain the workings of the key parts of the architecture in a concise yet thorough manner.

A. VSLBase

As presented by [3], the VSLBase architecture consists of feature extractors, feature encoders, context-query attention and a conditioned span predictor.

We used pre-extracted features, utilizing Omnivore [4] and EgoVLP [5] features depending on the model trained. Word embeddings were obtained using mainly BERT [7] architecture although GloVe [6] was for reference tested in the case of two separate runs.

Next we describe the parts consistently same throughout the variety of VSLBase models trained. After uploading the visual features $\mathbf{V} = \{\mathbf{v}_i\}_{i=1}^{n} \in \mathbb{R}^{n \times d_v}$ to the machine and creation of the word embeddings $\mathbf{Q} = \{\mathbf{q}_j\}_{j=1}^{m} \in \mathbb{R}^{m \times d_q}$, these vectors are projected into a common dimension by feed-forward linear network consisting of two layers. The transformation projects the vectors from their initial dimensions d_v and d_q to common dimension d. In Figure 1 the resulting vectors are denoted as $\mathbf{V}'(\in \mathbb{R}^{n \times d})$ and $\mathbf{Q}'(\in \mathbb{R}^{m \times d})$.

The VSLBase context-query attention only uses one encoder block [23] which consists of four convolution layers, followed by a multi-head attention layer [24] and a feed-forward layer. Layer normalization [25] and residual connections [26] are applied to each layer. In the model the trained parameters are shared by visual features and word embeddings.

The context-query attention part itself is constructed as described in [3] which is based on [21], [23], [27]. Between each visual feature vector and word embeddings vector, a similarity score $S \in \mathbb{R}^{n \times m}$ is calculated. Then row-wise and column-wise normalization is applied to the similarity matrix S by using SoftMax.

Using the similarity matrix, context-to-query attention weights A are obtained by applying a row-wise normalized similarity matrix, resulting in $A = S_r \cdot \tilde{Q} \in \mathbb{R}^{n \times d}$, where $\tilde{Q}$ is the encoded query and S_r denotes the row-wise normalized similarity scores. Query-to-context attention weights B are obtained by applying a column-wise normalized similarity matrix, resulting in $B = S_c \cdot S^T \cdot \tilde{V} \in \mathbb{R}^{n \times d}$, where $\tilde{V}$ is the encoded visual features and S_c denotes the column-wise normalized similarity scores. Finally the combined context-query attention $V_q \in \mathbb{R}^{n \times d}$ is obtained as $V_q = \text{FFN}([\tilde{V}; A; \tilde{V} \odot A; \tilde{V} \odot B])$, where FFN denotes a feed-forward network, [;] indicates concatenation, and $\odot$ represents element-wise multiplication

Next we describe the span predictor, last piece of the architecture, based on the explanation in [3]. It employs two unidirectional LSTMs and two feed-forward layers. The LSTMs are stacked so that the end boundary predictor is conditioned on the start boundary predictor. The hidden states of the LSTMs are computed as follows:

$$h_t^s = \text{UniLSTM}_{\text{start}}(v_t^q, h_{t-1}^s)$$
$$h_t^e = \text{UniLSTM}_{\text{end}}(h_t^s, h_{t-1}^e)$$

where h_t^s and h_t^e denote the hidden states for the start and end boundaries at position t, respectively, and v_t^q represents the visual query feature at position t.

Subsequent to computing the hidden states, the start and end scores are determined using the hidden states and the visual query features. The scores are calculated through the following feed-forward layers:

$$S_t^s = W_s \times [h_t^s; v_t^q] + b_s$$
$$S_t^e = W_e \times [h_t^e; v_t^q] + b_e$$

where S_t^s and S_t^e are the scores for the start and end boundaries at position t. Here, W_s and W_e denote the weight matrices, and b_s and b_e are the biases for the start and end layers, respectively.

The probability distributions P_s and P_e are retrieved by applying a softmax function to S_t^s and S_t^e. Loss $\mathcal{L}_{\text{span}}$ is calculated by applying cross-entropy loss function f_{CE} to both probability distributions: $\mathcal{L}_{\text{span}} = \frac{1}{2}\left[f_{CE}(P_s, Y_s) + f_{CE}(P_e, Y_e)\right]$.

