Self Evaluation Using Zero-shot Learning

2023 5th International Conference on Robotics and Computer Vision (ICRCV) | 979-8-3503-2636-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICRCV59470.2023.10329149

Anuva Banwasi*
Department of Computer Science
Columbia University
New York, USA
ab5084@columbia.edu
*Corresponding author

Xinghua Sun
Department of Computer Science
Columbia University
New York, USA
xs2445@columbia.edu

Rohith Ravindranath
Department of Computer Science
Stanford University
Palo Alto, USA
rohithr@stanford.edu

Marc Vazquez
Department of Computer Science
Columbia University
New York, USA
mav2179@columbia.edu

Abstract— **With recent advances in large language models (LLM) and visual language models (VLM), there has been great interest in extending these models to robotics and task execution. Recent studies have bridged the gap between large language models and robotic language processing by providing a real-world grounding to help the robot better understand and execute tasks. A major drawback of existing approaches is that they still require a human to monitor and evaluate the robot's performance. As robots expand into a wide range of industries from agriculture to healthcare, it is important that they can self-monitor their performance and identify when any mistake occurs. We propose a model that combines large-language models and visual language models to self-evaluate task execution based on video data. We utilize LLMs to generate outcome descriptions for each step and then provide these as input prompts to a VLM to determine whether a step has been completed. Based on video data, our solution identifies if each step in a task has been executed correctly and notices when a mistake has been made. We design a score metric that assesses the status of completion of each step in the task. Upon testing the model on a variety of household-related tasks, it demonstrated successful evaluation of task completion. Our model could be implemented in robots to self-evaluate task execution, leading to more accurate and efficient robotic systems.**

Keywords— Large language models, visual language models, language grounding for vision and video understanding, robotics

I. Introduction

The field of robotics has undergone rapid advancements due to developments in computing as well as learning techniques. In recent years, the emergence of large language models trained on massive web corpus has revolutionized how we interpret natural language instructions and potentially execute tasks on robots. The paper 'Do as I can, not As I say' [1] bridges the gap between large language models and robotic language processing by providing a real-world grounding to help the robot better understand and execute tasks. This method for task execution requires human evaluators to monitor the robot and ensure it is operating, especially to gauge success as it moves through a long-horizon task.

The goal of this project is to create a system that utilizes a self-evaluation policy to determine whether a task has been completed, and self-identify errors made during task execution without a human evaluator. Our proposed solution utilizes a combination of visual language models (VLM) and large-language models (LLM) to evaluate action completion during each step of a task. This will allow the robot to self-evaluate whether it is correctly performing a task as well as notice any errors it makes as it completes each step of a task sequence. Code is available at: SelfEval GitHub.

II. Related Work

In recent years, there have been many breakthroughs in AI and natural language processing, particularly with respect to transformers and large language models. The Attention is All You Need paper [2] demonstrated that a transformer composed of an encoder and decoder with an attention mechanism can successfully perform tasks such as machine translation. Since then, there have been several large language models that utilize transformer networks to learn context and perform word prediction. Most recently, GPT-4 as described in their technical report [3], showed impressive performance on exams designed for humans. Similar to large language models, there have been several advances in vision. Visual language models such as CLIP [4] and Flamingo [5] can perform tasks such as question-answering and captioning of image and video data. Despite these numerous advancements in the domains of vision and language, there remains a necessity to apply these developments in areas such as robotics. Language grounding for robotic task execution is a growing and exciting field. 'Do as I Can, not as I say' [1] shows that large language models can provide real-world grounding for robotic task execution. However, one main limitation of this paper and in most robots today is that they still require human evaluators to oversee task execution and identify when mistakes occur. In this paper, we propose a process that combines large language models and visual language models to self-evaluate task execution. This novel application of LLM and VLM for self-execution can help lead the way towards robots that monitor their own performance and ultimately self-improve.

III. Methodology

Our overall approach was to create a system that takes a video or sequence of images and goes through each step in a task sequence, assessing whether tasks have been completed. A diagram of the system methodology is shown in Figure 1. Videos of long-horizon tasks were collected for common kitchen tasks such as washing a bowl, cutting an apple and pouring a glass of juice. After preprocessing videos into images using OpenCV, the images are sent to the evaluator.

For each step in the sequence, the LLM generates prompts describing the outcome of each step action. The output prompts are then fed from the LLM to the VLM to see how well the image aligns with the generated outcomes for the step. Based on this, it determines whether the step was completed successfully. These models are trained on large-scale internet-based language corpus, so they are generalizable and equipped to parse natural language instructions [6].

