Enhancing Visual Language Models with Logic Reasoning for Situational Awareness

Anonymous Authors, PaperID 5433

Anonymous Institute

Abstract. Vision Language Models (VLMs) can provide natural language descriptions of complex activities from images and videos. However, VLMs cannot isolate individual objects and only provide a generic caption for (or description of) the scene, making informative fine-tuning difficult. This paper proposes a novel fine-tuning mechanism that uses traditional computer vision techniques to recognize more straightforward proxy activities corresponding to the more complex activities for which the VLM is fine-tuned. Thus, by creating multiple fine-tuned VLMs for correlated activities and using explicit logic reasoning, we can estimate inconsistencies between them and conduct multi-step directed fine-tuning across them. Experiments with several VLMs (including those that operate on images and videos) and two very different video datasets (road traffic and taekwondo) show that our approach consistently increases the VLM accuracy by about 20 percentage points beyond that is achieved via undirected fine-tuning. The mechanism is very general and can be exploited to justify VLM output during inferencing.

Keywords: Vision Large Language Models · Logic Reasoning · Object recognition/tracking · Satisfiability Modulo Theories

1 Introduction

Video-driven Visual Language Models (VLMs) have recently been developed to effectively summarize the content of an image or short video at an advanced level. Many VLMs have recently been put forward, including Clip [30], X-Clip [23], Video-LLAMA [40], LLAVA [21], MiniGPT [41], VideoMAE [33], and Video-chatGPT [24]. Some of these work only with images (e.g., MiniGPT4, LLAVA, Clip), while others work with (short) videos (e.g., Video-LLAMA, Video-ChatGPT, X-Clip, VideoMAE). VLMs are generally trained on a huge amount of available video data and text captions. Most VLMs (excluding Clip and X-Clip) have been integrated with the Large Language Models (LLMs) on the backend to support detailed Q&A capability and lucid descriptions of what is happening in the image/video. These descriptions can provide rich descriptions (e.g., type of venue where the activity occurs), which goes well beyond what is reasonably possible using Traditional Computer Vision (TCV) techniques without extensive, application-specific training.

In this paper, we propose a novel fine-tuning mechanism for VLMs by exploiting logical reasoning along with TCV that can substantially improve their performance on the targeted tasks. Our approach uses three key ideas to accomplish this. First, instead of only tuning a VLM for the targeted task, we also fine-tune one (or more) additional copies of the VLM on correlated tasks. Second, we use TCV to identify the objects involved and track them, thereby enabling the representation of VLM output in terms of concrete logic assertions. Third, we set up logic assertions to detect (a) consistency between the VLMs based on the task correlation and (b) consistency between VLM outputs and TCV regarding the identified objects. It is thus possible to use standard logic reasoning tools to detect inconsistencies, which we exploit to choose the videos/images representing classes (or situations) where the VLM performs poorly. These videos/images are then used to fine-tune the VLMs further to improve their discrimination ability. The essential advantage of the mechanism is two-fold. First, it eases the burden of selecting and labeling videos/images for fine-tuning. Second, it reduces the resource requirements of fine-tuning, which can be pretty substantial. The process can be repeated until the accuracy has reached the limit dictated by the aleatoric uncertainty, data availability, or other considerations.

By comparing our directed fine-tuning mechanism against the undirected one, we demonstrate a consistent improvement in accuracy by a huge 20 percentage points, i.e., 70-80% achieved accuracy with directed tuning as opposed to 50-60% with undirected tuning. We show that this differential applies with both image-based VLMs such as Minigpt4 [41] and video-based VLMs such as XClip [23] and Video-MAE [35]. We also show that the improvement is sustained for two datasets, one relating to road traffic and traffic accidents and the other to the Taekwondo classroom. Furthermore, the mechanism is general and can be extended in several directions, as discussed in section 5. One exciting use of this mechanism is to provide justifications for the VLM output in the form of the results of the consistency checks. If the checks pass, we justify why the result can be trusted; if not, we indicate that we do not trust the results. *To the best of our knowledge, this is the first work of its type to integrate explicit logic reasoning with computer vision to improve the fine-tuning of VLMs.*

The rest of this paper is organized as follows. Section 2 discussed the background and related work. Section 3 presents the detailed design of our directed fine-tuning mechanism. Section 4 discusses the experimental assessment of the mechanism. Finally, section 5 concludes the discussion.

2 Methodology and Related Work

2.1 Fine-Tuning Vision Based Language Models

Despite their rapidly increasing popularity, VLMs (and, more generally, LLMs) suffer many challenges. They require significant resources even to run and far more resources to fine-tune. Furthermore, VLMs usually are not very good at the details since they are trained to provide a rather generic "caption" for the image or video. They lack any specific mechanism to follow the activities/interactions of individual objects. For example, the image in Fig. 1(b) will likely be described as "several" cars on the street, and if the VLM is fine-tuned to recognize accidents, it will probably say that two (or even "some") cars are involved in an accident. This lack of specificity not only diminishes the value of the description but also makes fine-tuning difficult since an accurate description would need to point out which cars are involved in what type of activity. Segmentation and masking of the images have been used as potential ways to learn more details of what exists in the image [9], but that makes VLMs even more heavy-duty and less accurate.

This paper discusses a fine-tuning mechanism that utilizes Traditional Computer Vision (TCV) for object (and, if necessary, pose) detection, along with tracking and logical reasoning to allow the output of the fined-tuned VLM to be associated with specific objects. For object/pose detection, we use YOLOv8 [32] as it can work in real-time. We also track the objects to maintain their persistent IDs. Note that when a VLM is fine-tuned to recognize a set of N classes of activities, its output is generally limited to only those N classes. For each of these, we define a much simpler *proxy activity* that can be detected by TCV easily, ideally, based on basic parameters such as object type, size, location, separation, movement, etc. For example, the proxy activity for a rear-end accident is a car behind another car with a minimal distance between them. Similarly, a rather complex VLM-recognized activity of two people assembling some part in a factory may be characterized by the proxy activity of two people standing close together. (This assumes that the same proxy activity describes no other actual activity among the other $N-1$ classes; if not, we need to include some more detail.)

