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 (Yi et al. 2018) 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 (Mao et al. 2019) 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 (Johnson et al. 2016) and Minecraft (Wu, Tenenbaum, and Kohli 2017; Yi et al. 2018), they can not perform well on real-world images. One promising neural-symbolic approach is Neural Module Networks (NMNs) (Andreas et al. 2015). 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 (Agrawal et al. 2015), 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 (Grunde-McLaughlin, Krishna, and Agrawala 2022) question-program pairs (which is publicly available) can generate plausible programs for questions from other

[*]Corresponding Author

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 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 (Grunde-McLaughlin, Krishna, and Agrawala 2021, 2022), 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 (Wu et al. 2021) and MSRVTT-QA (Gao et al. 2018) 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 (Zhang et al. 2019; Li et al. 2019; Kumar et al. 2019) that adopt spatial and/or temporal attention to fuse information from question and video; (2) **Memory network based** methods (Xu et al. 2017; Gao et al. 2018; Fan et al. 2019; Kim et al. 2019) that use recurrent read and write operations to process video and question features; (3) **Graph based** methods (Jin et al. 2021; Seo et al. 2021; Xiao et al. 2021; Cherian et al. 2022; Park, Lee, and Sohn 2021; Zhao et al. 2022) that process videos as (usually object level) graphs and use graph neural networks to obtain informative video representations; and (4) **Pre-trained models** (Lei et al. 2021; Fu et al. 2021; Zellers et al. 2021, 2022; Wang et al. 2023) that pre-train a model in self-supervised manner with a mass of video-text multimodal data. Recently, many works also try to solve video QA in zero-shot settings using large pre-trained transformer-based models (Alayrac et al. 2022; Li et al. 2023; Zhang, Li, and Bing 2023; Lyu et al. 2023). 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 (Kim et al. 2020; Gao et al. 2022; Li et al. 2022) 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 (Yi et al. 2019) (Qian et al. 2022) or construct physics models (Ding et al. 2021; Chen et al. 2021).

Neural Module Networks. Neural Module Networks (NMN) have been widely used in image question answering (Andreas et al. 2015; Hu et al. 2017; Johnson et al. 2017; Mascharka et al. 2018; Hu et al. 2018). 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 (Andreas et al. 2015) and CLEVR (Johnson et al. 2016). A more advanced NMN for image-text tasks is the recently-proposed Visual Programming (Gupta and Kembhavi 2023). Taking advantage of several off-the-shelf models such as CLIP (Radford et al. 2021), GPT-3 (Brown et al. 2020) and Stable Diffusion (Rombach et al. 2021), 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 (Le, Chen, and Hoi 2022; Qian et al. 2022). 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 (Qian et al. 2022), 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.

