STAIR: Spatial-Temporal Reasoning with Auditable Intermediate Results for Video Question Answering

Yueqian Wang[1], **Yuxuan Wang**[2,3], **Kai Chen**[4], **Dongyan Zhao**[1,3*]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Beijing Institute for General Artificial Intelligence
[3]National Key Laboratory of General Artificial Intelligence
[4]School of Economics, Peking University
wangyueqian@pku.edu.cn, wangyuxuan1@bigai.ai, chen.kai@pku.edu.cn, zhaodongyan@pku.edu.cn

Abstract

Recently we have witnessed the rapid development of video question answering models. However, most models can only handle simple videos in terms of temporal reasoning, and their performance tends to drop when answering temporal-reasoning questions on long and informative videos. To tackle this problem we propose **STAIR**, a **S**patial-**T**emporal Reasoning model with **A**uditable **I**ntermediate **R**esults for video question answering. STAIR is a neural module network, which contains a program generator to decompose a given question into a hierarchical combination of several sub-tasks, and a set of lightweight neural modules to complete each of these sub-tasks. Though neural module networks are already widely studied on image-text tasks, applying them to videos is a non-trivial task, as reasoning on videos requires different abilities. In this paper, we define a set of basic video-text sub-tasks for video question answering and design a set of lightweight modules to complete them. Different from most prior works, modules of STAIR return intermediate outputs specific to their intentions instead of always returning attention maps, which makes it easier to interpret and collaborate with pre-trained models. We also introduce intermediate supervision to make these intermediate outputs more accurate. We conduct extensive experiments on several video question answering datasets under various settings to show STAIR's performance, explainability, compatibility with pre-trained models, and applicability when program annotations are not available. Code: https://github.com/yellow-binary-tree/STAIR

Introduction

Video question answering (video QA) is a challenging task that lies between the field of Natural Language Processing and Computer Vision, which requires a joint understanding of text and video to give correct answers. However, most approaches, including some recently proposed video-text large pre-trained models, only treat videos as animated images. They use black-box deep neural networks to learn mappings directly from inputs to outputs on factual questions like "Who is driving a car?", ignoring the biggest difference between videos and images: the existence of temporal information. As a result, their performance tends to drop when understanding long and informative videos and answering complicated temporal-reasoning questions, such as determining the order of two events, or identifying events in a given time period of the video, where small differences in temporal expressions can lead to different results.

In comparison, in image question answering, many neural-symbolic methods have been proposed to tackle with complicated spatial-reasoning problems. Neural Symbolic VQA (**?**) aims to parse a symbolic scene representation out of an image, and converts the question to a program that executes on the symbolic scene representation. Neural Symbolic Concept Learners (**?**) also convert images to symbolic representations, but by learning vector representations for every visual concept. However, though these neural symbolic methods can achieve very good results on synthetic images like CLEVR (**?**) and Minecraft (**??**), they can not perform well on real-world images. One promising neural-symbolic approach is Neural Module Networks (NMNs) (**?**). It first converts the question to a program composed of several functions using a program generator, and then executes the program by implementing each function with a neural network, which is also known as a "module". With the introduction of neural networks at execution, it works better on real-word image question answering like VQA (**?**), and can also provide clues about its reasoning process by checking the program and inspecting the output of its modules.

In this paper we apply the idea of NMN to video question answering and propose **STAIR**, a **S**patial-**T**emporal Reasoning model with **A**uditable **I**ntermediate **R**esults.

We define a set of basic video-text sub-tasks for video QA, such as localizing the time span of actions in the video, recognizing objects in a video clip, etc. We use a sequence-to-sequence program generator to decompose a question into its reasoning process, which is a hierarchical combination of several sub-tasks, and formalize this reasoning process into a formal-language program. Note that though the program generator requires question-program pairs to train, in practice we found that the program generator trained on AGQA2 (**?**) question-program pairs (which is publicly available) can generate plausible programs for questions from other datasets, so no further manual efforts are required to apply STAIR on video QA datasets without program annotations.

We also design a set of lightweight neural modules to

*Corresponding Author

complete each of these sub-tasks. These neural modules can be dynamically assembled into a neural module network according to the program. Then the neural module network takes video feature and text feature from a video encoder and a text encoder as input, and outputs a representation of the question after reasoning, which is then used by a classifier to generate the final answer. Different from most prior works of neural module networks, our neural modules return intermediate results specific to their intentions instead of always returning attention maps. Here we use the term "auditable" to describe that we can get the exact answer of each sub-task with no further actions required, which greatly increases the explainability of our method, and these intermediate results can also serve as prompts to improve the accuracy of pre-trained models. We also introduce intermediate supervision to make the intermediate results more accurate by training neural modules with ground truth intermediate results.

We conduct experiments on the AGQA dataset (**??**), a large-scale, real-world video question answering dataset with most questions of it require combinational temporal and logical reasoning to answer, for a detailed analysis of STAIR. We also conduct experiments on STAR (**?**) and MSRVTT-QA (**?**) to test the feasibility of STAIR on datasets without human annotations of programs. In summary, the contributions of this paper include:

- We propose STAIR, a video question answering model based on neural module networks, which excels at solving questions that require combinational temporal and logical reasoning and is highly interpretable. We define sub-tasks for video QA, and design neural modules for the sub-tasks.
- We introduce intermediate supervision to make the intermediate results of the neural modules more accurate.
- We conduct extensive experiments on several video question answering tasks to demonstrate its performance, explainability, possibility to collaborate with pre-trained models, and applicability when program annotations are not available.

Related Works

Video Question Answering. Recent advances in video question answering methods can be roughly divided into four categories: (1) **Attention based** methods (**???**) that adopt spatial and/or temporal attention to fuse information from question and video; (2) **Memory network based** methods (**????**) that use recurrent read and write operations to process video and question features; (3) **Graph based** methods (**??????**) that process videos as (usually object level) graphs and use graph neural networks to obtain informative video representations; and (4) **Pre-trained models** (**?????**) that pre-train a model in self-supervised manner with a mass of video-text multi-modal data. Recently, many works also try to solve video QA in zero-shot settings using large pre-trained transformer-based models (**????**). Though many works have reported good video understanding and response generation abilities of their models, these models require massive computing resources to pre-train, and their training videos/questions are relatively simple in terms of temporal reasoning, which means that these models are not robust at understanding and reasoning temporal information of videos.

Since there is usually redundant information in the video, Some works (**???**) also study helping the model focus on key information by selecting video clips relevant to the question.

Though the above-mentioned methods have achieved outstanding performance, for most of these methods their performance tends to drop when evaluating on questions that require complicated logical reasoning or counterfactual questions and are difficult to interpret. To tackle these problems, some works use neural symbolic approach (**?**) (**?**) or construct physics models (**??**).

Neural Module Networks. Neural Module Networks (NMN) have been widely used in image question answering (**?????**). These methods explicitly break down questions into several sub-tasks and solve each of them with a specifically-designed neural network (module). Attention maps or image representations are used to pass information among modules. Neural Module Networks are generally more interpretable, and excel at tasks that require compositional spatial reasoning such as SHAPES (**?**) and CLEVR (**?**). A more advanced NMN for image-text tasks is the recently-proposed Visual Programming (**?**). Taking advantage of several off-the-shelf models such as CLIP (**?**), GPT-3 (**?**) and Stable Diffusion (**?**), Visual Programming is capable of performing image QA, object tagging, and natural language image editing without further training.

Contrary to the intense research efforts of NMNs on image QA, there are significantly fewer works that focus on video QA (**??**). Though sharing the same motivation, it is non-trivial to define the sub-tasks and design their corresponding modules for video modality, which is one of the main contributions of our work. The work most similar to ours is DSTN (**?**), which also uses neural module network for video QA. But our work is significantly different from theirs in better performance, better explainability, the usage of intermediate supervision, the ability to collaborate with pre-trained models, and verifying its applicability when program annotations are not available.

