NaQ: Leveraging Narrations as Queries to Supervise Episodic Memory Supplementary Materials

Santhosh Kumar Ramakrishnan[1] **Ziad Al-Halah**[2] **Kristen Grauman**[1,3]
[1]UT Austin [2]University of Utah [3]FAIR, Meta AI

	$S = 2.5$	$S = 5.0$	$S = 10.0$	$S = 20.0$
R@1, IoU=0.5	8.64	**9.46**	8.08	8.47

Table S1. Varying TRJ scale S with EgoVLP (and **NaQ** stage 1).

Supplementary Materials

We now provide additional information about our experimental settings, and qualitative and quantitative analyses to support our experiments in the main paper.

S1. Implementation details

We perform joint **NaQ** + NLQ training with a large batch sizes and high learning rates for accelerated convergence. For VSLNet and EgoVLP methods, we use a batch size of 2048 and initial learning rate of 0.001 on 2 A40 GPUs with a memory size of 46GB per GPU. For ReLER*, we use a batch size of 1536 and an initial learning rate of 0.001 on 8 A40 GPUs since it has larger memory and compute requirements. We train each method for up to 200 epochs on **NaQ** + NLQ training data, and then finetune them for up to 30 epochs on NLQ training data alone with a lower learning rate. We found finetuning to be unnecessary for VSLNet. For EgoVLP, we finetuned with the original hyperparameter settings from [1] and a learning rate of 0.00001. For ReLER*, we finetuned with the original hyperparameter setting from [2] and a learning rate of 0.0001. We perform early stopping in each case using the performance on NLQ validation split.

For temporal random jittering (TRJ), we performed a grid search with the expansion factor values S=$\{2.5, 5.0, 10.0, 20.0\}$. We found S=5.0 to work best for EgoVLP based on the NLQ validation performance (see Tab. S1). Similarly, we found S=2.5 to work best for ReLER* and VSLNet.

S2. Long-tail of objects in NLQ

Fig. S1 shows the long-tail of objects queried about in NLQ, and the split of low-shot, mid-shot, and high-shot objects used in Sec. 4.3. Note that for a given point x on X-axis, the Y-axis shows the number of objects that have x queries in the NLQ train dataset. For example, there are more than 1000 objects with only 1 training sample.

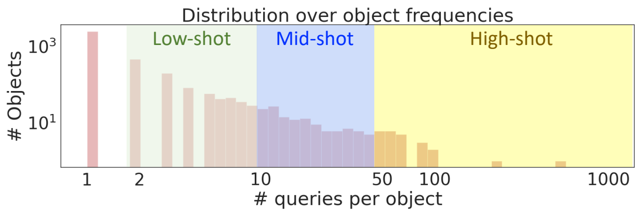

Figure S1. Long-tail of objects in NLQ.

Method	TRJ	IoU=0.3 R@1	IoU=0.3 R@5	IoU=0.5 R@1	IoU=0.5 R@5
VSLNet + **NaQ**	✗	9.80	18.05	5.27	11.05
VSLNet + **NaQ**	✓	**10.26**	**19.01**	**5.81**	**12.67**
absolute gain		**+0.46**	**+0.96**	**+0.54**	**+1.62**
EgoVLP + **NaQ**	✗	15.25	26.15	9.12	17.63
EgoVLP + **NaQ**	✓	**15.90**	**26.38**	**9.46**	**17.80**
absolute gain		**+0.65**	**+0.23**	**+0.34**	**+0.17**
ReLER* + **NaQ**	✗	18.51	23.23	11.36	15.44
ReLER* + **NaQ**	✓	**19.31**	**23.62**	**11.59**	**15.75**
absolute gain		**+0.80**	**+0.39**	**+0.23**	**+0.31**

Table S2. Ablation study of temporal random jittering (TRJ).

S3. Ablation study for Temporal Response Jittering

We study the impact of using temporal response jittering (TRJ) described in Eq. (2). In Tab. S2, we measure the performance of using **NaQ** with and without TRJ, where not using TRJ implies that the seed temporal window from Eq. (1) is used. Overall, we observe a consistent improvement of up to 0.80 in R@1 metrics and 1.62 in R@5 metrics. This indicates that TRJ is able to address the limitations of the seed temporal window.