During interference the start and end boundaries are retrieved by maximising the joint probability of the probability distributions $\text{span}(a_s, a_e) = \arg\max_{a_s, a_e} P_s(a_s) P_e(a_e)$ so that $0 \leq a_s \leq a_e \leq n$.

B. VSLNet

The VSLNet model extends VSLBase by incorporating a Query-Guided Highlighting (QGH) module [3]. In this section we explain QGH based on the explanation in paper [3].

QGH is a binary classification module that predicts the confidence a visual feature belongs to foreground or background. It exends the boundaries of the target moment (foreground) in the clip (span(a_s, a_e)) by hyperparameter α so that the resulting length is $L\alpha + L + L\alpha$ where L is the foreground lenght and the two $L\alpha$s represent the extended parts, termed background. As labels QGH has a sequence of 0s and 1s (hereby denoted by Y_h) where 0s represent background and 1s the foreground.

QGH influences the prediction made by the model by highlighting some of the features. This is achieved by calculating $\tilde{V}_q$ containing modified features, which are then passed on to the conditioned span predictor in stead of V_q as we can see in Figure 1b. The highlighting is done by calculating $\tilde{V}_q = \big(\sigma(\text{Conv1D}(\bar{V}_q))\big) \cdot \bar{V}_q$ where $\bar{V}_q = [[v_1^q; h_Q], \ldots, [v_n^q; h_Q]]$ and furthermore v_n^q is the n-th feature in V^q and h_Q a sentence representation of query features $\tilde{Q}$ retrieved with self-attention mechanism [28]. Examining the calculation closer, we can see that $\tilde{V}_q$ is attained by first calculating highlighting scores $S_h = \sigma(\text{Conv1D}(\bar{V}_q)) \in \mathbb{R}^n$. After this highlighted features are achieved by applying the scores to $\bar{V}_q$ by element-wise multiplication.

The loss function of query-guided highlighting is formulated as $\mathcal{L}_{\text{QGH}} = f_{\text{CE}}(S_h, Y_h)$ and in the context of VSLNet added to the span predictor loss $\mathcal{L} = \mathcal{L}_{\text{span}} + \mathcal{L}_{\text{QGH}}$.

C. Extension: Video Interval to Textual Answer

Our problem can be extended to generate a textual output given a query and a video clip. This extension leverages the output from the NLQ task, which predicts the time intervals containing the answers to given natural language queries. The videos can then be trimmed with the predicted time intervals which makes them small an precise enough to be processed by a Large Vision-Language Model (LVLM). A LVLM takes as input a video clip and a natural language query, it then predicts a natural language response to the query based on the features of the video [29].

In our implementation, we selected 50 clips that closely matched the ground truth. We used the model with the best predictions to generate these intervals. The original full-length clips were then cut according to the predicted intervals, and ground truths were manually annotated. For the LVLM in this project, we utilized **video-LLaVA** [29], a relatively lightweight state-of-the-art model that delivers good results compared to other models. Specifically, we used the pretrained weights **LanguageBind/Video-LLaVA-7B-hf** for the LVLM. We then manually compared the predictions to the manually annotated ground truths to evaluate the accuracy of our extension.

IV. Experiments and Results

A. Data Analysis

We present a brief statistical summary of the Ego4D dataset at table IV-A. Here we have outlined key statistics of the query and clip sizes. The relative query size is also included which is calculated as the fraction of query to clip size. The Ego4D dataset consists of clips, natural language queries and query times [1]. Notable is that the query intervals are heavily skewed towards shorter queries, this is apparent by the large positive skewness as showed in table IV-A. Their relative positions in the clip have a similar distribution and skew as the duration's. This indicates that the vast majority of queries are short and in the beginning of the clips.

TABLE I
Summary Statistics for Query Size, Clip Size, and Relative Query Size

	Query Size	**Clip Size**	**Relative Query Size**
Count	11296	11296	11296
Mean	9.67	522.68	0.020
Std	22.83	197.65	0.046
Kurtosis	114.25	7.38	126.64
Skewness	8.53	2.95	8.88
Min	0.00	207.17	0.000
Max	480.00	1200.07	0.999

The Ego4D queries are all based on 13 different question templates applied to the videos and then the time intervals represent the relevant parts of the clips [1]. These templates are not evenly distributed and thus could induce a bias into the model where it gives more accurate predictions to specific types of prompts. As outline in figure 2 we can see that prompts related to where objects are placed are most prevalent while other type of prompts for instance related to people are less represented. Visualisations and more elaborate analysis is available in the codebase.