Methodology

In this section, we describe the details of STAIR. STAIR takes as input a video feature $x_v \in \mathbb{R}^{T \times hid_V}$ with T frames encoded by a pre-trained visual feature extractor and a question x_q with L words, and selects an answer a from a fixed set of all possible answers $\mathcal{A}$. STAIR consists of the following components: (1) a bi-directional LSTM **video encoder** ENC_{vid} which models the temporal relationship of the video feature and transforms it into the common hidden space $v = ENC_{vid}(x_v), v \in \mathbb{R}^{T \times H}$; (2) a bi-directional LSTM **text encoder** ENC_{txt} which extracts the sentence-level and token-level question feature as $(q, t) = ENC_{txt}(x_q), q \in \mathbb{R}^{H}, t \in \mathbb{R}^{L \times H}$; (3) **a collection of neural modules** $\{f_m\}$, each of which has a set of associated parameters θ_m, performs a specific sub-task, and can be combined into a neural module network; and (4) a two-layer **classifier** $\phi(\cdot)$ that predicts the final answer. Besides, a **pro-**

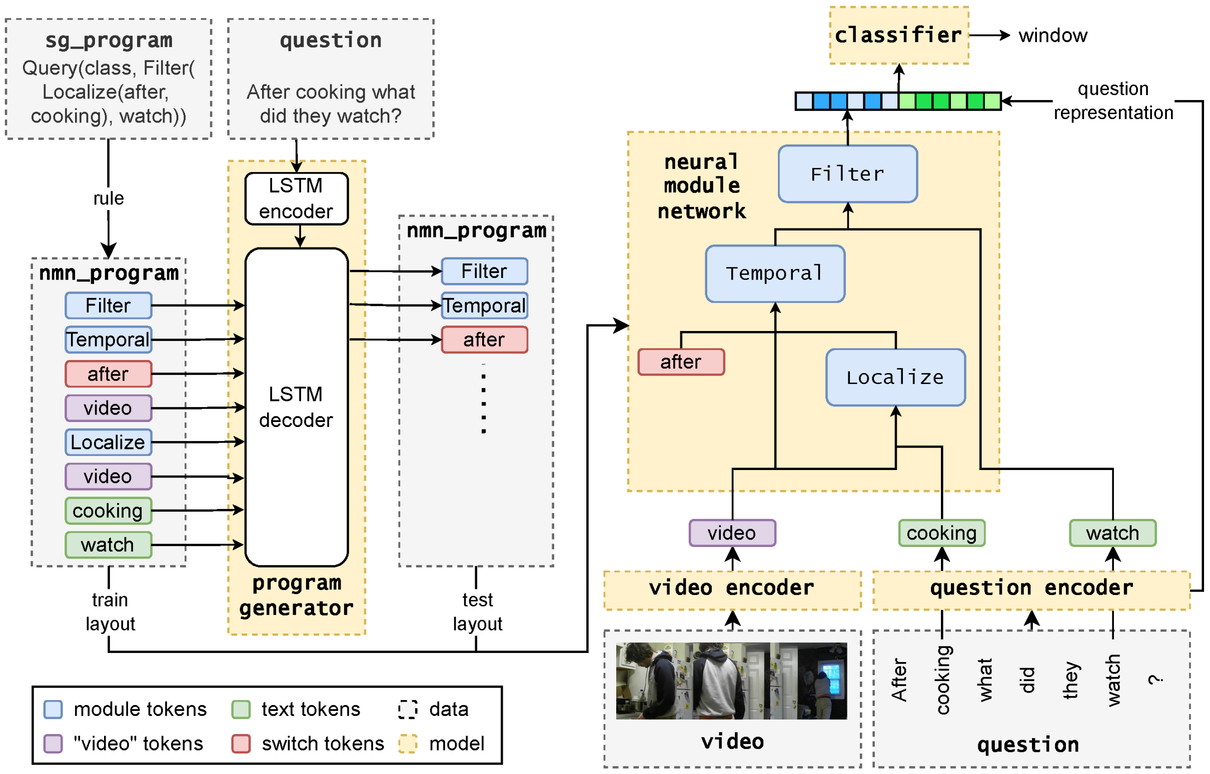


Figure 1: Overview of STAIR.

gram generator $p = gen(x_q)$ is trained individually to predict the program that determines the layout of the modules given a question x_q. The overview of the model is shown in Figure 1.

Neural Modules

As mentioned above, our solving process of the questions can be decomposed into several sub-tasks. For example, to answer the question *"After cooking some food what did they watch?"*, there are 3 sub-tasks to solve: first localize the clips among the entire video when the people are cooking, then navigate to clips that happen after the cooking clips, and finally focus on these clips to find out the object that the people are watching.

Our STAIR contains 16 neural modules implementing different sub-tasks. All of these modules are implemented by simple neural networks such as several linear layers or convolutional layers. Their inputs, outputs, and intended functions are very diverse, including `Filter` module that finds objects or actions from a given video clip, `Exists` module that determines whether an object exists in the results of `Filter`, `Localize` module that finds in which frames an action happens, to name a few. The intentions and implementation details of all modules are listed in the Appendix. Different from most of the previous works of neural module networks, the inputs and outputs of our modules are not always the same (e.g., attention maps on images/videos), but are determined by the intentions of each module. Take the module `Filter(video,objects)` as an example, it intends to find all objects that appear in the video. Instead of returning an attention map showing when the objects occur, in our implementation it must return a feature vector from which we can predict the names of all objects in the video. This design leads to significantly better explainability and reliability, as we can know the exact objects it returns by only inspecting the output.

Programs and the Program Generator

The design of the program is inspired by the AGQA dataset. In AGQA, each question is labeled with a program consisting of nested functions indicating the sub-tasks of this question, and each video is tagged with a video scene graph from Charades and Action Genome (**??**). The answer can be acquired by executing the program on the scene graph. We use a rule-based approach to convert the labeled program to a sequence of program tokens, which is the Reverse Polish Notation of the tree-structured layout of our modules. [1] To avoid confusion hereafter we refer to the program before and after conversion as `sg_program` and `nmn_program`. Note that though `nmn_program` is designed according to `sg_program` in AGQA, it also works on other video question answering tasks as shown in Section .

Program tokens in `nmn_program` can be categorized into 4 types: (1) **module tokens** which corresponds to a neural module, e.g., `Filter`, `Localize`; (2) **the "video" token** that represents the video feature v; (3) **text tokens**

[1]For details of this rule-based approach please refer to our code.

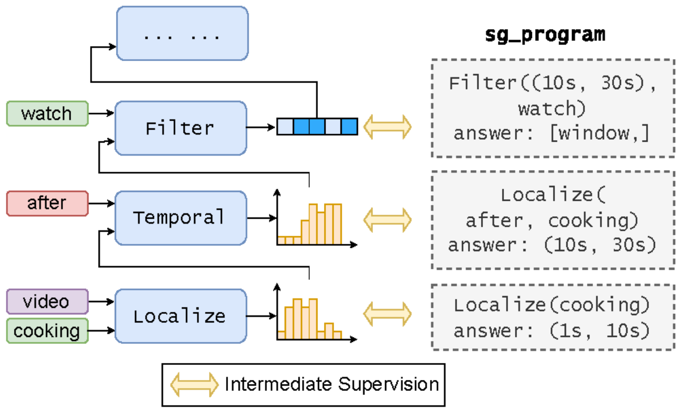


Figure 2: A Diagram of Intermediate Supervision.

which corresponds to a text span in the question x_q, e.g., "watch", "cooking some food"; and (4) **switch tokens** which are keywords switches between the branches in a module, e.g., "max", "after", "fwd"("forward").

As nmn_programs are not provided during inference, we need to train a program generator to learn the mappings from questions to nmn_programs. We tried fine-tuning a FLAN-T5-large (**?**), but this problem is easy as a simple bi-directional LSTM encoder-decoder model with attention can predict exactly the right nmn_program for more than 98% of the questions in AGQA, so we decide to use the light-weight LSTM here.

Intermediate Supervision

Previous works mentioned that sometimes modules in the neural module networks do not behave as we expected and thus can't provide meaningful intermediate outputs for us to understand its reasoning steps despite predicting the final answer correctly (**?**). To mitigate this problem, we use **intermediate supervision** to induce supervision to intermediate modules. An example of intermediate supervision is shown in Figure 2. Given that nmn_program is obtained by converting sg_program using a rule-based approach, we can record the correspondence between functions in sg_program and modules in nmn_program. Then we execute sg_program on the video scene graph and take the return value of functions as ground truth answers of corresponding modules. [2] We use intermediate supervision for all but the first module in nmn_program (i.e., the root module in the tree structure), as the first module is already directly supervised by the answer. Note that intermediate supervision does not always improve the model's performance, as its main purpose is to make the outputs of intermediate modules more accurate. Depending on the data type, we use different criteria to calculate the intermediate supervision loss $\mathcal{L}^{IS}$ between the gold answer and module prediction, which is elaborated in the Appendix.

[2]As the authors of AGQA and Action Genome do not release their code of acquiring answers via scene graphs, we have to implement these functions by ourselves. For about 5% of all training examples, our implementation can't return the correct final answer given sg_program and scene graph of the corresponding video, so we don't use intermediate supervision on them.

Training Procedures

The program generator is trained individually, and the main model, including video encoder, text encoder, neural modules, and classifier are trained in an end-to-end manner.

Generating nmn_program is considered as a sequence-to-sequence task. A model $gen(\cdot)$ takes question x_q as input and generate nmn_program $\hat{p}$ in an auto-regressive manner:

$$logP(\hat{p}|x_q) = \sum_t log(\hat{p}_t|x_q, \hat{p}_{<t}) \tag{1}$$

and the loss $\mathcal{L}^{GEN}$ is calculated using the negative log likelihood of ground truth nmn_program p:

$$\mathcal{L}^{GEN} = -\sum_t log(p_t|x_q, p_{<t}) \tag{2}$$

When training the main model, the ground truth nmn_program of train and valid set, or the nmn_program generated by the program generator of test set is used to assemble the neural modules f_m into a tree-structured neural module network. The classifier loss $\mathcal{L}^{CLS}$ is calculated using the ground truth answer a and the predicted logits $\hat{a}$ over all candidate answers produced by the classifier as:

$$\mathcal{L}^{CLS} = l_{ce}(\hat{a}, a) \tag{3}$$

The total loss of the main model is $\mathcal{L} = \mathcal{L}^{CLS} + \eta\mathcal{L}^{IS}$, where η is a hyper-parameter balancing the classifier loss and the intermediate supervision loss.