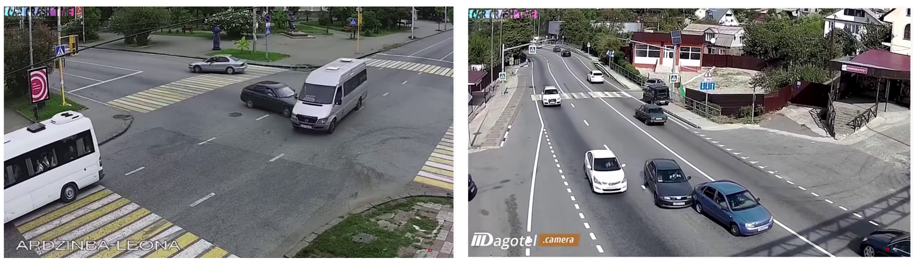
We take the unique approach of fine-tuning the same VLM for two related sets of *tasks*. For example, for the road traffic dataset introduced in section 4.2, we can identify task T_1 (performed by VLM1) as recording different types of accidents. Similarly, we can identify task T_2 (performed by VLM2) as recording the relative movements of vehicles. Each task T_i involves the classification across a set of N_i *activities* A_{ij}, $(j = 1, 2, ..., N_i)$, as depicted in Table 1. Activity A_{ij} in task T_i corresponds to a class that VLM_i is fine-tuned to recognize.

For each activity A_{ij}, we identify a distinct proxy activity A'_{ij} that can be easily recognized using TCV. We now have three distinct possibilities for detecting deficiencies in the VLM outputs and improving them via further focused fine-tuning. One is the consistency between the class identified by the VLM1 output and the class identified via TCV-recognized proxy activity for task T_1. Similarly, another possibility is the consistency between the class identified by the VLM2 output and the class identified by TCV-recognized proxy activity for task T_2. The third one is the compatibility between the classes identified by the two VLMs. The compatibility relationships are derived based on the knowledge of the two tasks; for example, for a rear-end accident to happen, the two vehicles must be moving in the same direction close together in the same lane. Such checks are helpful for fine-tuning and providing justifications at inference time, as discussed later.

2.2 Integrating Logic Reasoning with Computer Vision

Given the recognition of objects and their movements via TCV from video frames, we can define higher-level concepts as reusable functions using logic.

As a simple example, consider the definition of a function such as "following(V1, V2)" that asserts that vehicle V1 is following vehicle V2. The truth value of this assertion for any given pair of vehicles will be established (i.e., the function will be "grounded") by concluding from a sequence of frames of some minimum length that V1 is right behind V2. These definitions are needed only once and can be invoked in other parts of the logic "program" as needed. The reasoning generally also requires additional "theories" depending on the relevant physics either directly (e.g., Newton's Laws of motion) or in simplified form if needed. In addition, we surely need "theories" of basic arithmetic/comparison operations and any qualitative relationships we introduce, such as behind, ahead, etc.



(a) (b)

Fig. 1: TU_DAT dataset (a) car hit by another car from the side, and (b) shows a rear-end accident scenario.

We describe all such *Rules of Inference* (RoIs) and groundings in binary logic form for reasoning purposes. For example, if an object travels at speed s for time τ, the distance traveled d can be expressed as $d = s\tau$ being true. Such a representation allows the use of Boolean satisfiability modulo theory (SMT) based tools, the best known of them being Z3 [25] and YICES [12]. SMT tools can routinely solve significant practical problems despite the underlying NP-hardness and undecidability results, primarily because practical issues often have a lot of structure that can be exploited and further evidenced by their extensive use in many domains [1].

Unlike neuro-symbolic AI techniques [11, 19], explicit logic-based modeling does not require additional training. However, it does require putting together necessary assertions, which in this case concern consistency and compatibility between outputs of VLMs and TCV.

2.3 Related Work

Reasoning using VLM/LLM outputs has been extensively discussed in the literature [9, 26, 37, 38], although since VLMs/LLMs are simply large transformer models, the claims of reasoning ability can be questionable [15, 36]. The answers provided by a VLM/LLM entirely depend on the veracity of the extensive data used for pretraining and the limitations of the data used for fine-tuning. Because of this range and intentional randomness in LLM outputs (controlled by the temperature parameter), reasoning that directly uses the outputs of the VLM/LLM is different from the deductive reasoning proposed here, governed by explicitly specified Rules of Inference (RoIs). However, the RoIs can only establish consistency and sanity rather than ensure the correctness of VLM outputs. It is possible to explore the formulation of these RoIs based on the observed relationships similar to what inductive or analogical reasoning attempts to do; however, that is out of the scope of the current paper.

Ref [36] surveys "reasoning abilities" of multimodal LLMs. It discusses many Q&A datasets to test LLMs/VLMs in various domains and the genealogy of many LLMs/VLMs. LLMs generally provide a limited context window that maintains the previous Q&A in the conversation, which could help make better later predictions in the dialog. It is also possible to keep the knowledge externally and use it for later prompts [39]. Ref [9] introduces 3-stage LLM-based reasoning: see, think, and confirm. The see module uses a scene parser to detect all the candidate objects (concepts) in the image. Using an image captioner, the think module generates textual descriptions of relationships/concepts semantically related to the query. This description is given to LLM to answer the question. The confirm module requires the LLM to continue to generate the answer's supporting rationale (or justification) and verify them with a cross-modality classifier. The generated rationale is added back to the prompting context, begins a new think-confirm process, and iterates until the answer predictions in two consecutive iterations are consistent. A similar approach (observe, think, rethink) is described in ref [38]. The Chain of Thought (COT) [37] uses prompting to teach LLMs about formulating intermediate assertions to help them find the answer to a complex question. Ref [15] provides another survey of reasoning by LLMs (including multiple variants of COT) and argues why LLMs are still incapable of reasoning.