B. Pre-Extracted Features

In our experiments, we used two sets of pre-trained model features: Omnivore FP16 (version 1) and EgoVLP FP16. We compare the results with the baseline results in [1], where the SlowFast [30] features were used.

Training models end-to-end on video datasets is often impractical due to the large computational budgets required. For instance, training EgoVLP [5] requires approximately 1536 hours on Nvidia A100 GPUs. A common strategy to circumvent this problem is to leverage pre-trained models and fine-tune them for specific downstream tasks. Among these pre-trained models, we focus on Omnivore [4] and EgoVLP [5].

Omnivore FP16 V1: These features are pre-extracted using the model Omnivore (Swin L) with a video head [4], trained on the Kinetics 400 [31] and ImageNet-1K [32] datasets.

EgoVLP FP16: These features are obtained using egocentric video-language pretraining [5]. EgoVLP is a video-language contrastive model pre-trained on a subset of Ego4D.

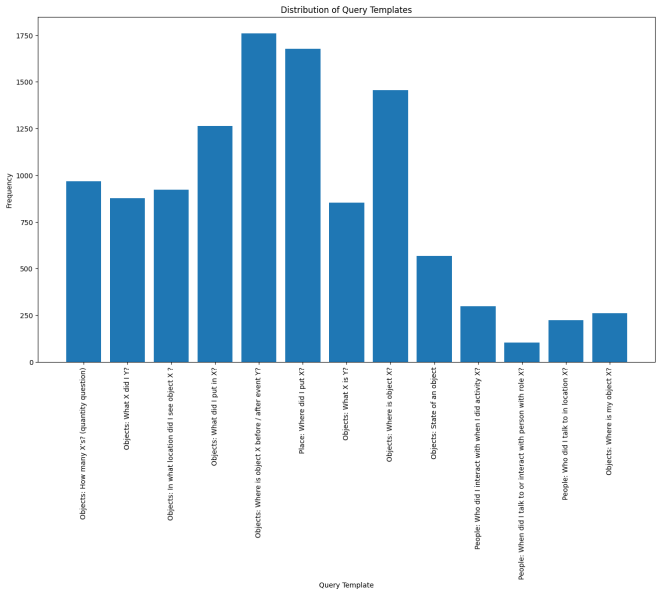

Fig. 2. Distribution of Query Templates

The pre-extracted features for both Omnivore and EgoVLP were used to analyze the subset of videos in the NLQ benchmark, facilitating a comparison with the baseline model in [1], where SlowFast [30] features were employed.

C. Experimental settings

Metrics. We use "R@n, IoU $= \mu$" and "mIoU" as the evaluation metrics, following the original paper [3]. The "R@n, IoU $= \mu$" denotes the percentage of language queries having at least one result whose Intersection over Union (IoU) with ground truth is larger than μ in top-n retrieved moments. "mIoU" is the average IoU over all testing samples. In our experiments, we use $n = 1$ and $\mu \in \{0.3, 0.5, 0.7\}$. In the codebase you can find full statistics of the results using the tensorboard runs, in this report we only use some of them.

Implementation. We employ BERT [7] as the text encoder, showcasing its better performance over GloVe [6]. Following the methodology in the referenced paper, Adam [33] is used as the optimizer. Various initial learning rates are tested, along with a linear decay of the learning rate and a gradient clipping threshold of 1.0. To mitigate overfitting, a dropout rate of 0.2 [34] is applied.

D. Results

1) Omnivore: We trained both the VSLBase and VSLNet models on different sets of hyperparameters. We first started with VSLBase using Omnivore features. Table II shows the results with the starting hyperparameters.

We then trained VSLNet with some of the optimal hyperparameters selected from the previous experience, to avoid a time-consuming computation. Results are shown in table III

2) EgoVLP: As expected, VSLNet shows better performance with respect to VSLBase because of the QGH component. We trained VSLNet on the EgoVLP features as well.