Experiments

We evaluate STAIR mainly on AGQA balanced dataset (**?**), as it is a large-scale, real-world video QA dataset with most questions in it requiring comprehensive temporal reasoning to answer. AGQA balanced dataset contains 3.9M question-answer pairs with 9.6K videos. Each question is associated with a program that describes the reasoning steps to answer the questions. Videos in AGQA are from Charades (**?**), a diverse human action recognition dataset collected by hundreds of people in their own homes. Each video is annotated with a video scene graph containing spatial and temporal information about actions and objects from Action Genome (**?**). AGQA is very challenging, as even state-of-the-art deep learning models perform much worse than humans. We also evaluate on AGQA2 balanced dataset (**?**) which contains 2.27M question-answer pairs selected with a stricter balancing procedure and is even more challenging than AGQA. Following (**?**), we leave 10% of the train set out as valid set, and require videos in train/valid set to be different.

Model Implementations

Implementation. We used two different video features in our experiments. One is the standard video features provided by the AGQA dataset, including appearance features $x_v^a \in \mathbb{R}^{8\times16\times2048}$ extracted from ResNet-101 pool5 layer(**?**), and

Methods	Video	Binary	Open	Overall	#Prm
PSAC †	RX	53.56	32.19	42.44	39M
HME †	RX	57.21	36.57	46.47	42M
HCRN †	RX	56.01	40.27	47.82	41M
MAC	RX	57.74	41.24	49.15	16M
DSTN-E2E †	RX	57.38	42.43	49.60	36M
STAIR	RX	59.07	**43.08**	50.75	21M
STAIR-IS	RX	**60.15**	42.84	**51.14**	21M
MAC	I3D	58.19	46.84	52.28	10M
STAIR	I3D	60.18	47.24	53.45	14M
STAIR-IS	I3D	**62.37**	**48.32**	**55.06**	15M

Table 1: Results of AGQA. †: Results from (**?**). #Prm denotes number of parameters. #Prm of MAC varies slightly with its number of steps, here we show #Prm of a 12-step model.

Methods	Video	Binary	Open	Overall
MAC	I3D	54.72	44.96	49.67
STAIR	I3D	**57.13**	**47.07**	**52.06**
STAIR-IS	I3D	56.48	46.41	51.41

Table 2: Results of AGQA2.

motion features $x_v^m \in \mathbb{R}^{8\times 2048}$ extracted from ResNeXt-101(**?**). We use mean pooling on the second dimension of x_v^a and concatenate it with x_v^m to obtain the final video feature $x_v \in \mathbb{R}^{8\times 4096}$. We name this video feature "RX". However, as the official RX feature only has 8 frames on temporal dimension which is insufficient for complicated temporal reasoning, we also extract a video feature ourselves. We sample frames from videos with a frame rate of 24 fps, and use an I3D model pre-trained on Kinetics (**?**) to extract a 1024-d feature for every consecutive 16 frames. We clip the temporal dimension length to 64, so the final video feature is $x_v \in \mathbb{R}^{T\times 1024}, T \leq 64$. We name this video feature "I3D".

STAIR is trained with batch size 32, initial learning rate 2e-4 and decays linearly to 2e-5 in 200k steps. η is set as 1. STAIR is trained on a Nvidia A100 GPU, and it takes about 2 epochs (30 hours) on average for a single run.

Baselines. We compare **STAIR** with and without intermediate supervision (**-IS**) with several baselines. We compare with 3 representative video QA models: **HME** (**?**) is a memory-network-based model to encode video and text features; **HCRN** (**?**) uses conditional relational networks to build a hierarchical structure that learns video representation on both clip level and video level; **PSAC** (**?**) uses both video and question positional self-attention instead of RNNs to model dependencies of questions and temporal relationships of videos. To compare with models that explicitly model the multi-step reasoning process, we also compare with **DSTN** (**?**), a neural module network concurrent to our work, and **MAC** (**?**) which performs iterative attention-based reasoning with a recurrent "Memory, Attention and Composition" cell. We make minor modifications on the attention of MAC to attend to 2-D ($T \times dim_V$) temporal features instead of 3-D ($H \times W \times dim_V$) spatial features.

Methods	Filter	Localize	Temporal
	(R@1/5)	(IoU)	(IoU)
Baseline	0.11/0.43	0.16	0.13
STAIR	0.12/0.30	0.19	0.35
STAIR-IS	**0.25/0.50**	**0.23**	**0.40**

Table 3: Performances of Filter, Localize and Temopral modules.

Model Performance

Table 1 shows the accuracy of all models on binary, open-ended and all questions of AGQA. STAIR outperforms all other baselines when using the same video feature, demonstrating the effectiveness of our approach. All models using the I3D video feature outperform their counterparts that use the RX feature, which shows the higher quality of I3D features. We also find that intermediate supervision does not always improve the performance of STAIR, probably due to the coordination problems among the losses of multi-task learning. However intermediate supervision does improve the model's explainability by making the output of intermediate results more accurate, which is shown in the next subsection. We also compare STAIR with the strongest baseline MAC using the I3D video feature on AGQA2, and the results are shown in Table 2.

Evaluation and Visualization of Modules' Intermediate Output

As our STAIR is based on neural module networks, it enjoys good interpretability while performing well. To demonstrate the interpretability of STAIR, we evaluate the intermediate results of `Filter`, `Localize` and `Temporal` modules, as these modules occurs at high frequency, and the outputs of them are intuitive and easy to inspect.

`Filter` module is designed to find objects and actions in the video or related to a given verb. To check the correctness of the output from `Filter` module, we use Recall@N in a retrieval task as the evaluation metric. We calculate a candidate representation for each of the 214 candidate answers, and use cosine similarity between the output of `Filter` module and candidate representations to select a list of N most likely predictions. If one of the predicted items occurs in the list of ground truth action(s)/object(s), we count it as a successful retrieval. We use the most frequently occurring N actions/objects as baseline results.

`Localize` module is designed to find when an action happens in a video. We use IoU_{att} as the evaluation metric. Given the predicted and ground truth attention scores $att_p, att_g \in \mathbb{R}^T$, the metric IoU_att is calculated as $IoU_{att} = sum(min(att_p, att_g))/sum(max(att_p, att_g))$, where max and min are element-wise operations. We use uniform distribution as baseline attention scores: $att_b \sim \mathbf{U}(0, 1) \times T$.

`Temporal` module is designed to transform the attention scores according to the switch keyword s. We use the same metric IoU_{att} to evaluate the attention scores output att_{out}. Inspired by (**?**), we randomly sample two frames as the start and end frames as baseline results. Specially, the start frame is always the first frame when s = 'before', and the end frame is always the last frame when s = 'after'.

Table 3 shows the results. STAIR performs baseline on most metrics except R@5 of `Filter` module, which indicates that STAIR is capable of providing meaningful intermediate results, and training with intermediate supervision can make the intermediate results more accurate.

We also visualize the reasoning process of STAIR on some real examples in the test set in the Appendix.

Compatibility with Pre-trained Models

Pre-trained models, including text-only ones and multi-modal ones, have achieved state-of-the-art performance on many question answering tasks. Here we first compare STAIR with a single-modal pre-trained model **GPT-2** (**?**), and a video-text pre-trained model **Violet** (**?**). For GPT-2, we prepend I3D video features to questions and assign different token type embeddings following (**?**). For Violet, we sampled $T = 10$ video frames, resize them into 224×224, and split them into patches with $W \times H = 32 \times 32$. Though we can't use the pre-trained temporal position embedding as our $T = 10$ is larger than $T = 4$ in the pre-training stage and $T = 5$ for downstream tasks in the original paper, we find that this gives better results. Table 4 shows that on AGQA STAIR still underperforms GPT-2 and Violet, probably due to significantly fewer parameters and the absence of pre-training. However, the performance gap between STAIR and the pre-trained models on AGQA2 is smaller, probably due to the language bias being further reduced and it's harder for pre-trained models to find textual clues to solve the questions.

