UNIVERSITY OF GOTHENBURG

DEPARTMENT OF PHILOSOPHY, LINGUISTICS AND THEORY OF SCIENCE

EMBODIED QUESTION ANSWERING IN ROBOTIC ENVIRONMENT

Automatic generation of a synthetic question-answer data-set

Ali Aruqi

Master's Thesis:	30 credits
Programme:	Master's Programme in Language Technology
Level:	Advanced Level
Semester and year:	Autumn, 2021
Supervisor	Simon Dobnik, Nikolai Ilinykh
Examiner	Staffan Larsson
Report number	(number will be provided by the administrators)
Keywords	Embodied Question Answering, Question Generation, Spatial Relations, Synthetic Datasets, Multi-Modality

Abstract

Our work extends a dataset for Embodied-Question-Answering. Embodied question answering is the task of asking a robot a question about objects in a 3D environment, where the agent is expected to navigate the environment and find the entities in question and answer. The answer system consists of navigation and VQA components. Each question in the dataset is an executable function that could be run in the environment to yield an answer. The published dataset for EQA is EQA-V1, and it is a limited dataset that includes only two types of questions, color and location questions. We use the navigational data, required for training the system, from EQA-V1 and generate new questions of two more types, size and spatial questions. Our data extension is intended to better train the system and enhance its ability in performing the task.

Preface

Acknowledgements, etc.

Contents

Introduction

An intelligent robot must be able to understand and resolve references in its environment (Russell & Norvig, 1995). Our human ability to interact with our visual surroundings, manifested in language, stems from faculties such as perception and memory (Regier, 1996). Perception, in particular, is central to our physical experience of the world (Barsalou et al., 1999). We conceive the physical world through perception, and we express our conceptualisation of the perceptual experience in words (Lakoff & Johnson, 2008). Therefore, as Nilsson (2007) argues, the exhibition of intelligent behaviour is necessitated by having a notion of meaning that associates 'words' with the visual/physical world.

meaning

The meaning of words is not a mere psychological phenomenon. Concrete nouns, for example, have references in the physical world, with physical properties indicated by their meaning. The meaning of a word is, thus, not only bound up with linguistic characters and mental notions but also with some physical representation in the world. For example, the word "chair" is represented by its token-characters (c, h, a, i, r), contains a perceptual symbolism(mental understanding of the chair's attributes and functions), and refers to an entity with physical features in the world(Mooney (2008)).

In this triangular definition of meaning, 'vision' has an integral part of the meaning in which it represents the physical world to language. To recognize a chair, one should, for example, identify the existence of legs, seats, their sizes, and geometric shape in a visual scene. These properties of the physical reference of a chair can be clearly represented in a visual form. Therefore visual recognition is part of the conceptualization process that forms the perceptual symbol of an entity.(Barsalou et al. (1999))

Perceptual information, however, is more than just visual information. The properties of an object include other sensory information such as the smell, taste, and texture of an object. For example, the meaning of rotten food could be more understood if the food is tasted.

The formation of a symbolic representation (meaning) requires more than the recognition of perceptual information. The construct of a meaning (symbol) could only be formulated by the existence of knowledge about the relations of the attributes that form an entity.(Barsalou et al. (1999)) refers to the process of forming a symbol as 'componential' or schematic, meaning that a notion of meaning is a scheme that is conceptualized or constructed. These schemes can either be logically constructed in our mind in terms of holding valid truthiness about the world, or they can be based on incoherent conceptualization forming a false knowledge.

The full meaning is the perceptual representation and the knowledge about it. The meaning of rotten apple is fully understood when we construct knowledge of the negative aspects of eating it. For example, rotten apples have red-brown colors, and the stomachache that results by eating them leads us to form the belief that red-brown apples are different from all-red apples, not only by color but also other health/taste attributes, so we classify red-brown apples in a different category called "rotten apples." The knowledge about health implications and the attributes such as the colors and smell of a rotten apple help us categorizing the rotten apple in the category of rotten food. (Lakoff & Johnson (2008)) explains that this attributive characterization can be expressed, for example, in the way we do prototyping and categorization of entities.