Batch Size	Epochs	Initial LR	IoU=0.3 (%)		IoU=0.5 (%)	
			r@1	r@5	r@1	r@5
32	10	0.001	5.29	11.95	2.92	7.30
32	10	0.0025	5.80	12.18	3.33	7.59
32	40	0.001	5.99	11.54	2.68	6.99
32	40	0.0025	5.63	11.72	3.02	6.89
64	10	0.001	4.98	10.89	2.53	6.61
64	10	0.0025	5.96	**12.83**	3.20	**7.77**
64	40	0.001	5.03	11.64	2.66	7.10
64	40	0.0025	**6.09**	11.69	**3.41**	7.12

TABLE II
VSLBASE TRAINING RESULTS

Batch Size	Epochs	Initial LR	IoU=0.3 (%)		IoU=0.5 (%)	
			r@1	r@5	r@1	r@5
32	40	0.001	6.04	11.93	3.12	7.41
32	40	0.0025	**6.56**	**12.36**	3.43	7.72
64	40	0.001	5.94	12.08	**3.46**	**7.77**
64	40	0.0025	6.07	11.90	3.12	7.18

TABLE III
VLSNET TRAINING RESULTS

The results in Table IV are obtained using the EgoVLP FP16 variant features, and BERT [7] as the text encoder.

Batch Size	Epochs	Initial LR	IoU=0.3 (%)		IoU=0.5 (%)	
			r@1	r@5	r@1	r@5
64	10	0.001	5.34	12.13	3.38	8.10
64	10	0.0025	7.25	15.13	4.75	10.20
64	40	0.001	8.62	16.78	**5.06**	11.30
64	40	0.0025	**8.91**	**17.84**	4.85	**11.64**

TABLE IV
VSLNET EGOVLP

The results show that the EgoVLP video features demonstrate better performance in comparison with the Omnivore features. It is primarily due to EgoVLP's specialized pre-training on egocentric video datasets. EgoVLP is designed to capture the nuances of first-person perspectives, leading to more accurate feature extraction for egocentric video analysis. This specialization results in higher performance metrics compared to Omnivore, which is trained on a broader range of visual modalities and may not capture the specific dynamics of egocentric videos as effectively.

3) Text Encoder: To compare BERT [7] with GloVe [6], we used a single set of optimal hyperparameters (batch size of 64, 40 epochs, and initial learning rate of 0.001) and trained VSLNet on both Omnivore and EgoVLP features using GloVe as the text encoder. The results are available in table V.

Encoder	Features	IoU=0.3 (%)		IoU=0.5 (%)	
		r@1	r@5	r@1	r@5
GloVe	Omnivore	5.16	11.70	2.45	7.20
GloVe	EgoVLP	7.38	14.89	4.47	9.89
BERT	Omnivore	5.94	12.08	3.46	7.77
BERT	EgoVLP	**8.62**	**16.78**	**5.06**	**11.30**

TABLE V
GLOVE

BERT outperforms GloVe mainly because BERT generates dynamic embeddings for each word which change based on the

surrounding text. In contrast, GloVe uses static embeddings, where each word has a single fixed representation regardless of context, reducing the model's capability of capturing important features of the queries.

4) Comparison with the official baseline results: We present a comparison between our work and the experiments conducted in [1] NLQ baselines. The paper used 2D Temporal Adjacent Networks [35] and VSLNet [3] for this task, on SlowFast features [30]. Please see [1] for their implementation details.

We selected two of our best settings for this task. We used VSLNet as the model with BERT as the text encoder and EgoVLP as video features. The 3rd and 4th sets of hyperparameters from Table IV are denoted as 1 and 2, respectively. Furthermore, we also compared the results from Omnivore features using the 2nd and 3rd sets of hyperparameters from Table II, denoted as 3 and 4. The results shown in the table VI showcase that EgoVLP and Omnivore features outperform SlowFast.

