**MEASURING CLEVRNESS: BLACK-BOX TESTING OF VISUAL REASONING MODELS**

**Objective:**

Applying behavioral test in the form of a two-player game on an existing VQA (Visual Question Answering) model to test its reasoning capabilities.

**Visual Question Answering:**

Visual Question Answering (VQA) is a computer vision task where a system is given a text-based question about an image, and it must infer the answer.

Let’s see an example for visual question answering.



When the above pic is fed into the VQA model, it should be able to answer the questions put to it like:

1. How many books are there in the shelf?
2. What is the cat playing with?
3. What is the color of the books?

**Need for testing the reasoning capabilities of VQA model:**

Though VQA models may be performing well on some images giving the correct answers for the questions put to them, there might be a possibility that those models are not have learned to generalize but have learned the training data(biased).

So, to test the reasoning capabilities of the VQA models, we use CLEVR, a diagnostic dataset that contains challenging images and questions for studying the ability of VQA systems to perform visual reasoning.

**Dataset used:**

We refer to this dataset as the Compositional Language and Elementary Visual Reasoning diagnostics dataset. CLEVR contains 100k rendered images and about one million automatically generated questions, of which 853k are unique. It has challenging images and questions that test visual reasoning abilities such as counting, comparing, logical reasoning, and storing information in memory. CLEVR is designed with the explicit goal of enabling detailed analysis of visual reasoning.

**How it works:**

We do a strong black-box adversarial test, which makes no assumptions about the underlying mechanics of a tested model, formulated as a game between two players. Our test does not require any direct access to the tested model, even through its sensory information. It does not require gradients, output probabilities, or any access to the perceived image and deviates from bounded perturbations and instead focuses on global scene manipulations that are still consistent with the task constraints and can change the behavior of a tested model.

We reformulate visual reasoning by integrating visual question answering with zero-sum two-player game frameworks. Under our novel formulation, a visual and adversary agents compete against each other through content manipulation.

We use an interactive framework with the communication channel between two players. The first player, which reasoning capabilities we are about to test, performs visual reasoning tasks, we call it Visual-QA Player. The second player, which we call the Adversarial Player, is manipulating the scene so that it fools the first player even though those changes still lead to correct reasoning steps among humans. Both players interact with each other only through questions, answers, and the visual scene. If the Adversarial Player manipulating the scene causes the Visual-QA Player to change its answer even though the new scene is still valid for the same question and answer, it is then the reasoning failure.

**Game between two players:**

Adversarial Player and Visual-QA Player. Adversarial Player uses a multi-modal module to extract features conditioned on the visual and textual inputs. After transforming such features with a feed-forward architecture, it samples an action using object-specific heads. Each action corresponds to manipulating the corresponding object in the scene. After alternating the original scene graph, we use various environment enforcers to ensure validity of the constructed scene. A valid scene graph is rendered and introduced to the Visual-QA Player together with the original image. Finally, we collect responses of the Visual-QA Player and calculate suitable rewards based on them, and we repeat the whole cycle during the training phase. Invalid scene it gets Invalid Scene Reward. This reward encourages producing scenes that pass the environment enforcers tests. Finally, if Adversarial Player does not manage to fool the model, it gets Fail reward.

A picture containing text, screenshot, several

Description automatically generated

The above image shows few examples of how the original scenes are changed in such a way that the answer to the question doesn’t change.

**GAME ARCHITECTURE:**

Diagram

Description automatically generated

The input visual is fed into the multi-modal module. And this is passed onto the feed forward network. Here the adversarial player uses the multi-modal module to extract the features of the feed input visual. The feed forward network transforms the extracted features and samples an action using object-specific heads.

And each action corresponds to manipulating the corresponding object in the scene. And the A2C (Advantage Actor- Critic) algorithm evaluates the action by computing the value function calculating the rewards which are consistency drop reward or accuracy drop reward.

**STATE-INPUT TRANSFORMER:**

To test our two-player game against a state-input Visual-QA Player, we use a transformer-based architecture that receives six types of input features – object sizes, object shapes, object materials, object colors, object positions, and question tokens – and uses a cross-modal attention mechanism. In that mechanism, queries from one modality attend to keys and values from the other modality. Each data point contains variable-length inputs, describing all objects in the scene. It also contains question tokens that compose the question itself. Hence, we do padding with a special token to the maximal input length. We set the maximal length for the objects tokens to be 10 ∗ 6 = 60 and 50 for the question tokens. All in all, we have 110 input tokens. Each token type is projected into a different embedding space and a learnable type embedding is added to it, separately for object and question tokens. For instance, emb(material) + emb(object). Furthermore, three special learnable embeddings are used as an additional input. We use them as queries to reduce the overall computational costs of the cross-attention mechanism. The input tokens (concatenated object and question tokens) form keys and values. In every transformer block, we apply cross-attention between all input embeddings and those three special tokens. This is repeated five times. As the final block, a feed-forward network (classifier) receives as inputs the three latent tokens and outputs answer probabilities.

Diagram

Description automatically generated

The state-input Visual-QA Player receives a direct overview of the scene, bypassing any need for any image renderings. We use State-Input Transformer as the multi-modal component of that player. Since such a model gets the perfect visual information as the input, it makes it more robust under scene manipulations. Therefore, the Adversarial Player tends to manipulate the scene so that objects are placed closely together.

So, Transformer based architectures can also be used for effective analysis of the reasoning capabilities of Visual question answering models apart from building Natural language processing and computer vision models using the method explained in this project.