In the space of TCV and, more generally, deep learning, the issue of reasoning is often described as neuro-symbolic AI [11, 13, 19, 31]; however, it is mainly in the form of indirectly enforcing the constraints in neural net operations or loss function. For example, the popular Logic Tensor Networks (LTN) [7] enforces logic constraints implicitly and approximately by using differentiable extensions of Boolean operations [17]) to avoid the problem of exploding or vanishing gradients. Explicit logic reasoning approaches are relatively sparsely explored [5, 29]. Ref [27, 28] attempts to use explicit reasoning for accident and driver behavior characterization.

Explainable AI has seen a burgeoning amount of literature [4]. Although much of it concerns explaining the AI's decisions, the focus has now expanded to the more important problem of justifiability of those decisions [2, 14]. Our method supports justifiability in a simple way.

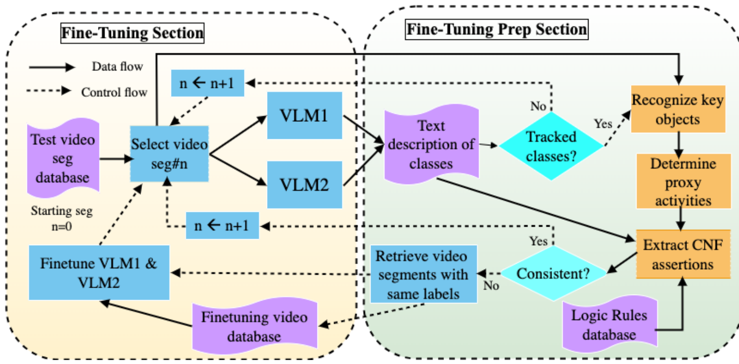


Fig. 2: Application Directed Fine-tuning of VLMs

3 Integrating Deep Learning with Logic Reasoning

The proposed fine-tuning approach is divided into two sections, (i) The Fine-Tuning Section and (ii) The Fine-Tuning Prep Section, as illustrated in Fig. 2. We explain it specifically for the traffic scenario, but the approach generally applies and can easily use multiple VLMs rather than the two shown. The fine-tuned version of each VLM would classify the input image/video into one of the defined number of classes. The description of each class concerns some detail of the type of events that VLM is intended to recognize. For example, suppose the events of interest relate to collision accidents. In that case, the individual classes may correspond to descriptions like rear-end accident between car and truck, head-on collision between vehicles, motorcycle hit from the side by a car, etc. Similarly, a VLM that concerns vehicular movements may use class descriptions such as a vehicle following another in the same lane, a car driving next to another, etc.

Fine-Tuning Section: Initially, both VLMs are fine-tuned using a set of randomly selected labeled inputs that could be images or videos, depending on the type of VLM used. We chop longer videos into concise ones so that each video focuses on only one class of interactions as far as possible. We label (or caption) these video segments according to the requirements of the specific VLM used. For image-based VLMs, we label each image in the video segment identically. We divide the entire video set into test and fine-tuning sets randomly. Each test video (numbered as $n = 0, 1, 2, ...$) is passed through both VLMs, which provide probabilities for each defined class. If the results do not assign a significant probability to any of the tracked events, we move on to the next video but retain this video in the set as it may be helpful later when the VLMs are better tuned.

Fine-Tuning Prep Section: This section mainly focuses on TCV, where we run an object detector (e.g., YOLOv8) on the video to identify and track significant objects (e.g., cars, motorbikes, pedestrians, etc.). This allows us to associate object IDs with VLM output (through the proxy activity mechanism discussed in section 2.1). These can be encoded using logic as assertions for direct reasoning. The RoIs further assist these assertions to enable consistency checking, shown as a “logic rules database.”

The goal now is to check if the activities detected (represented by class labels) by the two VLMs are mutually consistent. The RoIs for mutual consistency must be pre-established and become part of the logic rules database. For example, the two descriptions are consistent if VLM1 indicates a rear-end accident and VLM2 indicates movement in the same lane and direction. The two are inconsistent if VLM2 indicates movements in different lanes or opposite directions. The rigor with which the consistency can be checked depends on how detailed information we get from VLMs and our ability to associate correct object IDs with them. We could thus formulate Conjunctive Normal Form (CNF) assertions based on the VLM output and check the consistency according to the specified RoIs. In case of inconsistency, we identify a new video from the fine-tuning set with the same labels and use that for fine-tuning both VLMs. Usually, we would want to evaluate the inconsistency and subsequent directed fine-tuning in batches, with batch size being a hyperparameter of the algorithm. The fine-tuning process can be repeated until suitable stopping criteria are reached.

4 Experimental Evaluation

4.1 VLMs Used For Evaluation

For evaluation, we used one VLM with images (Minigpt-4) and two with videos (X-Clip and VideoMAE). MiniGPT-4 uses BLIP-2 (Bootstrapping Language-Image Pre-training) [20], which defines two trainable layers to align a frozen vision transformer model with a frozen LLM model. MiniGPT-4 uses the pre-trained vision component of BLIP-2 and adds a single projection layer to align the encoded visual features with frozen Vicuna LLM. Minigpt-4 is quite popular in academic environments because it can be fine-tuned on modest GPU machines.