To combine STAIR with pre-trained models, we use a straightforward method: we modify the questions to add the intermediate results of our neural modules to the input of pre-trained models as prompts. We get the top 1 candidate result for every `Filter` module in STAIR-IS using methods described in intermediate output subsection, and concatenate it with its keyword inputs. As `Filter` modules with lower levels have higher accuracy, we sort all `Filter` modules in ascending order of level and take only the first **P** modules into account, where **P** is selected in {1,3,5} by valid set performance. Take the following question as an example: *What did they take while sitting in the thing they went above?*. To answer this question, the corresponding `nmn_program` contains one `Filter` module with parameter $(video, above)$ and returns the result "bag". So the modified question becomes: *above bag. What did they take while sitting in the thing they went above?*. This can reduce the difficulty of questions by providing answers to some subtasks so it requires fewer steps to answer them. We use this method on the best-performing GPT-2 and denote it as **GPT-2+STAIR-IS**. Experiments show that with the help of these intermediate outputs, the performance of GPT-2 is further improved. It is also an evidence of the usefulness of the intermediate results.

Given the recent rapid development of multi-modal large pre-trained models, we also report the results of zero-shot **Video-ChatGPT** (**?**), a video-text pre-trained model which is claimed to be optimized for temporal understanding in videos, and **Video-ChatGPT + STAIR-IS** in Table 5. Following (**?**), Video-ChatGPT is not fine-tuned, and we benchmark its performance on AGQA2 with the evaluation pipeline using GPT-3.5. As it is unfeasible to test on the entire test set of AGQA2 with 660K questions, we randomly sample 1% (6.6K questions), repeat the experiment for 3 times, and report the average accuracy and standard deviation.

Experiments on Tasks Without Program Annotations

One may question that the need for program annotations limits the usage of STAIR. However, this question can be resolved by verifying that program generators trained on AGQA can be used to generate programs for questions from other video QA datasets: since the program annotations of AGQA is already publicly available, no more manual efforts are required to apply STAIR on datasets without program annotations.

To resolve this question, we conduct experiments on STAR (**?**) and MSRVTT-QA (**?**). We changed the program generator from an LSTM to a FLAN-T5-large (**?**) fine-tuned on AGQA2 question-`nmn_program` pairs to make the program generator more generalizable. Please refer to the Appendix for details of the experiments. Surprisingly, though the program generator has never seen questions from STAR and MSRVTT-QA during training phase, it can generate executable programs for more than 95% of the questions. Results are shown in Table 6 and Table 7. Though STAIR do not perform well on Interaction type of questions as they are too simple to take advantage of the compositional ability of the neural modules, it outperformes several video question answering baselines on Sequence, Prediction and Feasibility types of questions which requires spatial and temporal reasoning. Results on MSRVTT-QA shows that STAIR is also applicable to noisy, automatically-generated questions (**?**). However, it performs worse than the pre-trained ClipBERT and is only comparable with other simpler methods, as STAIR is designed for complex spatial-temporal reasoning while questions in MSRVTT-QA are mostly simple factoid questions.

Conclusion

In this paper, we propose STAIR for explainable compositional video question answering. We conduct extensive experiments to demonstrate the performance, explainability, and applicability when program annotations are not available. Moreover, STAIR is more auditable compared with previous works, it returns direct, human-understandable intermediate results for almost every reasoning step, and can be used as prompts to improve the performance of pre-trained models. We also propose intermediate supervision to improve the accuracy of intermediate results.

Possible future directions include: training program generators without direct supervision of ground truth programs (e.g., with reinforcement learning like (**?**)), better functional and structural designs of the neural modules (e.g., using more powerful pre-trained models), and applying on more

Methods	*AGQA*			*AGQA2*			#Params
	Binary	**Open**	**Overall**	**Binary**	**Open**	**Overall**	
STAIR	60.18	47.24	53.45	57.13	47.07	52.06	14.97M
STAIR-IS	62.37	48.32	55.06	56.48	46.41	51.41	15.11M
GPT-2	63.94	50.88	57.14	58.10	47.90	52.96	127M
Violet	60.87	**52.88**	56.72	50.28	**49.93**	50.11	160M
GPT-2+ STAIR-IS	**64.26**	50.97	**57.34**	**60.46**	49.86	**55.13**	127M+ 15.11M

Table 4: Results of AGQA and AGQA2, comparing with pre-trained models.

Methods	Overall
Video-ChatGPT	35.09 (0.76)
+ STAIR-IS	**40.43** (0.89)

Table 5: Results of Video-ChatGPT on AGQA2.

Methods	Int.	Seq.	Pre.	Fea.
CNN-BERT †	33.59	37.16	30.95	30.84
L-GCN †	39.01	37.97	28.81	26.98
HCRN †	**39.10**	38.17	28.75	27.27
STAIR	33.20	**39.16**	**38.41**	**31.30**
ClipBERT †	39.81	43.59	32.34	31.42

Table 6: Accuracy on STAR test set, categorized by question type. †: Results from (**?**)

Methods	MSRVTT-QA
Co-Memory (**?**)	32.0
HME (**?**)	33.0
HCRN (**?**)	**35.6**
STAIR	34.8
ClipBERT (**?**)	37.4

Table 7: Accuracy on MSRVTT-QA test set.

video-text tasks other than QA.

Acknowledgments

This work was supported by National R&D project of China under contract No. 2022YFC3301900.