Method	IoU=0.3 (%)		IoU=0.5 (%)	
	r@1	r@5	r@1	r@5
2D-TAN	5.04	12.89	2.02	5.88
VSLNet	5.45	10.74	3.12	6.63
1	8.62	16.78	**5.06**	11.30
2	**8.91**	**17.84**	4.85	**11.64**
3	6.56	12.36	3.43	7.72
4	5.94	12.08	3.46	7.77

TABLE VI
COMPARISON WITH [1]

Fig. 3. Prompt: how many dough did i roll?
Prediction: You rolled two doughs. (or two pieces of dough) in the video.

Fig. 4. Prompt: in what aisle did i see cheese balls?
Prediction: You saw cheese balls in the snack aisle. The image shows a grocery store with a variety of snacks on display. There are multiple bags of chips, including Doritos, and a variety of other snacks such as cheese balls

E. Extension

The extension model prediction indicate that the model performed well on clear videos with minimal extraneous information. Specifically, the model gave accurate predictions for clips where the focus was clear and unambiguous, such as in the example of a person rolling dough's (see Figure 3). However, the model struggled with videos containing a high density of information, such as a shelf with many different objects. In these cases, the predicted outputs were often incorrect, such as the misprediction of the snack aisle instead of the freezer aisle for picking up cheese balls (see Figure 4).

Additionally, some clips were very short, making it challenging to analyze the ground truth even for human annotators. For future work, it may be beneficial to filter out such problematic clips to improve overall results.

Our extension implementation and validation is quite naive. More in depth analysis on the performance of the model would need to be conducted for more definitive answers about the proficiency of the model. Improvements like fine tuning the LVLM to egocentric videos could provide possible improvements in the model and is a possible prospect for further research.

V. CONCLUSIONS

In this report, we investigated the task of natural language queries in egocentric videos, aiming to enhance action recognition models and extend their capabilities to generate natural language responses based on video segments. Our approach involved training two existing models, VSLBase and VSLNet, with various hyperparameters, video features, and text encoders. We compared these models against baseline results of the NLQ paper [1] and explored their performance using different sets of pre-extracted video features from Omnivore and EgoVLP.

Our experimental results demonstrate significant improvements in model performance, particularly with the use of EgoVLP features. VSLNet outperformed VSLBase, showcasing the benefits of its Query-Guided Highlighting (QGH) strategy in handling the complexities of egocentric video data. The use of BERT [7] as a text encoder also proved to be more effective than GloVe [6], further enhancing the model's ability to accurately localize relevant video segments in response to natural language queries.

Additionally, we extended our work to enable the models to generate natural language answers based on identified video segments. By manually annotating ground truth clips and leveraging a Large Vision-Language Model (LVLM), we demonstrated the feasibility of this approach especially for videos with a clear focus. Other implementations may be needed for more complex videos with a lot of information where our predictions where inaccurate.

Overall, our findings indicate that the combination of advanced feature extraction techniques, robust model architectures, and powerful text encoders can significantly enhance

the performance of egocentric action recognition models. Future research directions include exploring more sophisticated video-language pretraining methods, improving the efficiency of training processes, and further refining the natural language generation capabilities of these models. This work contributes to the growing field of egocentric vision, paving the way for more intelligent and responsive applications in wearable devices, robotics, and assistive technologies.

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] Gao, Jiyang, Sun, Chen, Yang, Zhenheng, Nevatia, Ram. "Tall: Temporal activity localization via language query." Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[3] Zhang, Hao, Sun, Aixin, Jing, Wei, Zhou, J. Tianyi. "Span-based Localizing Network for Natural Language Video Localization." Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020.

[4] Girdhar, R., Singh, M., Ravi, N., Van Der Maaten, L., Joulin, A., and Misra, I. "Omnivore: A single model for many visual modalities." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, pp. 16102-16112.

[5] Lin, Kevin Qinghong, et al. "Egocentric video-language pretraining." Advances in Neural Information Processing Systems 35 (2022): 7575-7586.

[6] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "GloVe: Global vectors for word representation." Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014.

[7] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, Toutanova, Kristina. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2019.

[8] Hendricks, Lisa Anne, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan C. Russell. "Localizing moments in video with natural language." Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017, pp. 5804-5813.

[9] Hendricks, Lisa Anne, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan C. Russell. "Localizing moments in video with temporal language." Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2018, pp. 1380-1390.

[10] Wu, Aming and Yahong Han. "Multi-modal circulant fusion for video-to-language and backward." Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18. 2018, pp. 1029-1035.