X-CLIP [23] is designed for video-text retrieval and generates multi-grained visual and textual representations. It then uses multi-grained contrast of features (i.e., video-sentence, video-word, sentence-frame, and frame-word) to obtain multi-grained similarity scores, vectors, and matrices. It dynamically considers the importance of each frame in the video and each word in the sentence so that the impact of unimportant words and unnecessary frames on retrieval performance is reduced.

VideoMAE [33, 35] uses a self-supervised training mechanism with videos. It randomly selects a sequence of frames in a time window. These are divided into a 16x16 grid in the image plane. This provides so-called “tubes”, or grid elements extended in the time dimension. The grid elements are embedded in the token space using the self-attention mechanism in space and time. The tubes are heavily masked, and the token representation is used to train an autoencoder, hence the name Video Masked AutoEncoder (VideoMAE).

4.2 Data Sets Used

Description of Datasets: The first dataset we used, called TU_DAT [18], concerns road traffic and contains diverse accident types, weather conditions, and videos collected in challenging environments. Fig. 1 (a) shows a car hit by another car from the side, and (b) shows a rear-end accident scenario.

Table 1: Description of Classes used for TU_DAT Dataset

Class#	Classes in VLM1	Classes in VLM2
Class1	Car hit by another car from behind	Cars moving in same direction
Class2	Car hit by another car from side	Cars moving in opposite direction
Class3	Car hit by another car from front	Cars moving next to one another
Class4	Car hits a static object	Cars moving behind one another
Class5	Motorcycle hits a pedestrian	Cars moving perpendicular to each other
Class6	Traffic videos	Car & motorcycle moving one behind another
Class7	Not defined	Car & motorcycle moving next to one another
Class8	Not defined	Pedestrians walking

Table 2: Description of Classes used for Taekwondo Dataset

Class#	Classes in VLM1	Classes in VLM2
Class1	Left leg still, right leg stands still	Left arms out, right arms out
Class2	Left leg still, right leg moving forward	Left arms out, right arms folded
Class3	Left leg still, Right leg moving backward	Left arms folded, right arms out
Class4	Right leg still, Left leg moving forward	Left arms folded, right arms folded
Class5	Right leg still, Left leg moving backward	Left arms on the head, right arms folded
Class6	Left leg moves forward, Right leg backward	Right arms on the head, left arms folded
Class7	Right leg moves forward, Left leg backward	Not defined

Our second dataset is the Taekwondo dataset, explicitly developed with data on movements performed by Taekwondo athletes. We collect videos of students at Darimar Martial Arts, Columbus, Ohio. The acquired dataset comprises various Taekwondo patterns, each symbolizing a distinct movement executed by an athlete for a specific belt. The patterns include the following belt colors: white, yellow, orange, green, and black. Understanding the movement patterns is a crucial component of Taekwondo training, as explained in the Taekwondo America student manual [3]. We have a collection of 35 videos in total, which feature either a single student or multiple students performing the movements in sequence for each belt pattern. Fig. 3 (a) shows the walking stance low block, and (b) shows the walking stance reverse punch of a student in a dark green belt pattern.

(a) (b)

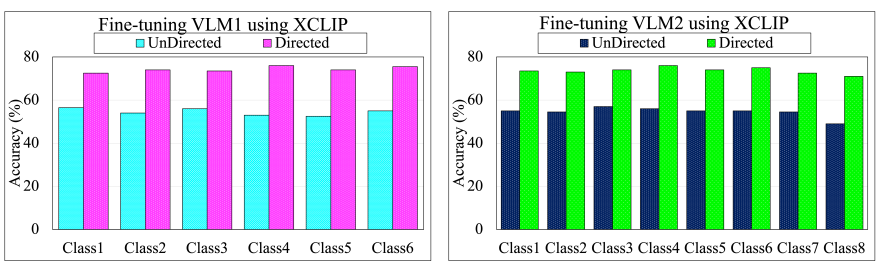
Fig. 3: Green belt movement patterns in Taekwondo

Description of Classes Used to Fine-Tune VLMs: The TU_DAT dataset contains several accident scenarios in road traffic, forming the classes for fine-tuning any VLM. Since our proposed method includes fine-tuning two VLMs, the videos in the TU_DAT dataset have been categorized into modeling accident scenarios for VLM1 and recognizing the relative position/movements of vehicles for VLM2. The description of classes used in fine-tuning VLM1 and VLM2 on TU_DAT are shown in Table 1. Although the table shows courses with the same number side by side for VLM1 and VLM2, the numbering does not reflect any relationship between them. Instead, any relationship will be captured via the logic assertions used for consistency.

For the taekwondo dataset, VLM1 is fine-tuned to recognize the leg movements of the students, while VLM2 is fine-tuned to identify the students' arm movements. The description of classes used in fine-tuning VLM1 and VLM2 on the Taekwondo dataset are shown in Table 2. Again, the same class number for VLM1 and VLM2 is not intended to reflect any relationship between them.

4.3 Experimental Results

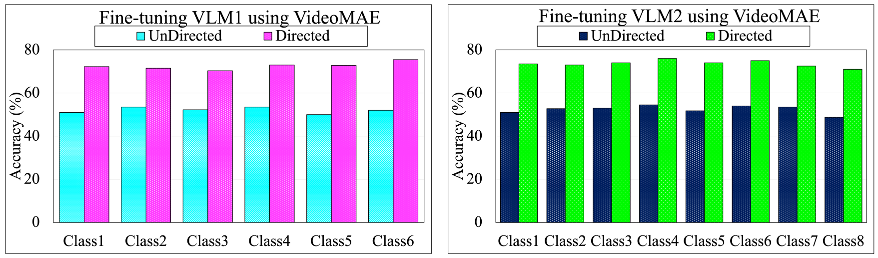
This section evaluates our proposed application-directed fine-tuning framework on the collected datasets discussed in Section 4.2. The following metrics determine the effectiveness of our framework: (a) Fine-tuning (FT) time of both VLMs, (b) Accuracy of inference, (c) Inference time, and (d) Justifiability time. The experiments were performed on a server with two NVIDIA RTX A6000 GPUs, each equipped with 10752 CUDA cores and 48GB GDRR6 memory.