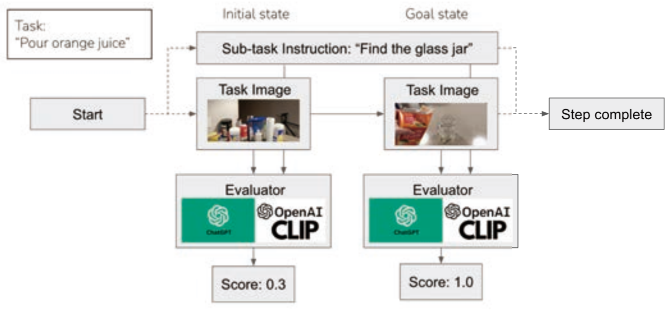
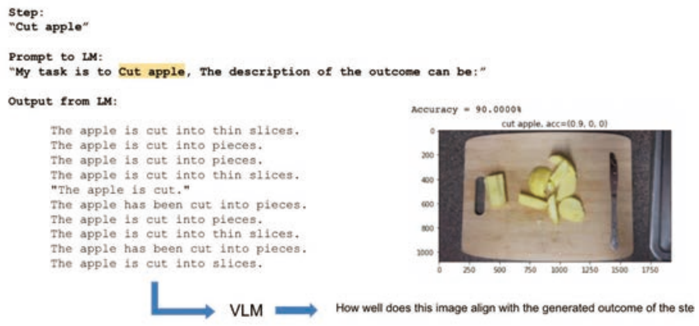


Fig. 1. A long-horizon task is divided into sub-tasks or steps that make up a task sequence. At each step, the evaluator is prompted, and a score is given based on the likelihood of successful completion.



```
Step:
"Cut apple"

Prompt to LM:
"My task is to Cut apple, The description of the outcome can be:"

Output from LM:

      The apple is cut into thin slices.
      The apple is cut into pieces.
      The apple is cut into pieces.
      The apple is cut into thin slices.
      "The apple is cut."
      The apple has been cut into pieces.
      The apple is cut into pieces.
      The apple is cut into thin slices.
      The apple has been cut into pieces.
      The apple is cut into slices.
```

Accuracy = 90.0000%

cut apple, acc=(0.9, 0, 0)

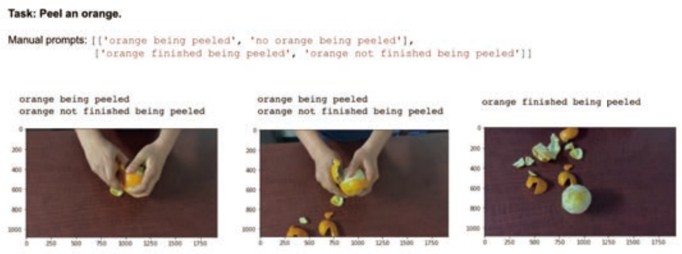
VLM → How well does this image align with the generated outcome of the ste

Fig. 2. The task is fed to the LLM in the form of an engineered prompt, and the ten most likely outputs are listed. These are fed to the VLM, and an accuracy score is generated based on how many outputs are correct.

IV. Results

A. Preliminary Results

This section encompasses various results gathered as we built a system to validate our ideas. Data was collected in the form of videos performing tasks within our home environments. In preliminary experiments, we took only a single image depiction for each task within a task sequence. We began experimenting with the VLM by manually entering prompts for each image in a sequence. Figure 3 shows an example task and how subsequent steps were handled using hard-coded prompting. This process validated our approach of using VLM to assess task completion. However, it is impractical to implement manual prompting as it is difficult to generalize across tasks. This is why we moved towards a new technique using LLMs.



Task: Peel an orange.

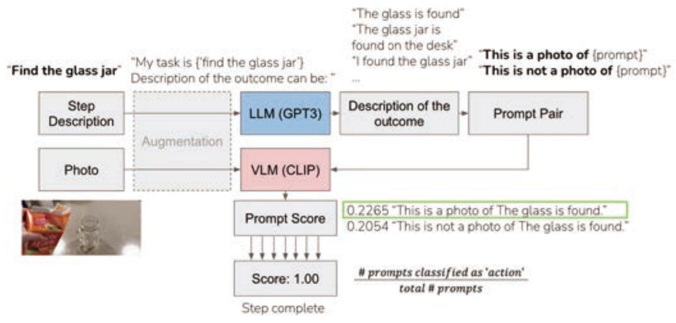
Manual prompts: [['orange being peeled', 'no orange being peeled'],
['orange finished being peeled', 'orange not finished being peeled']]

orange being peeled
orange not finished being peeled
orange being peeled
orange not finished being peeled
orange finished being peeled

Fig. 3. Our experiments started with manual prompts, hard-coded based on the task at hand. It can be observed that the VLM is correctly able to classify whether the orange is being peeled or if it has finished peeling.