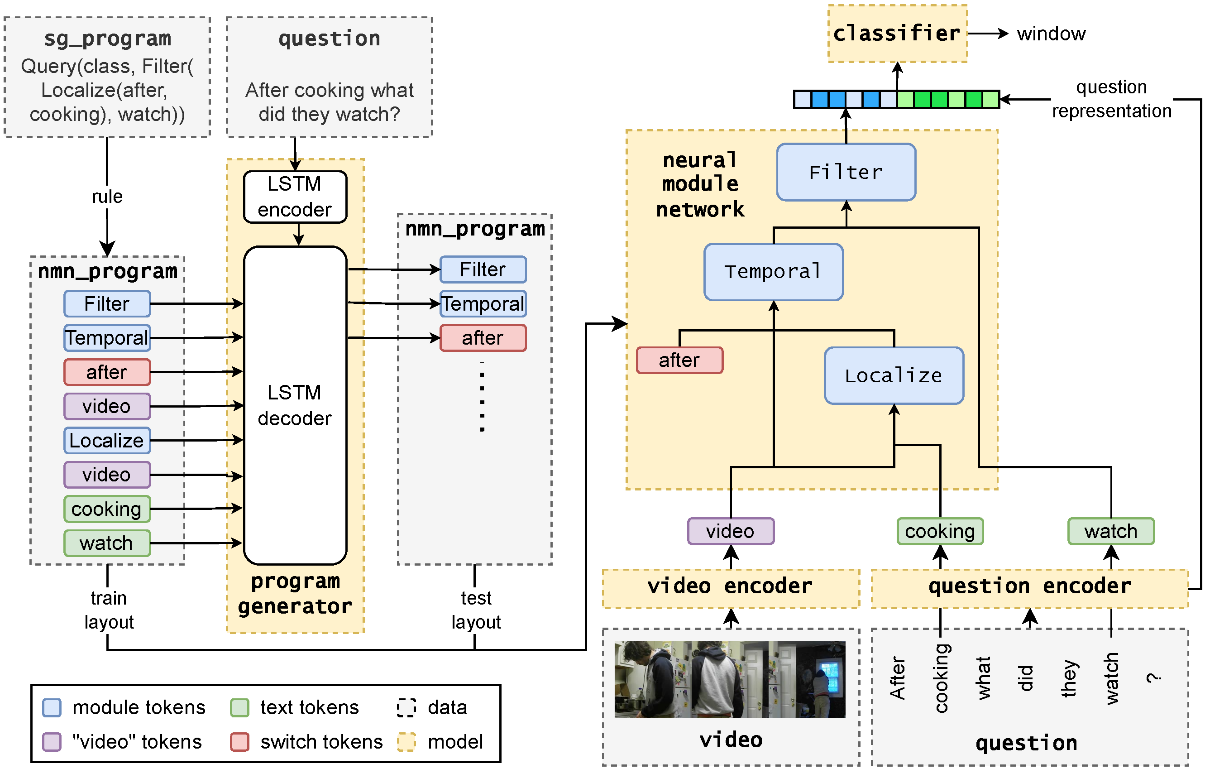

Figure 1: Overview of STAIR.

Methodology

In this section, we describe the details of STAIR. STAIR takes as input a video feature $x_v \in R^{T \times hid_V}$ 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 $v = ENC_{vid}(x_v), v \in R^{T \times H}$; (2) a bi-directional LSTM **text encoder** ENC_{txt} which extracts the sentence-level and token-level question feature as $(q, t) = ENC_{txt}(x_q), q \in R^H, t \in R^{L \times H}$; (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 **program 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 con-

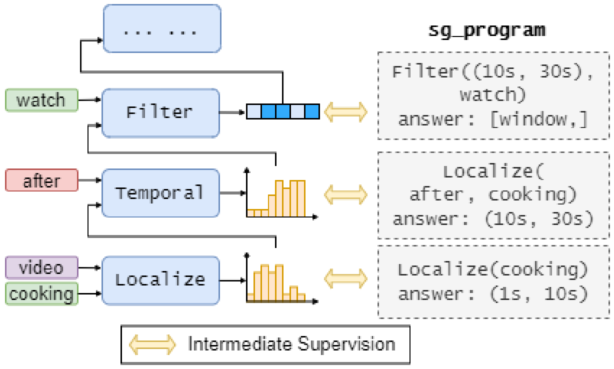

Figure 2: A Diagram of Intermediate Supervision.

sisting 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 (Sigurdsson et al. 2016; Ji et al. 2019). 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** 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 (Wei et al. 2022), 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 (Hu et al. 2017). 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.

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 (Grunde-McLaughlin, Krishna, and Agrawala 2021), 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

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

[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.

with a program that describes the reasoning steps to answer the questions. Videos in AGQA are from Charades (Sigurdsson et al. 2016), 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 (Ji et al. 2019). 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 (Grunde-McLaughlin, Krishna, and Agrawala 2022) which contains 2.27M question-answer pairs selected with a stricter balancing procedure and is even more challenging than AGQA. Following (Le et al. 2020), 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 $x_v^a \in R^{8\times16\times2048}$ extracted from ResNet-101 pool5 layer(He et al. 2015), and motion features $x_v^m \in R^{8\times2048}$ extracted from ResNeXt-101(Xie et al. 2016). 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 $x_v \in R^{8\times4096}$. 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 pretrained on Kinetics (Carreira and Zisserman 2017) 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 $x_v \in R^{T\times1024}, T \leq 64$. 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** (Fan et al. 2019) is a memory-network-based model to encode video and text features; **HCRN** (Le et al. 2020) uses conditional relational networks to build a hierarchical structure that learns video representation on both clip level and video level; **PSAC** (Li et al. 2019) 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** (Qian et al. 2022), a neural module network concurrent to our work, and **MAC** (Hudson and Manning 2018) 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	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 (Qian et al. 2022). #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.

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

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.