(a) (b)

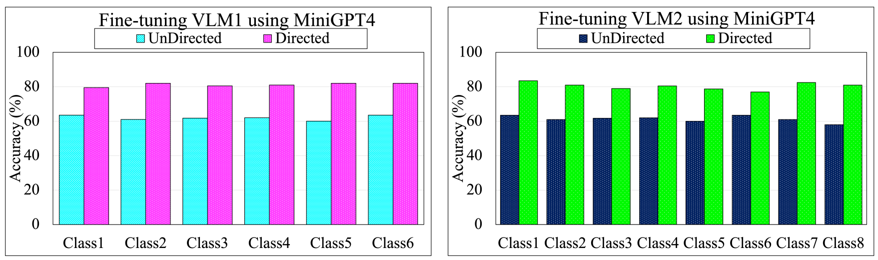
Fig. 4: FT Time for XCLIP (TU_DAT Dataset) (a) VLM1, (b)VLM2.

Calculating the accuracy involves assessing inconsistencies between the outputs of the two VLMs and the logical reasoning tool. We start with a base-line fine-tuning of both VLMs, which consists of the following steps: (1) select few videos from each class in the training set, then (2) fine-tune both VLM1 and VLM2 on this subset of videos; (3) run the test videos in batches on the fine-tuned VLMs; and (4) record the inconsistency between the outputs of both VLMs and the reasoning tool. Next, we do further fine-tuning using both undirected and directed methods. Our directed fine-tuning intelligently selects 20 videos from the classes that exhibit inconsistencies between the outputs of both VLMs and the reasoning tool. The choice of 20 videos is somewhat arbitrary and can



(a) (b)

Fig. 5: FT Time for Video-MAE (TU_DAT Dataset) (a) VLM1, (b)VLM2.

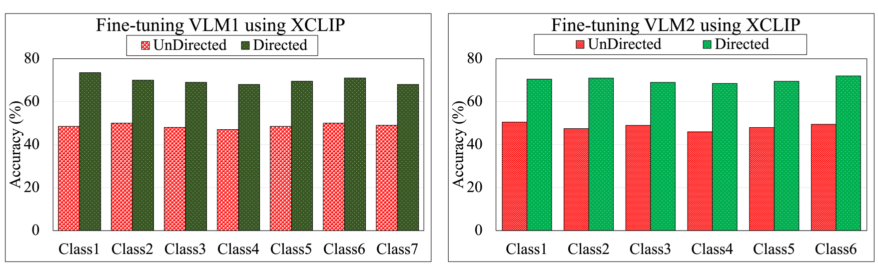


(a) (b)

Fig. 6: FT Time for MiniGPT4 (TU_DAT Dataset) (a) VLM1, (b)VLM2.

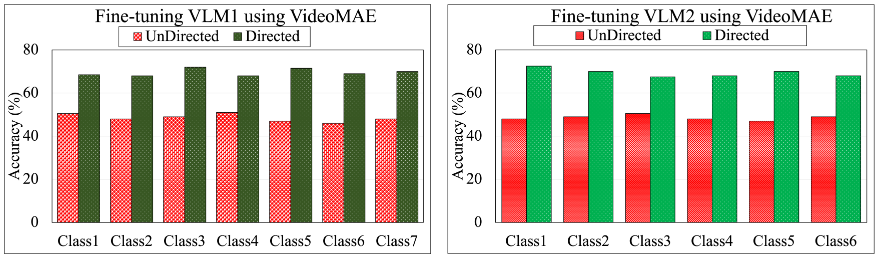
be chosen adaptively based on the misclassifications, although this aspect has not been investigated here. To make a fair comparison, we execute the loop four times for both directed and undirected cases, each using 20 videos. We stopped at four iterations since the improvement in accuracy appeared to be saturated after that.

Fine-Tuning Accuracy: Fig. 4, 5 and 6 show the results of fine-tuning both VLM1 and VLM2 on the TU_DAT dataset using XCLIP, VideoMAE and MiniGPT4 respectively. Similarly, Fig. 7, 8 and 9 shows the results of fine-tuning



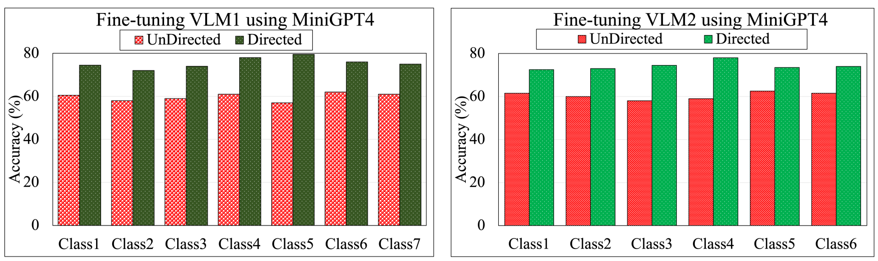
(a) (b)

Fig. 7: FT Time for XCLIP (Taekwondo Dataset) (a) VLM1, (b)VLM2.