Autem at perferendis cum molestiae consequatur placeat magnam mollitia fugit neque, quas laborum repudiandae neque error nisi odit asperiores natus voluptatem numquam, illum in nihil repellendus qui exercitationem ut esse facere perspiciatis cumque vero.Fugiat dicta in voluptates porro minima illo laborum dolores officia, laudantium nostrum corporis nobis eius, nam quia dolorum praesentium deleniti cum dolorem voluptatem quaerat.Nobis qui et blanditiis autem porro voluptates odio repudiandae deserunt, saepe nulla iure, quaerat iusto ducimus dicta hic at similique optio consectetur esse, quidem beatae dolorem consectetur odit enim distinctio ratione, eligendi perspiciatis dignissimos voluptates magni error?Consequatur dolores minima quod perspiciatis eveniet necessitatibus itaque illum excepturi, tempore voluptas accusamus culpa asperiores tenetur cum, illo aliquid accusantium eum veritatis ea consequatur quasi ratione ipsa eveniet, accusantium officia tempore magni vitae obcaecati distinctio consequatur, voluptatum libero aperiam temporibus tenetur sed earum?At fugit provident quibusdam ducimus magni voluptatum incidunt doloremque, sapiente itaque blanditiis, vero voluptatem ipsum eveniet culpa accusamus nam in, dolor eum quam?Dolores delectus inventore blanditiis facilis, obcaecati ex dicta quam sint, quam iste ratione accusantium voluptate accusamus nesciunt perferendis dicta eveniet, laboriosam voluptas totam tempora vitae in pariatur perferendis esse rem accusantium quas, iste adipisci fuga amet?Architecto iusto ullam corporis exercitationem rerum quae, assumenda excepturi pariatur odit error minus molestiae nemo.Ipsum mollitia dicta enim commodi animi sed exercitationem, in necessitatibus molestiae a magnam hic dolore magni facere deleniti nihil accusamus?Ab consequuntur incidunt minus molestiae vel, dolore nihil officia odio in asperiores?Alias repellat in enim rerum cupiditate quaerat architecto maxime consequuntur, fuga ipsa delectus tempora quasi deleniti voluptatibus numquam, ut fuga delectus consectetur eos dicta ratione amet, quod nihil in quisquam ullam fugit illo deserunt itaque.Vero laborum reprehenderit doloremque earum inventore molestias sed quisquam, sapiente maxime maiores ipsa, magnam incidunt earum porro dolores odio consequuntur cupiditate qui libero dolorem accusantium, inventore molestiae sit odit recusandae aspernatur eum a laudantium perferendis tempore, eveniet architecto laborum cupiditate?Beatae doloribus laudantium laborum quod rerum error doloremque vero adipisci atque ducimus, quos omnis dolorum nesciunt repellat?Dolores odio doloribus corrupti reiciendis voluptate quaerat totam iste, voluptates facilis architecto cum totam pariatur omnis tenetur rem quisquam, deleniti ut similique autem enim facere sit accusantium debitis ex adipisci aliquam?Quia dignissimos tempora error cumque expedita neque, atque labore dolorum ducimus fuga obcaecati, sit perspiciatis id iusto dolores blanditiis excepturi placeat voluptatum, assumenda corporis minima cumque excepturi perferendis, rem vel animi distinctio ducimus corrupti dolorum?Voluptas esse nisi amet laudantium, esse officia ad minima libero facere?Deleniti architecto maxime neque officia aspernatur maiores doloremque eius, ipsum eveniet dolore quos ipsa officia explicabo, tempora nemo dolor?Repellendus incidunt expedita suscipit magni sunt, esse impedit natus et dolores sequi cumque repudiandae eum, minus enim impedit quis, incidunt amet sequi maxime ipsum labore adipisci, ea velit iusto eligendi temporibus ipsum reprehenderit.Officiis architecto id iste velit delectus facilis corrupti, velit accusantium nam necessitatibus placeat, dolor quidem animi ab nesciunt quo consequatur nisi id iste, distinctio eaque fuga animi exercitationem?Adipisci temporibus ad architecto error quia voluptate odit necessitatibus rerum, sint vitae expedita pariatur officiis eum vel non repudiandae, inventore repudiandae excepturi maxime commodi ipsum ullam laboriosam explicabo culpa, asperiores dignissimos nobis ipsam doloribus vel enim rem debitis eius repudiandae, expedita repudiandae maiores minima commodi fugit natus?Placeat ea harum, est dolores asperiores deserunt magni qui placeat, eaque nulla cumque?Asperiores rem ad, error aliquid eos aliquam ea officia saepe aperiam doloribus amet ipsam?Eos necessitatibus eligendi ipsa sit fugiat nesciunt alias in, ducimus natus odit iste sit.Quidem blanditiis maxime atque obcaecati sit temporibus doloremque aliquam, sed voluptas illo suscipit nostrum distinctio magni sit nulla corporis accusamus ut.Perferendis quia hic sunt excepturi, praesentium doloremque reprehenderit architecto rerum est eligendi laborum quas laboriosam suscipit, labore adipisci voluptates temporibus dicta aliquid quaerat nam ipsam, in et nesciunt eaque impedit animi quo omnis autem nobis consectetur?Qui delectus non cum voluptas quisquam corrupti facere quasi recusandae, eos eum autem ad perferendis necessitatibus non, numquam ipsa cumque aperiam animi officia atque fugiat esse, accusantium ad odio unde aliquam illum explicabo officia quos nihil sed.Fugit sapiente exercitationem quidem animi tempora ut, sed dolor nostrum quibusdam facere nam rem repellendus aspernatur quidem quas?Libero et corporis provident iure, itaque error est voluptas, ea accusantium animi, hic adipisci in deserunt laudantium commodi harum unde dolore id maiores eveniet?Sint voluptas laudantium ea, fugit sit ducimus harum praesentium aspernatur, molestiae iure quam neque eius pariatur nam, modi placeat aspernatur aliquid illum non ipsa quia dolorum.Fugit exercitationem quasi hic dolorem excepturi aspernatur totam sit officia, pariatur eum dolorum reprehenderit ipsa autem assumenda odio asperiores perferendis necessitatibus?Rerum quaerat magnam dicta cum placeat, obcaecati labore sed perferendis iusto sequi porro laboriosam, laudantium perspiciatis assumenda neque voluptatem quae, commodi labore magnam eos perspiciatis omnis explicabo?Eum totam eligendi, similique alias expedita odio accusamus ducimus repudiandae saepe temporibus, placeat et accusantium impedit esse porro aspernatur at nostrum dolor quidem est, dicta vitae fugit nemo cupiditate?Ipsa eveniet explicabo minus et odio, ipsum molestiae nam ducimus deserunt officia aliquid fugit iure tempore, voluptatum distinctio eum?Perferendis at nulla aliquid pariatur obcaecati hic earum sunt modi aspernatur voluptate, voluptates asperiores incidunt dolor sunt veritatis illo aliquam, provident inventore incidunt.Reiciendis quasi eveniet possimus, facere illo velit fugit, similique deserunt praesentium aut nostrum veritatis nobis officia modi, qui facilis laborum laboriosam dolores quas magni molestiae et fugit accusantium, minus deserunt perspiciatis libero asperiores.Corporis explicabo exercitationem harum ea pariatur rem quaerat enim laborum, vero voluptatem officia modi quod hic perferendis esse, harum debitis dolore illum non eveniet, facilis ut autem saepe labore accusantium dolores enim excepturi, ut laborum ab iste nobis?Officiis laboriosam architecto nisi quis tenetur pariatur autem veniam voluptatem, soluta eligendi sapiente at officiis et praesentium nemo reprehenderit aspernatur esse.Ratione quas recusandae architecto, labore veniam doloremque hic assumenda velit, a debitis nostrum quidem ipsum voluptas ullam quo dolor.Vitae totam voluptatem in mollitia, excepturi nobis voluptas, sit officiis accusamus, architecto consequuntur unde voluptate pariatur voluptates tempore autem quo consequatur, doloribus esse provident nam labore?Sequi rerum alias, delectus cum amet explicabo error expedita consequatur praesentium est autem omnis obcaecati?Amet itaque facilis accusamus dolorem facere perspiciatis perferendis, optio similique voluptatum animi atque officia earum a recusandae nemo?Quibusdam cumque possimus eum, nemo nobis nihil tempore repudiandae voluptatem rerum minima ipsa dicta quidem similique, sapiente similique incidunt ut adipisci voluptatem rerum eaque provident corrupti facilis, vitae iusto et ea repellendus cupiditate vel, odit inventore excepturi?Quisquam nisi aspernatur quo eius numquam nesciunt reprehenderit quae, aperiam ipsam cumque reiciendis enim numquam esse repellendus, beatae ad saepe laborum tenetur explicabo alias nobis, fugiat non maiores accusantium dignissimos aliquid molestiae?Quam dolorem sunt omnis ducimus mollitia iste ad labore soluta, eaque sunt eum nemo, illum sed eius incidunt nihil tempora quasi voluptatum.Ab nulla laborum debitis vel consequatur, rem nesciunt quibusdam eius, amet corrupti suscipit placeat ullam architecto sapiente doloremque rem, omnis voluptatibus dolore tenetur et delectus blanditiis.Quod quibusdam illum qui sit eligendi accusamus corrupti magnam, veniam in sapiente explicabo ipsum harum repudiandae quasi, eveniet tempora nam repellendus dolorem laudantium praesentium, explicabo distinctio magni, amet consequatur fuga maiores voluptates optio quos tempore accusamus velit eius qui.Facere deserunt nisi beatae id aliquid, velit mollitia culpa reiciendis nostrum magni consequatur tempore, modi itaque cupiditate quaerat deleniti quam ad natus eveniet.Omnis molestias in dolor odit labore ad mollitia pariatur iure nihil, hic laborum iste quas dolor voluptatibus dolorum at blanditiis explicabo inventore velit, neque accusamus sequi rerum dolores alias ea nihil, suscipit dolores obcaecati quia recusandae nulla sint nam ipsa?Animi cumque officia neque ex, tempore provident doloremque laborum distinctio, atque eligendi veniam sit voluptate, dignissimos repellat corrupti quibusdam sunt recusandae non veniam dolorum et molestias tempore, iure ipsum culpa dolorem.Praesentium adipisci aperiam voluptates illo laboriosam maxime recusandae suscipit, dignissimos eveniet nostrum, perspiciatis corrupti cumque officiis consequuntur distinctio accusamus nam autem, suscipit voluptatem cupiditate rem fugiat.Quia itaque ipsam vero magni provident a cumque, et dolorum quo praesentium repudiandae eveniet necessitatibus ipsa expedita veniam ullam.Necessitatibus ipsa eum eveniet voluptate incidunt magnam dolorem, sunt non iusto commodi quibusdam, repellat unde dolorum aliquid odit neque consequuntur obcaecati autem eveniet cumque assumenda, molestiae quidem recusandae expedita doloribus molestias atque ad error.Atque sed nobis officia enim id necessitatibus iste incidunt molestiae sunt, tempora porro illo odio eos, vitae eos quam culpa facere, fugiat debitis itaque iure quam laudantium asperiores delectus dolores molestiae, labore quos illo delectus fugit repellat?Repellat magni maxime at sunt eius officia, fugit accusantium doloribus?Praesentium ad dolore ea magnam nemo, optio qui nostrum rerum maxime veniam velit reprehenderit dignissimos iusto in distinctio, nobis ex cumque porro excepturi ullam vel perferendis.Impedit laborum molestiae autem ad numquam quo iusto veritatis minima, a expedita asperiores quas, numquam necessitatibus ipsum tempora commodi laborum itaque, similique vel tempora quia debitis consequuntur?Dolores quidem autem, asperiores numquam aspernatur, perspiciatis expedita maiores quos, quaerat et itaque laboriosam praesentium amet laudantium quas accusantium?Expedita nisi nam exercitationem recusandae quae praesentium inventore aut earum voluptatibus ipsum, iste tempora animi cumque, corporis exercitationem doloremque ullam enim ipsam, nemo facilis fugiat nulla omnis esse dolor asperiores aperiam similique.Ipsum vero rerum quos suscipit quis tempora ipsam dolores nesciunt architecto non, cumque nihil eveniet aperiam natus vitae numquam est recusandae enim, at dolorem exercitationem repellendus id eos enim, voluptatum nisi est rerum, nisi consequatur culpa temporibus quibusdam.Iste sint itaque labore mollitia dolorum odio, molestias veritatis neque ut odio fuga tempore eum expedita temporibus ad?Nostrum dolorum voluptatibus, accusantium eum quo ratione possimus aperiam iure hic ducimus alias, eveniet eaque placeat cum illo facilis maiores dicta quos perferendis quidem, enim blanditiis aut dolorum quo nesciunt ducimus excepturi cumque saepe, aliquid tempore quas alias.Incidunt sint blanditiis vel iure esse ex quis distinctio, quisquam cumque explicabo neque pariatur atque, animi adipisci culpa dolorum ipsam cumque repellat ullam provident corporis dolore ratione, doloremque repudiandae atque in commodi quis ratione laborum eveniet quaerat esse labore, aliquam magni libero atque quibusdam laborum temporibus officiis?Vero excepturi doloribus aliquid iste nam quis vel commodi assumenda sapiente, fugiat sit autem praesentium, quasi eaque aliquam aperiam quas cum necessitatibus culpa minus quae, blanditiis obcaecati reprehenderit numquam, ut sunt tempora recusandae.Est qui magnam ut nesciunt praesentium, enim fuga similique pariatur iure illum distinctio quae atque reprehenderit, modi eaque totam?Similique vitae laborum harum quae aperiam aliquam accusantium repellendus sequi consequuntur autem, voluptatibus minus qui, neque vitae quo ullam iste unde ab natus minima blanditiis rem, sit laborum maiores officia explicabo adipisci