B. Automatic prompting model using LLM and VLM

Figure 4 is a diagram of the complete automatic prompting pipeline, which works as follows. For each frame of a task sequence, a step description is assigned. A basic augmentation of the step description is performed, framing it appropriately for the language models to interpret. The step description is parsed by the LLM which gathers a set of descriptions of the outcome. A pairing is generated based on the positive statement and a generated negative statement. This information along with the photo of the task is sent to the VLM and based on the weights assigned, a prompt score is generated, this score functions as a measure of step completeness. This score metric is calculated as the number of prompts that correctly classified the action as completed over the total prompts. For instance, if the step is "find a glass" and the score is 0.80, this means that 8 out of 10 prompt pairs were classified as "the glass is found."



"The glass is found"
"The glass jar is found on the desk"
"I found the glass jar"
...
"Find the glass jar"
"My task is {'find the glass jar'} Description of the outcome can be: "
"This is a photo of {prompt}"
"This is not a photo of {prompt}"
Step Description
Augmentation
LLM (GPT3)
Description of the outcome
Prompt Pair
Photo
VLM (CLIP)
Prompt Score
0.2265 "This is a photo of The glass is found."
0.2054 "This is not a photo of The glass is found."
Score: 1.00
# prompts classified as 'action' / total # prompts
Step complete

Fig. 4. Automatic prompting pipeline showing a progression of the 'pour orange juice' example.

C. Experiments

We tested our method on a variety of household tasks such as pour orange juice, cut apple, and wash bowl. We also included failure cases where a step is executed incorrectly. Figures 5 through 8 showcase experiments validating our ideas.

We show the automatic prompting pipeline showing a progression of the 'pour orange juice' example. Augmentation is performed to suit the prompt for the language model. The model correctly identifies when a step has been completed such as when the juice carton comes into sight or when glass is full.

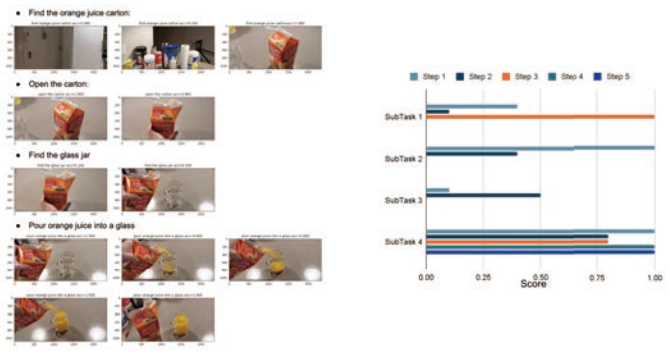


Fig. 5. Score progression for the 'pour orange juice' task. For each step or subtask in the process, the score increases as the step is executed. For example, for step 1, score increase from 0.4 to 1.0 when the carton comes into view.

The below example shows the task sequence and corresponding score for each step over time. The score increases as the apple is progressively cut, validating our score metric as a measure of step completion.

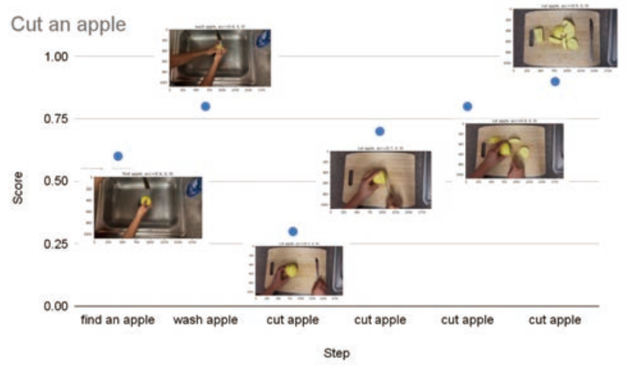


Fig. 6. As the apple is progressively cut, the score for completion increases until it reaches 0.9 at the final state.

Task sequence of cleaning a bowl, with each image corresponding to a task in the sequence. The model correctly determines each step as it is completed such as clean dirty bowl, add soap to sponge, clean bowl, etc. This shows the system can differentiate between states and correctly judge task execution.