[11] Liu, Meng, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. "Attentive moment retrieval in videos." In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pages 15–24. Association for Computing Machinery, 2018a.

[12] Liu, Meng, Xiang Wang, Liqiang Nie, Qi Tian, Baoquan Chen, and Tat-Seng Chua. "Crossmodal moment localization in videos." Proceedings of the 26th ACM International Conference on Multimedia. 2018b, pp. 843-851.

[13] Xu, Huijuan, Kun He, Bryan A. Plummer, L. Sigal, Stan Sclaroff, and Kate Saenko. "Multilevel language and vision integration for text-to-clip retrieval." Proceedings of the AAAI Conference on Artificial Intelligence. 2019, vol. 33, pp. 9062-9069.

[14] Zhang, Da, Xiyang Dai, Xin Wang, Yuan-Fang Wang, and Larry S. Davis. "MAN: Moment alignment network for natural language moment retrieval via iterative graph adjustment." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2019, pp. 1247-1257.

[15] Yuan, Yitian, Tao Mei, and Wenwu Zhu. "To find where you talk: Temporal sentence localization in video with attention based location regression." Proceedings of the AAAI Conference on Artificial Intelligence. 2019b, vol. 33, pp. 9159-9166.

[16] Lu, Chujie, Long Chen, Chilie Tan, Xiaolin Li, and Jun Xiao. "DEBUG: A dense bottom-up grounding approach for natural language video localization." Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019a, pp. 5147-5156.

[17] Wang, Weining, Yan Huang, and Liang Wang. "Language-driven temporal activity localization: A semantic matching reinforcement learning model." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2019, pp. 334-343.

[18] He, Dongliang, Xiang Zhao, Jizhou Huang, Fu Li, Xiao Liu, and Shilei Wen. "Read, watch, and move: Reinforcement learning for temporally grounding natural language descriptions in videos." Proceedings of the AAAI Conference on Artificial Intelligence. 2019, vol. 33, pp. 8393-8400.

[19] Lee, Kenton, Omer Levy, and Luke Zettlemoyer. "Learning recurrent span representations for extractive question answering." Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2016, pp. 511-521.

[20] Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "SQuAD: 100,000+ questions for machine comprehension of text." arXiv preprint arXiv:1606.05250. 2016.

[21] Seo, Minjoon, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. "Bidirectional attention flow for machine comprehension." International Conference on Learning Representations (ICLR). 2017.

[22] Damen, Dima, et al. "Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100." International Journal of Computer Vision. 2022.

[23] Adams Wei Yu, David Dohan, Quoc Le, Thang Luong, Rui Zhao, and Kai Chen. "Fast and accurate reading comprehension by combining self-attention and convolution." International Conference on Learning Representations (ICLR). 2018.

[24] Vaswani, Ashish, et al. "Attention is all you need." Advances in Neural Information Processing Systems. 2017, pp. 5998-6008.

[25] Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450. 2016.

[26] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016, pp. 770-778.

[27] Xiong, Caiming, Victor Zhong, and Richard Socher. "Dynamic coattention networks for question answering." International Conference on Learning Representations (ICLR). 2017.

[28] Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." International Conference on Learning Representations (ICLR). 2015.

[29] Lin, B., Zhu, B., Ye, Y., Ning, M., Jin, P., and Yuan, L. "Video-llava: Learning united visual representation by alignment before projection." arXiv preprint arXiv:2311.10122. 2023.

[30] Feichtenhofer, Christoph, Fan, Haoqi, Malik, Jitendra, He, Kaiming. "Slowfast networks for video recognition." Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019.

[31] Kay, Will, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustapha Suleyman, and Andrew Zisserman. "The kinetics human action video dataset." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017, pp. 1725-1735.

[32] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2009, pp. 248-255.

[33] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." International Conference on Learning Representations (ICLR). 2015.

[34] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A simple way to prevent neural networks from overfitting." Journal of Machine Learning Research, 15(56):1929–1958, 2014.

[35] Zhang, Songyang, Peng, Houwen, Fu, Jianlong, Luo, Jiebo. "Learning 2d temporal adjacent networks for moment localization with natural language." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. No. 07. 2020.