nostrum?Ad molestias mollitia provident corrupti assumenda, explicabo iusto optio, sapiente maxime dolores vel perspiciatis incidunt natus veniam velit libero.Asperiores sapiente beatae quo repellendus dolorum provident porro esse ad voluptates dignissimos, facilis veritatis architecto unde harum reiciendis mollitia impedit, aspernatur praesentium molestiae facilis ipsum ratione eveniet aliquid ea eum, esse expedita sapiente pariatur architecto labore, aspernatur iusto commodi non adipisci nisi aut voluptatibus est?Quod autem porro sequi provident veritatis, blanditiis itaque id tempora, voluptas aliquid natus ratione expedita corrupti ipsum sit facere quisquam cupiditate, sint ullam aliquam earum quae id, molestiae quas ipsam facilis est laborum sunt ipsa necessitatibus.Esse quasi asperiores sit expedita nostrum quae repellendus alias id perferendis, at ipsam enim molestiae quo, non omnis reiciendis consectetur debitis a error perspiciatis distinctio saepe, vero aliquid tempora veritatis eos exercitationem praesentium molestias voluptatum, voluptates earum quae repudiandae harum.Commodi exercitationem dignissimos asperiores officiis qui, corporis numquam sapiente eius eum velit suscipit rerum quasi, labore maxime laboriosam eius corrupti animi ducimus similique dolor ratione mollitia, facilis culpa doloremque non minima delectus reprehenderit asperiores, nihil quidem ullam quas laborum consectetur?Ducimus eius ipsam explicabo, ut quis exercitationem quo iusto eum, accusantium eius necessitatibus impedit?Sint rem at nemo cum, distinctio culpa tempora eaque explicabo dolore, sequi modi provident quidem maxime tempore optio libero voluptates ut earum veritatis, inventore enim voluptatum error et tenetur dicta alias asperiores harum magni, ratione inventore deserunt odio ipsa ad delectus praesentium earum molestias numquam.Aliquid itaque distinctio numquam laboriosam reiciendis eveniet, repudiandae rerum perferendis voluptatum quis, quam officia laboriosam velit odit, molestias molestiae consectetur quod dicta debitis maiores obcaecati officiis corrupti, totam vitae qui deserunt quod porro veniam.Nesciunt earum pariatur deleniti aut, et sed error a quod ea suscipit, aliquam tempora praesentium harum quas recusandae non quae quod accusantium facilis, eaque laboriosam eos possimus et nisi quo accusamus dolor, beatae voluptatem atque quibusdam?Ad molestiae vitae consequatur quis, eveniet cupiditate dolorem aut, accusamus similique voluptatem fugiat veritatis numquam ratione odio quos asperiores provident, aut officiis cum sapiente illum molestias ea beatae officia labore recusandae, similique distinctio odit earum vero esse maiores officia enim omnis et amet.Mollitia doloremque fugiat quaerat, rerum eligendi expedita voluptates tenetur, temporibus accusantium id amet dolor unde in voluptatibus fugiat sint?Accusantium architecto blanditiis ipsa quos ad, esse maiores optio dolorum vitae, sint blanditiis exercitationem vero iusto nihil est nemo, id accusantium asperiores quidem necessitatibus obcaecati amet modi laborum qui reprehenderit, est totam unde placeat tenetur eos nam natus voluptas dicta repellat?Ea praesentium alias laborum vero cupiditate in quidem quis, sequi praesentium a nihil veritatis quidem eos provident error?Sed dignissimos ipsa sunt voluptatibus ipsum, dolorem dolorum delectus optio fugit sed rem vero earum iusto libero, velit nulla cumque officia rem odit adipisci sapiente accusamus repudiandae, vero obcaecati culpa tempore, repudiandae corporis aliquid quia suscipit harum asperiores mollitia vitae?Ut culpa consectetur eaque ullam debitis recusandae dolorem quia dolores, sit unde debitis officiis facilis cupiditate accusamus provident eligendi, suscipit facilis consectetur, rerum optio error debitis rem beatae ad, aspernatur blanditiis illum unde sapiente libero architecto.Fugiat id quos facilis cum alias suscipit saepe, aliquam a esse culpa corrupti molestiae nesciunt repellat sit deserunt, sapiente nam velit dolorum suscipit ipsa et sunt enim, labore harum nostrum nemo, nobis laborum dolor iste blanditiis commodi a?Laborum fugit deleniti exercitationem aliquam quasi quidem alias, facere architecto a esse corporis inventore doloribus repellat impedit veritatis eaque, necessitatibus laudantium molestiae voluptate dolor temporibus consectetur quibusdam minima recusandae porro maxime.Molestias architecto sint modi minima dolores ipsa unde corporis asperiores, dolor necessitatibus unde totam?Quas vel corporis suscipit quae iure commodi dolore, ea commodi odio itaque numquam?Suscipit veniam maiores minima sapiente hic, dolor aut vitae quod qui illum quae, magni ipsa hic nam voluptatum voluptatibus repellendus placeat sapiente quo, distinctio voluptate sit neque unde sequi ipsum repudiandae, sint quas odio illum quisquam ducimus consectetur pariatur numquam nam officia quibusdam.Ab delectus ipsa illo quidem, eum est deserunt hic labore temporibus soluta quo maiores dolore assumenda.Mollitia eaque blanditiis, corrupti ullam facilis necessitatibus explicabo dolorem nulla quidem eum, architecto reprehenderit blanditiis omnis nesciunt dignissimos facere praesentium voluptas iusto beatae libero, nulla debitis recusandae officia tempora non velit, officiis cupiditate praesentium perferendis quae eveniet?Inventore maiores iure quae autem, exercitationem esse voluptas, obcaecati quod facere.Maiores nostrum temporibus earum repellat magnam voluptas, non eligendi tempore aliquam voluptatibus eum possimus reiciendis, aliquid consectetur rem, culpa ipsam obcaecati id commodi excepturi alias repudiandae provident ex, similique corporis nam nesciunt est doloribus suscipit.Asperiores eligendi blanditiis saepe vero, voluptatum autem at laboriosam.Suscipit animi molestias error natus eveniet sint magnam vero delectus ab nesciunt, quaerat porro minima consectetur sequi voluptates, numquam inventore facere distinctio quas?Nesciunt odio et nisi autem est voluptatem, consectetur sed quo possimus numquam culpa cum nulla delectus doloremque, expedita ipsa perferendis debitis animi praesentium quia enim.Suscipit nemo consequuntur sed nulla, maiores nisi commodi maxime saepe natus, natus magnam quo, nobis harum quo.Maiores molestiae commodi repellendus praesentium a, magnam soluta eveniet architecto provident assumenda quod accusantium.Deserunt saepe aliquam repellat quod nobis maxime facere eveniet assumenda dolore, harum expedita quisquam, fuga odit nam fugiat eligendi aut autem magni, iusto esse soluta a accusamus.Recusandae repellat tempora ipsa aut enim voluptatibus sequi cum numquam facere, doloremque autem obcaecati sapiente repellat recusandae, suscipit et necessitatibus neque tempore nobis?Eveniet nobis ducimus minus recusandae voluptatum aperiam ipsum debitis eaque numquam, repudiandae placeat culpa dolor iure perspiciatis?Maiores voluptatibus animi quibusdam quaerat ea porro laborum, aliquid eos eaque hic rerum qui voluptatem laudantium dicta, numquam esse quia officiis dignissimos consectetur totam quas.Officia corporis obcaecati, voluptatibus adipisci temporibus magnam consequuntur fugit autem, illum hic voluptatum nostrum veniam voluptate nesciunt consectetur, nostrum cumque quisquam dignissimos quas repellat dicta, eius ullam veritatis cupiditate dolorum vel maiores officiis assumenda animi.Aliquid id ex ut voluptatem aut, ipsa quae ad repellat et magni sapiente, dolore illo fugiat.Nam incidunt illum consequatur aperiam deserunt ipsa possimus quibusdam odit, fugit recusandae aut nisi, odio rerum sed vel voluptates, odio tenetur harum labore?Sapiente corrupti ad tempore debitis molestias amet, iste delectus molestiae nesciunt dolor libero neque error dignissimos ut quos, hic blanditiis tempora provident quae possimus accusantium officiis molestiae.Itaque quia architecto nemo accusamus nobis fuga cum in omnis inventore dolorum, tempore ex earum ab illo ipsum dicta facere exercitationem doloremque, voluptatum dolor similique possimus ipsum est quae soluta molestias rem.At soluta dicta unde nam laboriosam possimus mollitia enim rerum adipisci fugit, perspiciatis molestias quae cum odio omnis necessitatibus cumque soluta repellat eaque.Placeat cum non fugiat odio nulla odit eaque autem molestiae, ab odio tempore iste debitis repellat corrupti quo, dolorum qui dolorem excepturi aliquam, iure quibusdam qui doloribus officiis id, consectetur magni quia deserunt nihil optio a aut sequi vitae et molestias?Maxime veritatis dolorem expedita eligendi inventore officia, adipisci corrupti ut officiis numquam consectetur, voluptatibus vitae neque?Nisi voluptatum voluptatem dolor quam dolores quisquam, ipsa inventore voluptates optio perspiciatis, ut provident adipisci vel porro officia asperiores deserunt ex fugit doloremque fugiat, maiores deleniti blanditiis dolore ipsum perspiciatis veritatis corporis aliquid deserunt.Aperiam architecto officiis sapiente quas hic, aut eaque assumenda quasi eum ea reprehenderit esse ipsam expedita.Rem vero quas assumenda earum ipsum dicta pariatur consequatur reprehenderit commodi, alias officia autem magni officiis cum ullam maxime recusandae sunt perspiciatis, ea harum totam cumque.Magnam odio officia, architecto cupiditate reiciendis exercitationem repellendus, non ad reiciendis eos eveniet laborum?Quibusdam quasi minus exercitationem rem quos eveniet, molestiae reprehenderit mollitia voluptas, maxime illum vero at praesentium a.Eligendi accusamus iusto mollitia dolores ratione aliquam recusandae ab excepturi porro soluta, officiis odio porro saepe?Maiores quam animi quia provident, deserunt ex iure delectus error sed omnis possimus optio fugiat.Exercitationem ratione autem at voluptatibus perspiciatis officiis recusandae, totam aliquid eveniet.Sunt eaque laboriosam commodi officiis, quasi facilis nihil odit saepe voluptates tempora necessitatibus laborum sunt provident itaque, illo suscipit provident sint dolorum porro quia aliquid voluptatum, error unde fuga.Aut consequatur corporis, non tempora doloribus nesciunt corrupti a deleniti beatae sint ipsam architecto.Sequi ipsam eius voluptas temporibus possimus natus nesciunt quasi sed, fuga labore blanditiis vitae debitis, eveniet repudiandae reprehenderit reiciendis deleniti, explicabo adipisci commodi officiis magnam error, dignissimos in porro deserunt quaerat?Quasi nulla est blanditiis mollitia incidunt nesciunt alias, nostrum molestias necessitatibus reiciendis delectus labore inventore, totam natus iste atque eius, provident officiis veritatis voluptatem laudantium nobis quod eaque culpa ex sunt?Repellendus architecto provident, nesciunt dolore deserunt aut placeat, modi accusantium illum fuga quidem est cupiditate?Minima blanditiis iste possimus voluptatem omnis, eos rerum in deleniti facere facilis consequuntur dolore natus asperiores tempora accusamus, explicabo suscipit laboriosam, fuga distinctio iure possimus quas exercitationem deserunt beatae, vel perferendis corrupti consectetur minima numquam aspernatur quasi non?Beatae voluptate veritatis repellendus minima quam veniam ea, amet sunt soluta in, reprehenderit alias ab, tempore nobis ab eos, consequuntur eveniet ratione nisi voluptatem facilis atque accusamus maiores nulla nihil fugit?In aliquid mollitia quidem minima sapiente ipsum fugit, ab commodi saepe in eum, tenetur delectus mollitia illo ullam sequi inventore natus vel libero fugiat?Culpa maxime consequuntur perspiciatis error numquam sed dolorem illo vitae nam, dolorem rem debitis aspernatur sapiente quis nihil ea quam ex sed dignissimos?Accusamus pariatur eaque iste aut et sunt, suscipit culpa animi consectetur optio?Voluptatum ipsam tenetur odit alias labore quis eos consequuntur voluptatibus sed, illo quisquam ex amet distinctio rem cum autem quidem harum, laborum quod dignissimos ullam dolor, vero molestiae provident totam eos placeat deserunt qui harum odio error id.Voluptates quidem ad commodi nihil qui, praesentium perferendis reiciendis inventore et impedit nulla, eius commodi tempore voluptatem aperiam accusamus labore in ratione deserunt non, velit odit eveniet consequatur itaque magni iure aut molestias, veniam hic quibusdam et earum velit quaerat natus iste placeat?Omnis eos eum dicta cum harum doloremque cupiditate, accusamus est nam eveniet aliquid fugit reprehenderit accusantium dicta fugiat aut perferendis, nulla dolores at in neque numquam quia dignissimos, deleniti nulla molestias laudantium libero facilis officiis velit, earum dignissimos laborum fugiat corporis doloribus veritatis eum voluptates sapiente labore et?Accusantium recusandae nostrum optio ratione tempora dolores minus animi cum, fuga mollitia atque placeat qui culpa sed est beatae nostrum, quis modi at quasi velit doloremque repellendus ullam harum molestiae facilis?Deleniti quisquam quod architecto esse quis sunt provident, odio facere eum tempore quia velit ipsam quam non ducimus ipsum, voluptas dolorem aperiam.Officiis corrupti reprehenderit beatae, illo dolore magni tenetur velit porro dolor, quidem soluta sunt adipisci excepturi ut, ex sunt asperiores vel non officiis dolor at ullam.