(a) (b)

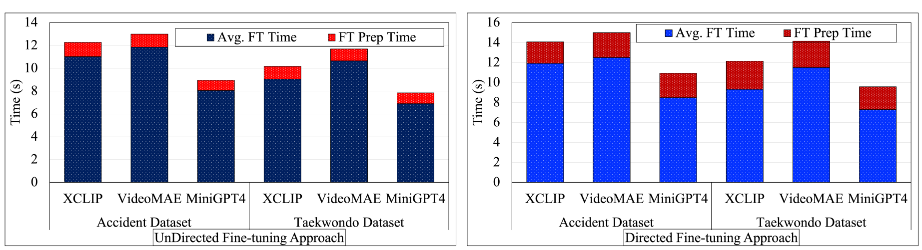
Fig. 8: FT Time for Video-MAE (Taekwondo Dataset) (a) VLM1, (b)VLM2.



(a) (b)

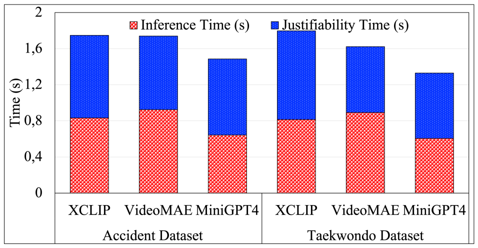
Fig. 9: FT Time for MiniGPT4 (Taekwondo Dataset) (a) VLM1, (b)VLM2.

both VLM1 and VLM2 on the Taekwondo dataset using XCLIP, VideoMAE and MiniGPT4 respectively. In all these figures, the x-axis indicates the acronyms of classes utilized by VLM1 and VLM2, with their descriptions found in Section 4, while the y-axis represents the accuracy. It is clear that our directed fine-tuning surpasses undirected fine-tuning methods in all cases by a very significant margin of roughly 20 percentage points. Note that the substantial improvement persists for two very different types of videos (road traffic vs. taekwondo), confirming that the improvement is not tied to the video characteristics.

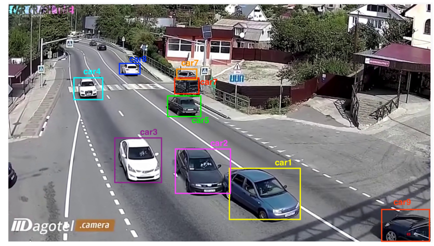


(a) (b)

Fig. 10: Overall per-epoch prep and FT Time for (a) Undirected Fine-tuning and (b) Directed Fine-tuning



(a) Inference & justification time



(b) Finding key objects in the scene

Fig. 11

VLM1 Output: Matching Caption: Car hit by another car from behind
VLM2 Output: Matching Caption: Cars moving next to one another

Consistency Check: NO - VLM1 output is inconsistent with VLM2 output

Retrieve Videos with labels: Car hit by another car from behind, Cars moving next to one another, Cars moving behind one another

Fig. 12: Directed Fine-tuning prep with XCLIP on TU_DAT Dataset

A Detailed Fine-Tuning Example: In this section, we present an example of a rear-end accident scenario as shown in Fig. 1 (b) from the TU_DAT dataset, demonstrating the functionality of our directed fine-tuning approach.

In this example, we apply directed fine-tuning on XCLIP. After the initial fine-tuning stage, VLM1 and VLM2 yield the captions as shown in the top two lines in Fig. 12. The next step is to identify the key objects in the scene using YOLOv8, as shown in Fig. 11 (b). On the basis of the identified key objects, we determine the proxy activities associated with various objects in the scene, as illustrated in Table 3. The assertions derived from the object relationships established in the previous step are shown in Table 4. These assertions substantiate the possibility of a scenario of a car being hit by another car from behind, whereas the VLM2 output indicates that vehicles are moving adjacently. As a result, the output of VLMs and the logical reasoning tool are inconsistent; therefore, we select and retrieve the videos with the labels as depicted in Fig. 12 for additional fine-tuning of both VLMs.

car1 and car2 moving one behind another
car1, car2, car3 & car4 are traveling in same lane
car1 and car2 are following very close to each other from behind
car1 and car9 are traveling in the opposite lane
car7 is parked and not moving
car5, car8, & car 9 are traveling in same lane

Table 3: Determine proxy activities

Variables: car1, car2.., car9 are unbound integers
Functions: Boolean, each with one Integer argument move_behind(), move_very_close(), move_opp_dirn() move_same_dirn() car_hit_from_behind()
Groundings:
move_behind(car1) $\wedge$ move_behind(car2)
move_very_close(car1) $\wedge$ move_very_close(car2)
move_same_dirn(car1) $\wedge$ move_same_dirn(car2)
move_same_dirn(car5) $\wedge$ move_same_dirn(car8) $\wedge$ move_same-_dirn(car9)
move_opp_dirn(car1) $\wedge$ move_opp_dirn(car9)
move_behind(car1) $\wedge$ move_behind(car2) $\wedge$ move_very_close(car1) $\wedge$ move_very_close(car2) $\implies$ car_hit_from_behind (car1)

Table 4: Assertions for Reasoning

Fine-Tuning Time: We also compare the time required for both directed and undirected fine-tuning approaches for all three VLMs under consideration. As stated earlier, we use four iterations of fine-tuning for both directed and undirected cases, each time using 20 videos. For each iteration, the time spent consists of two parts, shown by the dotted boxes in Fig. 2.

1. Actual fine-tuning time, reported as average per epoch (over 500 epochs).
2. Preparation time, reported as average per epoch. For the undirected FT, it is the time to retrieve 20 videos from the disk (randomly in this case). For directed FT, it *also* includes the overhead of running YOLO, querying VLM1 and VLM2, generating assertions, and using them for consistency checking.

Figs. 10 (a) and (b) show the average per-epoch fine-tuning and fine-tuning prep time, respectively, for both undirected and directed cases. As expected, the fine-tuning time is almost identical in both cases and is in the ∼10-12 sec range. The prep time is much shorter; a significant piece is the time to retrieve and load videos from the disk. The time taken by other pieces of directed fine-tuning is relatively modest.