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 $att_p, att_g \in R^T$, 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 (Qian et al. 2022), 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** (Radford et al. 2019), and a video-text pre-trained model **Violet** (Fu et al. 2021). For GPT-2, we prepend I3D video features to questions and assign different token type embeddings following (Li et al. 2020). 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 $\mathbf{P}$ modules into account, where $\mathbf{P}$ 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** (Maaz et al. 2023), 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 (Maaz et al. 2023), 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 (Wu et al. 2021) and MSRVTT-QA (Xu et al. 2017). We changed the program generator from an LSTM to a FLAN-T5-large (Wei et al. 2022) 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

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 (Wu et al. 2021)

types of questions which requires spatial and temporal reasoning. Results on MSRVTT-QA shows that STAIR is also applicable to noisy, automatically-generated questions (Lin et al. 2022). 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 (Mao et al. 2019)), better functional and structural designs of the neural modules (e.g., using more powerful pre-trained models), and applying on more video-text tasks other than QA.

Acknowledgments

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

Methods	MSRVTT-QA
Co-Memory (Gao et al. 2018)	32.0
HME (Fan et al. 2019)	33.0
HCRN (Le et al. 2020)	**35.6**
STAIR	34.8
ClipBERT (Lei et al. 2021)	37.4

Table 7: Accuracy on MSRVTT-QA test set.

References

Agrawal, A.; Lu, J.; Antol, S.; Mitchell, M.; Zitnick, C. L.; Parikh, D.; and Batra, D. 2015. VQA: Visual Question Answering. *International Journal of Computer Vision*, 123: 4–31.

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; Ring, R.; Rutherford, E.; Cabi, S.; Han, T.; Gong, Z.; Samangooei, S.; Monteiro, M.; Menick, J.; Borgeaud, S.; Brock, A.; Nematzadeh, A.; Sharifzadeh, S.; Binkowski, M.; Barreira, R.; Vinyals, O.; Zisserman, A.; and Simonyan, K. 2022. Flamingo: a Visual Language Model for Few-Shot Learning. *ArXiv*, abs/2204.14198.

Andreas, J.; Rohrbach, M.; Darrell, T.; and Klein, D. 2015. Neural Module Networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 39–48.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T. J.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. *ArXiv*, abs/2005.14165.

Carreira, J.; and Zisserman, A. 2017. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4724–4733.

Chen, Z.; Mao, J.; Wu, J.; Wong, K.-Y. K.; Tenenbaum, J. B.; and Gan, C. 2021. Grounding Physical Concepts of Objects and Events Through Dynamic Visual Reasoning. In *International Conference on Learning Representations*.

Cherian, A.; Hori, C.; Marks, T. K.; and Roux, J. L. 2022. (2.5+1)D Spatio-Temporal Scene Graphs for Video Question Answering. In *AAAI Conference on Artificial Intelligence*, 444–453.

Ding, M.; Chen, Z.; Du, T.; Luo, P.; Tenenbaum, J. B.; and Gan, C. 2021. Dynamic Visual Reasoning by Learning Differentiable Physics Models from Video and Language. In *Neural Information Processing Systems*.

Fan, C.; Zhang, X.; Zhang, S.; Wang, W.; Zhang, C.; and Huang, H. 2019. Heterogeneous Memory Enhanced Multimodal Attention Model for Video Question Answering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999–2007.

Fu, T.-J.; Li, L.; Gan, Z.; Lin, K.; Wang, W. Y.; Wang, L.; and Liu, Z. 2021. VIOLET : End-to-End Video-Language Transformers with Masked Visual-token Modeling. *ArXiv*, abs/2111.12681.

Gao, D.; Zhou, L.; Ji, L.; Zhu, L.; Yang, Y.; and Shou, M. Z. 2022. MIST: Multi-modal Iterative Spatial-Temporal Transformer for Long-form Video Question Answering. *ArXiv*, abs/2212.09522.

Gao, J.; Ge, R.; Chen, K.; and Nevatia, R. 2018. Motion-Appearance Co-memory Networks for Video Question Answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6576–6585.

Grunde-McLaughlin, M.; Krishna, R.; and Agrawala, M. 2021. AGQA: A Benchmark for Compositional Spatio-Temporal Reasoning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11282–11292.

Grunde-McLaughlin, M.; Krishna, R.; and Agrawala, M. 2022. AGQA 2.0: An Updated Benchmark for Compositional Spatio-Temporal Reasoning. *ArXiv*, abs/2204.06105.