Method	Object / place queries									People queries
	Where is X before/after Y?	Where did I put X?	Where is X?	What did I put in X?	How many X's?	In what location did I see X?	What X did I Y?	What X is Y?	State?	Who did I interact with during Y?
VSLNet	2.04	1.07	3.38	3.23	4.91	2.60	3.64	2.34	3.07	3.26
+NaQ	**6.62**	**3.58**	3.14	**5.76**	**10.18**	2.60	**8.61**	**5.86**	**8.59**	**6.52**
EgoVLP	5.77	3.58	4.11	9.45	9.82	2.97	7.62	5.08	7.98	8.70
+NaQ	**10.70**	**6.44**	**4.83**	**13.13**	**15.79**	2.60	**11.59**	**7.03**	**12.88**	**13.04**
ReLER*	9.63	6.87	5.82	10.71	14.33	5.46	11.54	6.54	10.12	4.90
+NaQ	**13.98**	**11.34**	6.04	**12.39**	**21.00**	**4.78**	**15.38**	6.54	**14.29**	**7.84**

Table S3. **Performance over NLQ query types.** We report recall@1 at IoU=0.5. We include query types with $\geq$ 100 val samples. We highlight cases where **NaQ** improves recall by more than 0.5 points.

Method	IoU = 0.3		IoU=0.5	
	R@1	R@5	R@1	R@5
VSLNet	5.21	11.19	2.78	6.72
VSLNet + **NaQ** stage 1	**10.26**	**19.01**	**5.81**	**12.67**
VSLNet + **NaQ** stage 1,2	9.34	17.66	5.29	11.85
EgoVLP	10.40	19.33	6.18	13.03
EgoVLP + **NaQ** stage 1	15.61	25.64	**9.46**	16.97
EgoVLP + **NaQ** stage 1,2	**15.90**	**26.38**	**9.46**	**17.80**
ReLER*	14.66	17.84	8.67	11.54
ReLER* + **NaQ** stage 1	18.28	22.95	10.38	14.82
ReLER* + **NaQ** stages 1,2	**19.31**	**23.62**	**11.59**	**15.75**

Table S4. Impact of two-stage **NaQ** training.

S4. Impact of two-stage NaQ training

As we stated in Sec. 4, we train models using **NaQ** augmentation in two stages. In the first stage, we jointly train models on the combined **NaQ** and NLQ dataset with large batch training. In the second stage, we finetune the models on only the NLQ dataset with standard training. In Tab. S4, we study the impact of each stage of training. The first stage helps the most. Stage 2 is not critical, but useful nonetheless (except for VSLNet).

S5. Few-shot analysis

We perform a more detailed analysis of the few-shot performance discussed in Sec. 4.3 and Fig. 6. Specifically, we analyze the zero-/few-shot performance across the various query templates in Tab. S5. When tested zero-shot, **NaQ** already competes with or outperforms the baseline on object/place templates such as 'where did I put X?', 'what X is Y?', and 'object state'. As we inject NLQ data into **NaQ** training, the performance improves steadily on the remaining templates, and outperforms the baseline on 9/10 templates.

S6. Qualitative examples

In `supplementary.html` shared here, we link to qualitative videos for the following:

- Comparing annotations for NLQ vs. Narrations
- NaQ benefits performance on most query templates
- NaQ benefits performance on queries about long-tail objects
- NaQ facilitates zero-shot NLQ

References

[1] Kevin Qinghong Lin, Alex Jinpeng Wang, Mattia Soldan, Michael Wray, Rui Yan, Eric Zhongcong Xu, Difei Gao, Rongcheng Tu, Wenzhe Zhao, Weijie Kong, et al. Egocentric video-language pretraining. *arXiv preprint arXiv:2206.01670*, 2022. 1

[2] Naiyuan Liu, Xiaohan Wang, Xiaobo Li, Yi Yang, and Yueting Zhuang. Reler@ zju-alibaba submission to the ego4d natural language queries challenge 2022. *arXiv preprint arXiv:2207.00383*, 2022. 1

		Object / place queries									People queries
% NLQ	% `NaQ`	Where is X before/after Y?	Where did I put X?	Where is X?	What did I put in X?	How many X's?	In what location did I see X?	What X did I Y?	What X is Y?	State?	Who did I interact with during Y?
100	0	5.77	3.58	4.11	9.45	9.82	2.97	7.62	5.08	7.98	8.70
0	100	3.88	3.83	2.68	2.31	5.00	1.37	4.17	5.23	7.14	2.94
10	100	7.45	5.11	2.46	4.62	6.00	1.02	3.53	4.90	5.95	3.92
25	100	9.63	4.63	3.13	5.46	7.67	1.37	4.81	5.23	5.95	3.92
35	100	8.54	4.95	3.58	7.14	13.33	4.10	7.37	6.54	7.74	4.90
100	100	10.70	6.44	4.83	13.13	15.79	2.60	11.59	7.03	12.88	13.04

Table S5. **Few-shot analysis.** We split the few-shot results from Fig. 6 in the main paper across the various query templates. We report recall@1 at IoU=0.5. The first two columns show the percentage of the NLQ and `NaQ` data used for training. For example, the first row with 100% NLQ and 0% `NaQ` is the baseline, the second row with 0% NLQ and 100% `NaQ` is our zero-shot setting, and so on.