Inference and Justification Time: The fine-tuning prep section in Fig. 2 can also be viewed as a mechanism to augment inferencing with justifiability, which is crucial with Blackbox AI models. For this, we take out the fine-tuning section during inferencing (thus breaking the loop) but retain other parts. In this case, each inference will also be accompanied with the following information:

1. Output justified by alluding to the consistency between VLMs, and across VLMs and TCV-based proxy activity detection.
2. Output marked as faulty along with a reason why it is considered questionable.

Fig. 11 displays the inference and justifiability time for all 3 VLMs on both datasets. Note that the justifiability time differs from the fine-tuning prep time reported above since we no longer have the significant overhead of retrieving videos from disk for fine-tuning. It is seen that the justifiability time is about the same as the inference time. This should be reasonable for critical applications; for others, we may exclude the justifiability at inferencing or run it only occasionally as a sanity check.

Catastrophic Forgetting: Catastrophic forgetting (CF) is a phenomenon where a model loses previously acquired knowledge while learning new information [22]. To understand this phenomenon in the context of our fine-tuning, we conducted a small study as follows: We first evaluated the ability of the original XCLIP (say, version 0) to identify VLM1 classes, specifically accidents. It achieved an average accuracy of approximately 36% in recognizing accident classes. We then fine-tuned XCLIP on accident videos, thereby creating, say, version 1. The accuracy of accident recognition for version 1 increased to 44%.

Next, we fine-tuned version 1 to recognize the relative movements of VLM2 vehicle classes to create version 2. We then evaluated version 2's ability to identify accident classes, resulting in a reduced accuracy of 32%, lower than version 0. *This demonstrates the CF phenomenon and justifies the creation of two distinct fine-tuned versions (VLM1 and VLM2) for the two tasks instead of using just one and evaluating its consistency with the objects identified by the TCV part.*

5 Conclusions and Discussion

In this paper, we propose a novel fine-tuning mechanism for VLMs. This mechanism combines traditional computer vision (TCV) to recognize details with explicit logical reasoning to improve the performance of emerging vision LLMs (VLMs). The mechanism substantially reduces the effort and resource needs of fine-tuning while providing considerably higher accuracy and a justification mechanism that can continue to be used at inference time.

In particular, we demonstrated that identifying the objects and proxy activities in the video stream can formulate a simple yet powerful way of detecting the areas where the fine-tuned VLM is deficient. This allows us to conduct informed fine-tuning that can be used with both image and video-based VLMs. We demonstrated that the proposed mechanism increases the accuracy by about 20 percentage points in all cases compared to the one achievable via undirected fine-tuning.

The proposed mechanism is quite general, as it can be applied to any VLM and dataset. It can also be extended in multiple directions:

1. The proxy activities could be more complex to ensure separation between different classes recognized by the fine-tuned VLMs and to enrich opportunities for accuracy/consistency checking.
2. The mechanism can be generalized to more than two VLMs to capture many activities and events.
3. Since the TCV algorithms can make mistakes, we improve robustness by exploiting conditions like the smoothness of change across video frames (e.g., a car identified as a truck in some frames or its movements not conforming to the feasible rate of change). Such enhancements also support better justifications at inference time at the cost of higher processing time.

It may be noted that detecting more complex proxy activities may require us to go beyond SMT-based reasoning and bring in issues of temporal ordering, real-time, and ongoing processes in the reasoning itself. This can be done through temporal extensions [16], real-time extensions [8], and process extensions [6, 10, 34]. Such extensions have been used in ref [27, 28] for recognizing more complex activities. Other potential extensions include detecting and correcting mistakes in the TCV by exploiting continuity and smoothness constraints in what can happen over successive frames. More generally, modifying VLM output through NLP techniques to inject the identified object IDs is possible.

Dataset Collection: We certify that our taekwondo dataset was collected with proper permissions.

References

1. Ábrahám, E., Kremer, G.: Satisfiability checking: Theory and applications. In: Software Engineering and Formal Methods: 14th International Conference, SEFM 2016, Held as Part of STAF 2016, Vienna, Austria, July 4-8, 2016, Proceedings 14. pp. 9–23. Springer (2016)
2. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE access **6**, 52138–52160 (2018)
3. America, T.: Taekwondo student manual. `https://taekwondoamerica.org/wp-content/uploads/2017/09/Student-Manual-2012.pdf` (2011)
4. Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion **58**, 82–115 (2020)
5. Artikis, A., et al.: A logic programming approach to activity recognition. In: Proceedings of the 2nd ACM international workshop on Events in multimedia. pp. 3–8 (2010)
6. Artikis, A., et al.: An event calculus for event recognition. IEEE TKDE **27**(4), 895–908 (2014)
7. Badreddine, S., Garcez, A.d., Serafini, L., Spranger, M.: Logic tensor networks (ltn). Artificial Intelligence **303**, 103649 (2022)
8. Cavada, R., Cimatti, A., Dorigatti, M., Griggio, A., Mariotti, A., Micheli, A., Mover, S., Roveri, M., Tonetta, S.: The nuxmv symbolic model checker. In: CAV. pp. 334–342 (2014)
9. Chen, Z., Zhou, Q., Shen, Y., Hong, Y., Zhang, H., Gan, C.: See, think, confirm: Interactive prompting between vision and language models for knowledge-based visual reasoning. arXiv preprint arXiv:2301.05226 (2023)
10. Chisalita, I., et al.: Traffic accidents modeling and analysis using temporal reasoning. In: ITSC. pp. 378–383 (2004). `https://doi.org/10.1109/ITSC.2004.1398928`
11. De Raedt, L., Dumančić, S., Manhaeve, R., Marra, G.: From statistical relational to neuro-symbolic artificial intelligence. arXiv preprint arXiv:2003.08316 (2020)
12. Dutertre, B.: Yices 2.2. Computer-Aided Verification (CAV'2014) **8559**, 737–744 (July 2014)
13. Garcez, A.d., Lamb, L.C.: Neurosymbolic ai: the 3rd wave. arXiv preprint arXiv:2012.05876 (2020)
14. Holzinger, A., Goebel, R., Fong, R., Moon, T., Müller, K.R., Samek, W.: xxai-beyond explainable artificial intelligence. In: International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers. pp. 3–10. Springer (2020)
15. Huang, J., Chang, K.C.C.: Towards reasoning in large language models: A survey. arXiv preprint arXiv:2212.10403 (2022)
16. Konur, S.: A survey on temporal logics for specifying and verifying real-time systems. Frontiers of Computer Science **7**(3), 370–403 (2013)
17. van Krieken, E., Acar, E., van Harmelen, F.: Analyzing differentiable fuzzy logic operators. Artificial Intelligence **302**, 103602 (2022)
18. Kumar., P.P.: Tu-dat dataset. `https://github.com/pavana27/TU-DAT` (2021)
19. Lee, J.H., Sioutis, M., Ahrens, K., Alirezaie, M., Kerzel, M., Wermter, S.: Neuro-symbolic spatio-temporal reasoning. arXiv preprint arXiv:2211.15566 (2022)
20. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597 (2023)

21. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. arXiv preprint arXiv:2304.08485 (2023)
22. Luo, Y., Yang, Z., Meng, F., Li, Y., Zhou, J., Zhang, Y.: An empirical study of catastrophic forgetting in large language models during continual fine-tuning. arXiv preprint arXiv:2308.08747 (2023)
23. Ma, Y., Xu, G., Sun, X., Yan, M., Zhang, J., Ji, R.: X-clip: End-to-end multi-grained contrastive learning for video-text retrieval. In: Proceedings of the 30th ACM International Conference on Multimedia. pp. 638–647 (2022)
24. Maaz, M., Rasheed, H., Khan, S., Khan, F.S.: Video-chatgpt: Towards detailed video understanding via large vision and language models. arXiv preprint arXiv:2306.05424 (2023)
25. de Moura, L., Bjørner, N.: Z3: An efficient smt solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS. pp. 337–340 (2008)
26. Paranjape, B., Lundberg, S., Singh, S., Hajishirzi, H., Zettlemoyer, L., Ribeiro, M.T.: Art: Automatic multi-step reasoning and tool-use for large language models. arXiv preprint arXiv:2303.09014 (2023)
27. Pradeep, P., Kant, K., Pal, A.: C-far: A compositional framework for anomaly resolution in intelligent transportation system. IEEE Trans. on Intelligent Transportation Systems (Aug 2022). https://doi.org/10.1109/TITS.2022.3196548
28. Pradeep, P., Kant, K., Pal, A.: Non-intrusive driver behavior characterization from road-side cameras. IEEE IoT Journal (Sept 2023). https://doi.org/10.1109/JIOT.2023.3285886
29. Ramirez-Amaro, K., Kim, E.S., Kim, J., Zhang, B.T., Beetz, M., Cheng, G.: Enhancing human action recognition through spatio-temporal feature learning and semantic rules. In: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids). pp. 456–461. IEEE (2013)
30. Rasheed, H., Khattak, M.U., Maaz, M., Khan, S., Khan, F.S.: Fine-tuned clip models are efficient video learners. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6545–6554 (2023)
31. Sarker, M.K., Zhou, L., Eberhart, A., Hitzler, P.: Neuro-symbolic artificial intelligence. AI Communications **34**(3), 197–209 (2021)
32. Terven, J., Córdova-Esparza, D.M., Romero-González, J.A.: A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. Machine Learning and Knowledge Extraction **5**(4), 1680–1716 (2023)
33. Tong, Z., Song, Y., Wang, J., Wang, L.: Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. Advances in neural information processing systems **35**, 10078–10093 (2022)
34. Vlassopoulos, C., Artikis, A.: Towards a simple event calculus for run-time reasoning (rtec). In: COMMONSENSE (2017)
35. Wang, L., Huang, B., Zhao, Z., Tong, Z., He, Y., Wang, Y., Wang, Y., Qiao, Y.: Videomae v2: Scaling video masked autoencoders with dual masking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14549–14560 (2023)
36. Wang, Y., Chen, W., Han, X., Lin, X., Zhao, H., Liu, Y., Zhai, B., Yuan, J., You, Q., Yang, H.: Exploring the reasoning abilities of multimodal large language models (mllms): A comprehensive survey on emerging trends in multimodal reasoning. arXiv preprint arXiv:2401.06805 (2024)
37. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems **35**, 24824–24837 (2022)

38. Yang, Y., Zhang, X., Han, W.: Enhance reasoning ability of visual-language models via large language models. arXiv preprint arXiv:2305.13267 (2023)
39. Yoneda, T., Fang, J., Li, P., Zhang, H., Jiang, T., Lin, S., Picker, B., Yunis, D., Mei, H., Walter, M.R.: Statler: State-maintaining language models for embodied reasoning. arXiv preprint arXiv:2306.17840 (2023)
40. Zhang, H., Li, X., Bing, L.: Video-llama: An instruction-tuned audio-visual language model for video understanding. arXiv preprint arXiv:2306.02858 (2023)
41. Zhu, D., Chen, J., Shen, X., Li, X., Elhoseiny, M.: Minigpt-4: Enhancing vision-language understanding with advanced large language models. arXiv preprint arXiv:2304.10592 (2023)