Gupta, T.; and Kembhavi, A. 2023. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14953–14962.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Hu, R.; Andreas, J.; Darrell, T.; and Saenko, K. 2018. Explainable Neural Computation via Stack Neural Module Networks. In *European Conference on Computer Vision*.

Hu, R.; Andreas, J.; Rohrbach, M.; Darrell, T.; and Saenko, K. 2017. Learning to Reason: End-to-End Module Networks for Visual Question Answering. *2017 IEEE International Conference on Computer Vision (ICCV)*, 804–813.

Hudson, D. A.; and Manning, C. D. 2018. Compositional Attention Networks for Machine Reasoning. In *International Conference on Learning Representations*.

Ji, J.; Krishna, R.; Fei-Fei, L.; and Niebles, J. C. 2019. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10233–10244.

Jin, W.; Zhao, Z.; Cao, X.; Zhu, J.; He, X.; and Zhuang, Y. 2021. Adaptive Spatio-Temporal Graph Enhanced Vision-Language Representation for Video QA. *IEEE Transactions on Image Processing*, 30: 5477–5489.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. B. 2016. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1988–1997.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Hoffman, J.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. B. 2017. Inferring and Executing Programs for Visual Reasoning. *2017 IEEE International Conference on Computer Vision (ICCV)*, 3008–3017.

Kim, J.; Ma, M.; Kim, K.; Kim, S.; and Yoo, C. D. 2019. Progressive Attention Memory Network for Movie Story Question Answering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8329–8338.

Kim, J.; Ma, M.; Pham, T. X.; Kim, K.; and Yoo, C. D. 2020. Modality Shifting Attention Network for Multi-Modal Video Question Answering. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10103–10112.

Kumar, S. H.; Okur, E.; Sahay, S.; Huang, J.; and Nachman, L. 2019. Leveraging Topics and Audio Features with Multimodal Attention for Audio Visual Scene-Aware Dialog. *ArXiv*, abs/1912.10131.

Le, H.; Chen, N. F.; and Hoi, S. C. H. 2022. VGNMN: Video-grounded Neural Module Networks for Video-Grounded Dialogue Systems. In *North American Chapter of the Association for Computational Linguistics*.

Le, T. M.; Le, V.; Venkatesh, S.; and Tran, T. 2020. Hierarchical Conditional Relation Networks for Video Question Answering. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9969–9978.

Lei, J.; Li, L.; Zhou, L.; Gan, Z.; Berg, T. L.; Bansal, M.; and Liu, J. 2021. Less is More: CLIPBERT for Video-and-Language Learning via Sparse Sampling. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7327–7337.

Li, K.; He, Y.; Wang, Y.; Li, Y.; Wang, W.; Luo, P.; Wang, Y.; Wang, L.; and Qiao, Y. 2023. VideoChat: Chat-Centric Video Understanding. *ArXiv*, abs/2305.06355.

Li, X.; Song, J.; Gao, L.; Liu, X.; Huang, W.; He, X.; and Gan, C. 2019. Beyond RNNs: Positional Self-Attention with Co-Attention for Video Question Answering. In *AAAI Conference on Artificial Intelligence*.

Li, Y.; Wang, X.; Xiao, J.; Ji, W.; and Chua, T.-S. 2022. Invariant Grounding for Video Question Answering. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2918–2927.

Li, Z.; Li, Z.; Zhang, J.; Feng, Y.; Niu, C.; and Zhou, J. 2020. Bridging Text and Video: A Universal Multimodal Transformer for Audio-Visual Scene-Aware Dialog. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29: 2476–2483.

Lin, X.; Tiwari, S.; Huang, S.; Li, M.; Shou, M. Z.; Ji, H.; and Chang, S.-F. 2022. Towards Fast Adaptation of Pretrained Contrastive Models for Multi-channel Video-Language Retrieval. *ArXiv*, abs/2206.02082.

Lyu, C.; Wu, M.; Wang, L.; Huang, X.; Liu, B.; Du, Z.; Shi, S.; and Tu, Z. 2023. Macaw-LLM: Multi-Modal Language Modeling with Image, Audio, Video, and Text Integration. *ArXiv*, abs/2306.09093.

Maaz, M.; Rasheed, H.; Khan, S.; and Khan, F. S. 2023. Video-ChatGPT: Towards Detailed Video Understanding via Large Vision and Language Models. *ArXiv*, abs/2306.05424.