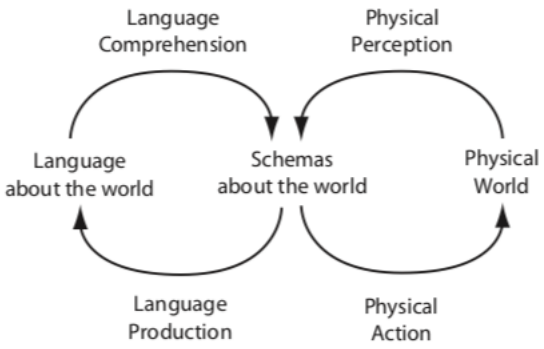


Figure 1: Roy (2005)

.

Interactions in the semantic world have an exchangeable nature. The interaction allows us to form meaning, and the formed meaning shapes our language and actions. In Figure 1 we see that the outcomes of our interaction in the world includes not only linguistic implications but also affects the actions(Roy (2005)). 'schemes about the world' are the beliefs we make from the interactions. For example, our negative experience with the red-brown apple made us form the "belief" that rotten apples are bad. The knowledge that "rotten apples are bad" influences our future actions- makes us not eat the apples with attributes of "rotten."

The process of comprehending meaning through associating attributes with each other to form a belief or draw a conclusion denotes the notion of reasoning. The process of classifying the apple as rotten includes multiple abstractions. We might first identity the apple by its general shape structure, then recognize, from previous experiences, that apples in red-brown color are not like all-red apples, then conclude that the apple is rotten. Reasoning is the ability to take the logical steps to conclude.

Grounding meaning

The approaches to ground meaning(form meaning) vary depending on the aspect of meaning that each approach focuses on. The different methods we review below-approaching meaning as a mental notion, a map of connected knowledge nodes, vision and language representation, or the combination of different aspects of meaning representation.

Word-meaning in a Vector Semantic Space (VSM) represents a mental aspect of meaning notion(Turney & Pantel (2010))). Space can be understood by imagining our minds as a space that we allocate meaning representation in them. In VSM, the mind (represented as neural language model) is an artificial space where word meanings are allocated at different distances from each other depending on their categorization, such as a rotten apple is closer to fresh apples than to a chair.

There are multiple hypotheses to representing word-meaning in a Vector Semantic Space(VSM). Distributional hypothesis is a popular example of word representation in VSM. The premise of this approach is that language is compositional, and word-meaning can be defined by its context–The meaning of a word is represented by the word and the words surrounding it Turney & Pantel (2010). The formulated word meaning representation is known as "word-embeddings" Mikolov et al. (2013). Using language to define language has proven promising in inferential tasks such as inferring that "university" and "student" are close to each other given their common context of "education."

Representing meaning with a combined linguistic and visual representation is the second example of an extensively researched field. Early research in combining vision and language used probabilistic learning by aiming at drawing an alignment between sentences, phrases, and words with the corresponding perceptual representations.Lowe (1999). An approach to probabilistic learning estimates the probability of a grammatical entity(text) being related to a perceptual representation Zitnick et al. (2013). A second probabilistic method is classifying each word in a sentence through the probability distribution of words over a perceptual representation. In Matuszek et al. (2012) Larsson (2015) we see examples of connecting entities of formal semantics(First Order Logic) with perception.

There are examples of research that attempt to incorporate knowledge graphs in the representation of meaning. Examples, Zhu et al. (2015), Zhu et al. (2014).

background

image captioning

Image captioning is an extensively researched field where visual-grounding is at the center of its focus. The methods of image captioning provide insights helpful for tasks that combine vision an language. In this section we review two main methods used in image captioning,feature extraction and attention mechanisms.

Feature extraction

Feature extraction methods can be divided into two main groups. The first group relies on statistical language models. The second group relies on encoder-decoder neural network model that deep extracts features.Wang et al. (2020)

Encoder-Decoder(CNN-RNN) Convulutional neural networks are at the core of feature extraction methods. CNN applications, nonetheless, take a vital role in many computer vision tasks. We see CNN and its modified models (such as recurrent-CNN) used in tasks as object recognition Liang & Hu (2015) Girshick et al. (2016) Ren et al. (2015b) , image classification Simonyan & Zisserman (2014) Krizhevsky et al. (2012),and semantic segmentation Hariharan et al. (2015) Long et al. (2015).

A main reason for using CNN for image processing is its ability to reduce the high dimensionality of images. Image features contain large sizes represented in pixels which would require large number of parameters to train. CNN reduces the dimensions of an image by learning how to process a matrix from a large window such as 250x250 pixels into a smaller one as 25x25. Through computing the convolution values of the image matrices and executing pooling computations, this process reduces the image into a smaller representation. The latter reduces the computational load and helps in processing and classifying the images faster.