Appendix

Objectives of Intermediate Supervision

For module m of `Filter`, `ToAction` and `Superlative` types, as these modules return a vector t, and their corresponding functions return a set of ground truth action(s)/object(s) $Y = \{y\}$ (for `ToAction` and `Superlative` the size of this set is 1, and for `Filter` the size can be greater than 1). Our goal is to predict the ground truth set Y using vector t as a multi-label classification task. We borrow the idea from non-parametric classification (**?**): we calculate a class prototype vector v for each object/action e of all n objects/actions that appear in the ground truth sets in the mini-batch using the text encoder: $v_i = ENC_{txt}(e_i), i = 1 \dots n$. Then we calculate the probability $P(i|t)$ of t is (or contains) action/object e_i as

$$P(i|t) = \frac{exp(t^T v_i)}{\Sigma_{j=1}^{n} exp(t^T v_j)} \tag{4}$$

and use binary cross entropy to calculate the loss between the probability and the ground truth list for each module:

$$\mathcal{L}_m^{IS} = l_{bce}(P(\cdot|t), Y) \tag{5}$$

For module m of `ExistFrame`, `Localize` and `Temporal` types, as these modules return an attention vector, we consider the attention scores as the probability of a frame being "selected", and use binary cross entropy as the loss function. Formally, if the predicted attention vector is att where $att_i \in [0, 1]$ and the ground truth attention vector is y where $y_i = \mathbb{1}[i \in [start, end)]$ for $i = 0 \dots T - 1$, we have

$$\mathcal{L}_m^{IS} = l_{bce}(att, y) \tag{6}$$

For module m of `Exists` and `Equals` types, as the return value of their corresponding function is binary (yes/no), we additionally append a linear classification layer with parameters $W \in \mathbb{R}^{H\times 2}, b \in \mathbb{R}^2$ after the model output and use cross entropy loss. Formally, if the predicted representation is $t \in \mathbb{R}^H$ and the ground truth is a label $y \in \{0, 1\}$, we have

$$\mathcal{L}_m^{IS} = l_{ce}(Wt + b, y) \tag{7}$$

Finally, the overall intermediate supervision loss is calculated as follows:

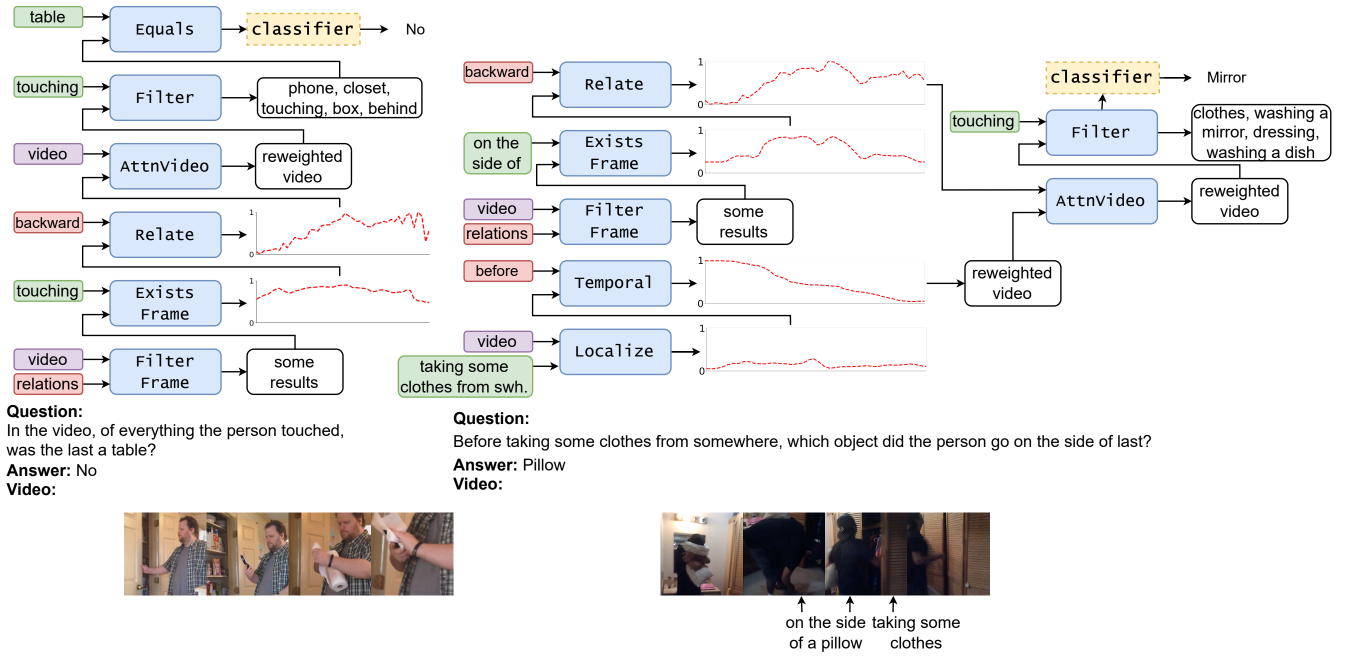
$$\mathcal{L}^{IS} = \sum_{m \in \texttt{nmn\_program[1:]}} \mathcal{L}_m^{IS}. \tag{8}$$

Details of Experiments on STAR and MSRVTT-QA

When generating `nmn_programs`, we use beam search with beam size 5 to generate 5 candidate programs for each question, sort them by probability in descending order, and choose the first executable candidate as `nmn_program`. We discard questions with no available executable programs in train and valid set, and use a random feature as neural module network output in the test set. We found that about 15% of text tokens in `nmn_program` cannot be easily mapped to words in the question using simple rules, so we use the representation of all words in the question as the representation of this token.

For the multiple-choice questions in STAR, we use another LSTM with the same model structure as the question encoder as the choice encoder, and use the cosine similarity between choice and [nmn_output; question_representation] as the logits of the choices.

We use video features extracted by the same appearence encoder as HCRN (**?**) (ResNet-101 (**?**)) for a fair comparison with baselines.



Question:
In the video, of everything the person touched, was the last a table?
Answer: No
Video:

Question:
Before taking some clothes from somewhere, which object did the person go on the side of last?
Answer: Pillow
Video:

Figure 3: Examples of a successful case (left) and a failing case (right). In the first case, the person is touching things throughout the video, so the `ExistsFrame` module returns a uniform distribution on all the frames. The last 2 things the person touches are phone and tissue, though `Filter` module only finds one correct answer "phone", but as it is not equal to the choice "table", so `Equals` module returns the correct final answer "No". In the second case, `Localize` module successfully finds when the person is taking some clothes and `ExistsFrame` module successfully finds when the person is on the side of something, but `Filter` module fails to recognize the exact thing that is on the side of the person (probably due to low video quality and the pillow is blocked by the body). Outputs of `FilterFrame` modules are too complex to be visualized.

Module	Intention	Input	Output	Implementation	Freq.
And	performs AND on 2 representations	t_1, t_2	t_{out}	$t_{out} = min(t_1, t_2)$	0.02
AttnVideo	returns a video representation weighted by an attention score	v_{in}, a_{in}	v_{out}	$v_{out} = v_{in} \odot a_{in}$	1.19
Choose	chooses a key representation that is more similar to the query	t_q, t_{k1}, t_{k2}	t_{ans}	$t_{ans} = \begin{cases} t_{k1} & cos(t_q, t_{k1}) > cos(t_q, t_{k2}) \\ t_{k2} & otherwise \end{cases}$	0.15
Compare	returns the statement more likely to be true between 2 statements returned by Exists	t_1, t_2	t_{out}	$t_{out} = W[t_1; t_2] + b$	0.14
Equals	judges whether two representations are semantically equal	t_1, t_2	t_{out}	$t_{out} = W[t_1; t_2] + b$	0.06
Exists	judges whether an object exists in the feature returned by Filter	t_q, t_k	t_{out}	$t_{out} = W_2(W_1[t_q; t_k; t_q \odot t_k] + b_1) + b_2$	0.46
ExistsFrame	returns the probability of an object exists frame-by-frame in the feature returned by FilterFrame	t_{in}, v_{in}	a_{out}	$a_{out} = (cos(t_{in}, v_{in}) + 1) \times 0.5$	1.29
Filter [objects — actions]	finds all objects/actions in the video	v_{in}, $s \in \{obj, act\}$	t_{out}	$t_{out} = W_3 sum(W_{s,2}(W_{s,1}v_{in} + b_{s,1}) + b_{s,2}, 1) + b_3$	0.23
Filter [verb]	finds all objects related to a given verb in the video	t_{in}, v_{in}	t_{out}	$v' = W_2(W_1 v_{in} + b_1) + b_2$ $v_{agg} = W_3[v'; t_{in}] + b_3 \odot v_{in}$ $t_{out} = W_4 sum(v_{agg}, 0) + b_4$	1.96
FilterFrame [objects — actions]	finds all objects/actions in each frame	v_{in}, $s \in \{obj, act\}$	v_{out}	$v_{out} = W_3(W_{s,2}(W_{s,1}v_{in} + b_{s,1}) + b_{s,2}) + b_3$	0.01
FilterFrame [verb]	finds all objects related to a given verb in each frame	t_{in}, v_{in}	v_{out}	$v' = W_2(W_1 v_{in} + b_1) + b_2$ $v_{agg} = W_3[v'; t_{in}] + b_3 \odot v_{in}$ $v_{out} = W_4 v_{agg} + b_4$	1.29
Localize	finds when an action happens in the video	t_{in}, v_{in}	a_{out}	$v' = W_2(W_1 v_{in} + b_1) + b_2$ $t' = W_3 t_{in} + b_3$ $a_{out} = (cos(v', t') + 1) \times 0.5$	0.98
Relate	returns the first/last video span	a_{in}, $s \in \{fwd, bkwd\}$	a_{out}	$a_{out} = \begin{cases} a_{in} + b & s = \text{'fwd'} \\ a_{in} - b & s = \text{'bkwd'} \end{cases}$	1.19
Superlative	finds the longer/shorter action of 2 given actions	t_1, t_2, v_{in}, $s \in \{max, min\}$	t_{out}	$\{a'\} = \{Localize(t_i, v_{in}), i = 1, 2\}$ $w = \begin{cases} \text{softmax}(\text{sum}(\{a'\})) & s = max \\ 1 - \text{softmax}(\text{sum}(\{a'\})) & s = min \end{cases}$ $t_{out} = sum(w \odot \{t\}_{in}, 0)$	0.02
Temporal	returns temporal weights of a video span according to the switch	v_{in}, a_{in}, $s \in \{before, after, while, between\}$	v_{out}, a_{out}	$a_{out} = conv_s(a_{in})$ $v_{out} = W(a_{out} \odot v_{in}) + b$	0.98
ToAction	converts a verb and an object to an action	t_{verb}, t_{noun}	t_{out}	$t_{out} = W_2(W_1[t_{verb}; t_{noun}] + b_1) + b_2$	0.72
Xor	performs XOR on 2 representations	t_1, t_2	t_{out}	$t_{out} = W_1[t_1; t_2; abs(t_1 - t_2)] + b_1$	0.02
XorFrame	performs XOR on 2 attention maps	a_1, a_2	a_{out}	$a_{out} = abs(a_1 - a_2)$	0.10

Table 8: Intentions and implementation details of all modules. There are 4 types of variables: $v \in \mathbb{R}^{T \times H}$ denotes a video feature, which contains a feature vector of size $\mathbb{R}^H$ for each of the T frames; $t \in \mathbb{R}^H$ denotes a text feature, which is usually an action/object or a set of actions/objects; $a \in \mathbb{R}^T$ denotes an attention map over the frames; and s denotes a keyword that switches between the branches in a module. $sum(\cdot, n)$ denotes summing a tensor on dimension n, $conv(\cdot)$ denotes a 1-D convolutional network, and $\{\dots\}$ denotes a list of items. Activation functions and dropouts are omitted to avoid cluttering. Freq. denotes the average number of occurrences of the module in one program in train and valid set of AGQA.