Mao, J.; Gan, C.; Kohli, P.; Tenenbaum, J. B.; and Wu, J. 2019. The Neuro-Symbolic Concept Learner: Interpreting Scenes Words and Sentences from Natural Supervision. *ArXiv*, abs/1904.12584.

Mascharka, D.; Tran, P.; Soklaski, R.; and Majumdar, A. 2018. Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4942–4950.

Park, J.; Lee, J.; and Sohn, K. 2021. Bridge to Answer: Structure-aware Graph Interaction Network for Video Question Answering. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15521–15530.

Qian, Z.; Wang, X.; Duan, X.; Chen, H.; and Zhu, W. 2022. Dynamic Spatio-Temporal Modular Network for Video Question Answering. *Proceedings of the 30th ACM International Conference on Multimedia*.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2021. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752.

Seo, A.; Kang, G.-C.; Park, J.; and Zhang, B.-T. 2021. Attend What You Need: Motion-Appearance Synergistic Networks for Video Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 6167–6177.

Sigurdsson, G. A.; Varol, G.; Wang, X.; Farhadi, A.; Laptev, I.; and Gupta, A. 2016. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 510–526. Springer.

Wang, Y.; Zheng, Z.; Zhao, X.; Li, J.; Wang, Y.; and Zhao, D. 2023. VSTAR: A Video-grounded Dialogue Dataset for Situated Semantic Understanding with Scene and Topic Transitions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5036–5048.

Wei, J.; Bosma, M.; Zhao, V.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2022. Finetuned Language Models are Zero-Shot Learners.

Wu, B.; Yu, S.; Chen, Z.; Tenenbaum, J.; and Gan, C. 2021. STAR: A Benchmark for Situated Reasoning in Real-World Videos. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS)*.

Wu, J.; Tenenbaum, J. B.; and Kohli, P. 2017. Neural scene de-rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 699–707.

Xiao, J.; Yao, A.; Liu, Z.; Li, Y.; Ji, W.; and Chua, T.-S. 2021. Video as Conditional Graph Hierarchy for Multi-Granular Question Answering. In *AAAI Conference on Artificial Intelligence*.

Xie, S.; Girshick, R. B.; Dollár, P.; Tu, Z.; and He, K. 2016. Aggregated Residual Transformations for Deep Neural Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5987–5995.

Xu, D.; Zhao, Z.; Xiao, J.; Wu, F.; Zhang, H.; He, X.; and Zhuang, Y. 2017. Video Question Answering via Gradually Refined Attention over Appearance and Motion. *Proceedings of the 25th ACM international conference on Multimedia*.

Yi, K.; Gan, C.; Li, Y.; Kohli, P.; Wu, J.; Torralba, A.; and Tenenbaum, J. B. 2019. CLEVRER: Collision Events for Video Representation and Reasoning. In *International Conference on Learning Representations*.

Yi, K.; Wu, J.; Gan, C.; Torralba, A.; Kohli, P.; and Tenenbaum, J. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *Advances in Neural Information Processing Systems*, 31.

Zellers, R.; Lu, J.; Lu, X.; Yu, Y.; Zhao, Y.; Salehi, M.; Kusupati, A.; Hessel, J.; Farhadi, A.; and Choi, Y. 2022. MERLOT RESERVE: Neural Script Knowledge through Vision and Language and Sound. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16354–16366.

Zellers, R.; Lu, X.; Hessel, J.; Yu, Y.; Park, J. S.; Cao, J.; Farhadi, A.; and Choi, Y. 2021. MERLOT: Multimodal Neural Script Knowledge Models. In *Neural Information Processing Systems*.

Zhang, H.; Li, X.; and Bing, L. 2023. Video-LLaMA: An Instruction-tuned Audio-Visual Language Model for Video Understanding. *ArXiv*, abs/2306.02858.

Zhang, Z.; Zhao, Z.; Lin, Z.; Song, J.; and He, X. 2019. Open-Ended Long-Form Video Question Answering via Hierarchical Convolutional Self-Attention Networks. In *International Joint Conference on Artificial Intelligence*.

Zhao, X.; Wang, Y.; Tao, C.; Wang, C.; and Zhao, D. 2022. Collaborative Reasoning on Multi-Modal Semantic Graphs for Video-Grounded Dialogue Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 5988–5998.