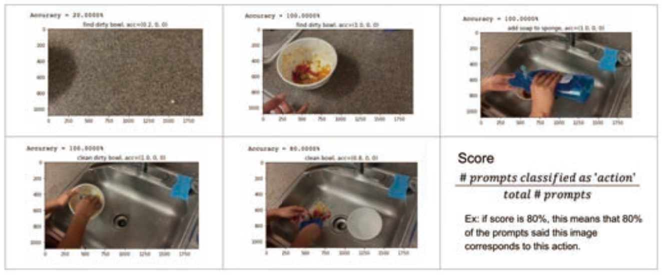

Accuracy = 20.0000%
find dirty bowl, acc=[0.2, 0, 0]
Accuracy = 100.0000%
find dirty bowl, acc=[1.0, 0, 0]
Accuracy = 100.0000%
add soap to sponge, acc=[1.0, 0, 0]
Accuracy = 100.0000%
clean dirty bowl, acc=[1.0, 0, 0]
Accuracy = 80.0000%
clean bowl, acc=[0.8, 0, 0]
Score
$\frac{\#\ prompts\ classified\ as\ 'action'}{total\ \#\ prompts}$
Ex: if score is 80%, this means that 80% of the prompts said this image corresponds to this action.

Fig. 7. Score progression for 'wash bowl' task. The model identifies each state and evaluates step completion. For example, in the step 'find dirty bowl,' the score increases from 20% to 100% from the first image to the second image

We also tested our method on task sequences encompassing a failure case. In the below example, we continue with the 'pour orange juice' task but purposefully include mango juice. The step 'find the orange juice' reaches a max score of 0.20 and the last step score is 0.50. Compare this to the correct sequence in Figure 5 where the score for the 'find the orange juice' step reaches 1.0 and the final score also reaches 1.0. The system correctly notices the error during execution and determines that there is a low certainty of the task being completed correctly. This shows that assessing task execution in a step-by-step manner can provide greater understanding of task completeness and more accurately recognize when an error occurs.

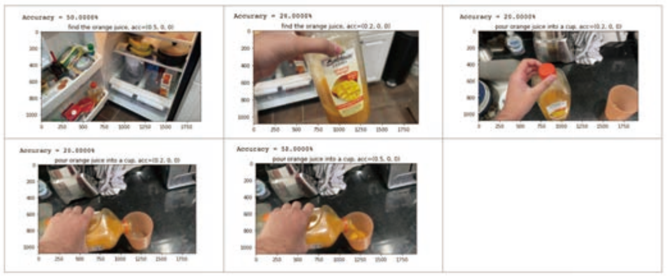

Accuracy = 50.0000%
find the orange juice, acc=[0.5, 0, 0]
Accuracy = 20.0000%
find the orange juice, acc=[0.2, 0, 0]
Accuracy = 20.0000%
pour orange juice into a cup, acc=[0.2, 0, 0]
Accuracy = 20.0000%
pour orange juice into a cup, acc=[0.2, 0, 0]
Accuracy = 50.0000%
pour orange juice into a cup, acc=[0.5, 0, 0]

Fig. 8. Score progression for 'pour orange juice' task when the task is incorrectly executed. The model successfully determines the task was not completed correctly and identifies that a mistake occurred (mango juice poured instead of orange juice).

D. *Full-sequence Results*

Once we had validated our system with task sequences, we moved on to experimenting with full-sequence videos. We provide a video for the "pour orange juice" task with a side-by-side score bar that updates in real-time during task execution. We include the results for "cut apple" and "wash bowl" in plot form (Figures 9 and 10) showing the score trajectory for each step in the sequence.

a) Cut apple task: The plot below shows the score vs. step when every image in the sequence is considered. We can see that for the 'wash apple' step, the score increases from 0.8 to 1.0 as the apple is being washed. The score drops to 0.4 in the final image in the 'wash step' sequence when the apple is clearly no longer being washed. This is an example of where the scoring metric is a good marker for step completion: the score increases as the step is being completed and decreases once the step is complete and it has moved on to the next step. We see a similar pattern with the 'cut apple' step where the score increases as the apple is cut up. However, there are some cases where the score is not as accurate: for example, in the 'find apple' step, the model gives it a low score of 0.2 even when an apple is in sight. This shows that the scoring metric is relatively effective as a metric for step completion but there is still a need for future work to improve its accuracy.

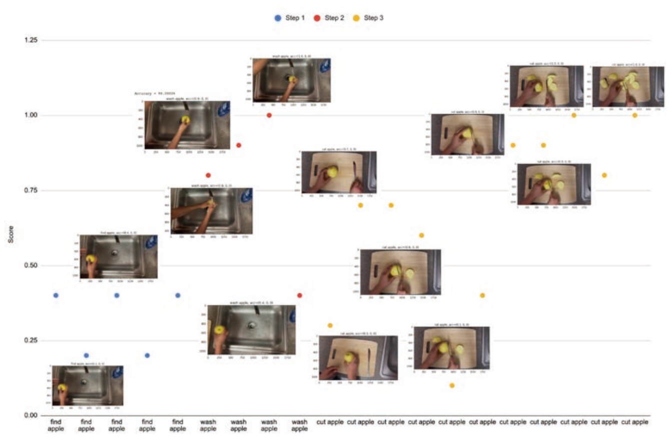


Fig. 9. Full sequence score progression for 'cut apple' task.

b) Wash bowl task: This plot shows the results for the wash bowl task. We can see that most of the images for 'wash bowl' have a score of 1.0, meaning that the model can correctly determine when the bowl is being washed. This is also true of the 'add water to bowl' step where there are three consecutive images, all of which involve water being added to the bowl and all are given a score of 1.0. The model can also differentiate between a dirty bowl and a clean bowl. The results for this task show that the model can not only differentiate between states but also determine when a step like 'washing a bowl' or 'adding soap to sponge' are being executed.

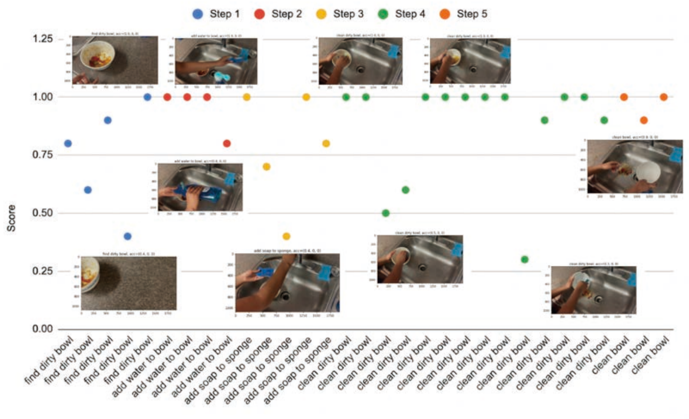


Fig. 10. Full sequence score progression for 'wash bowl' task.

V. Discussion

Harnessing the power and breadth of state-of-the-art vision and NLP models, we were able to create a system that could potentially serve in part or fully in the place of a human evaluator during experiments with a robot AI. With that being said, such systems will require much future work to be ready for real-world applications. This section will go over the limitations of the system and propose future work that can be conducted related to our system and methodology.

In an ideal scenario, the models could be trained specifically for the purpose of task assessment and response. However, it was only feasible for us to work with pre-trained models and therefore much of our efforts were towards working within the confines of what the model could interpret. Prompt engineering was an important step for us to understand how the visual language model interpreted captions as well as images. While validating our results, we tried different methods of prompting the model and discussed the need for a way to validate and measure the certainty of the model's responses. This is where our score metric came into play. Future work could simply focus on prompt engineering, assessing if there are better ways to prompt for the action by asking the LLM "Another way of saying {task} can be:" A disadvantage of this prompting is that the image sequence must be properly labeled, and will not allow for generalization to unforeseen tasks. The wording of the prompting is also important; a simple difference like "Find the orange juice carton" vs. "Find the orange juice box" impacts the certainty of the task at hand.

Another potential direction is ways to augment the data fed to the model. We found that random clipping, rotating, and brightness changes of images had no effect due to the robustness of the VLM. The VLM was confused by clutter, for example when pouring orange soda with some cooking scraps on the table, the model notices small peppers next to the drink and proposes they be added to the drink. An augmentation we made that was successful in validating our system was failure states. Future work could take multiple kinds of failure states, either an entirely new failure sequence or when collecting a task sequence taking failure tasks to be replaced with successful ones.

VI. Conclusion

The advent of GPT-4 and popularization of AI and large-language models have intensely influenced fields as diverse as engineering and the arts, and robotics is no exception. We have presented an automatic prompting system ART-E that can take a labeled video or image sequence, and harness LLM in conjunction with VLM to identify a task at hand and determine the likelihood that task steps have been completed. The system is surprisingly robust considering the minimal prompting conducted and more importantly highly generalizable to many tasks. The results gathered highly support a need for continued exploration into self-validating agents that can ultimately be implemented into future robotics systems.

Acknowledgment

We would like to thank Professor Shuran Song from the Department of Computer Science at Columbia University for her advice and guidance.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. 2022. Do as I can, not as I say: Grounding language in robotic affordances.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017.

Attention is all you need.

[3] Open AI. 2023. GPT-4 Technical Report.

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, AdityaRamesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

[5] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a visual language model for few-shot learning.

[6] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. 2022. Socratic models: Composing zero-shot multimodal reasoning with language.