The encoder-decoder caption generation has a CNN encoder and an RNN decoder.Vinyals et al. (2015) is an example of an end-to-end neural captioon generation model. In the neural model the CNN process the image features, and the last hidden layer passed to an RNN to generate a description. This method is a sequence modeling that is similar to machine translation. This means that image features are translated into words. The sequence is predicted by finding the probability of a certain description from a corpora given the features of an image.

RNN are known to be used widely in language technology applications. Rnn is used , for example in text-to-speech Arik et al. (2017) and machine-translation Cho et al. (2014),Wu et al. (2016). The advantage that the RNN gives to these tasks is that the output size is not fixed and that each output depends on the previous one. Such an incremental-sequence prediction is suitable for sentence predictions in respect to word dependency.

RNNs have a issue of vanishing gradient-descent. The gradient descent is an optimization algorithm that minimizes the error calculated in the loss function. Optimization, in brief description, is important for the learning process. It updates the model's parameters which determines the direction taken in the next time-step. This information is calculated given the input-output and the values of the parameters from the previous time-stamps. The gradients is reduced at every step due the value deductions in the activation function. When the gradient is reduced to almost zero value, it will be updating the parameters with no useful values, and therefore, learning seizes to improve.

Long-Short-Memory network (LSTM) provides a good alternative for avoiding the disappearing gradient. The gradient in RNNS vanishes in long sequences where the gradient keeps reducing. The architecture of the LSTM allows it to keep information stored for very long sequences. The latter gives it the ability to control the values of the gradient by updating it with information stored in the 'forget gate' form previous steps, preventing the gradient from vanishing.

Feature extraction- statistical language model Statistical language model, as in Fang et al. (2015), generate descriptions in three stages. First it detects words in an image using a convolutional neural network (CNN) for extracting image features. The incorporation of language at this stage happens using multi-instance learning(MIT)Zhang et al. (2005). The second stage, the statistical language model detects the most likely sentence to make of the words from a pre-defined corpus. In the third stage the sentences undergoes a re-ranking stage where the sentences are combined to generate captions.

Dialogue and VQA

In this section of the text we discuss the capabilities of computers to exhibit more intelligent behaviour. Image-captioning and its methods showed an insight to how much computers could see and understand what its seeing. However, acquiring language in the visual world would require computers to be able to communicate what it sees. Otherwise, in order to say that a computer is visually or linguistically intelligent one should imagine the computer having to pass the Turing test in a visual surrounding.

Researchers attempt to improve systems that are capable to hold a dialogue with a visual content. Das et al. (2017) trains a system in encoder-decoder model on a data set of 2 pairs dialogue with an image content.Skocaj et al. (2011) trains a system on learning concepts with visual content in an interactive-learning approach. In the similar context of improving systems that are capable of having more natural interactions, we see example in Lin et al. (2014) of a VideoQA.

To make a true statement about the computer's capability to engage in a visual dialogue, it must be first ensured that the computer actually understands the questions being asked to it. Otherwise, dialogue is very complex with many elements determining its succession. In a dialogue with visual content, the computer must, furthermore, understand the questions within their visual context. It is reasonable that we see increasing research on "Visual Question Answering" and less on visual dialogue as a whole. Improvements in VQA intuitively means that we are moving closer in the direction of having an interaction with a computer in a visual dialogue.

Antol et al. (2015) is the first notable data-set published for Visual Question answering (VQA). The data-set consist of open-ended and free-form questions. The data contains 250,207 images from MS COCO Lin et al. (2015) and other abstract scenes.The question types in the dataset require a range of different capabilities such as common-sense reasoning, knowledge-based reasoning, object-detection and active recognition.

Data-sets that use MS COCO scenes such as Gao et al. (2015), Yu et al. (2015) in addition to Antol et al. (2015) used human workers to write the texts for the scenes. Other data-sets are generated automatically such as Ren et al. (2015a).

Zhu et al. (2016) introduces a unique QA data-set. The Visual7W consist of questions about an image with objects marked with regions in the image. Object grounding with image region introduced in Krishna et al. (2016) contains the largest data-set with regions for both VQA and Image-captioning. Object-region approach is intended to improve visual grounding, by marking the regions of the image that the strings refer to.

Embodied Question Answering

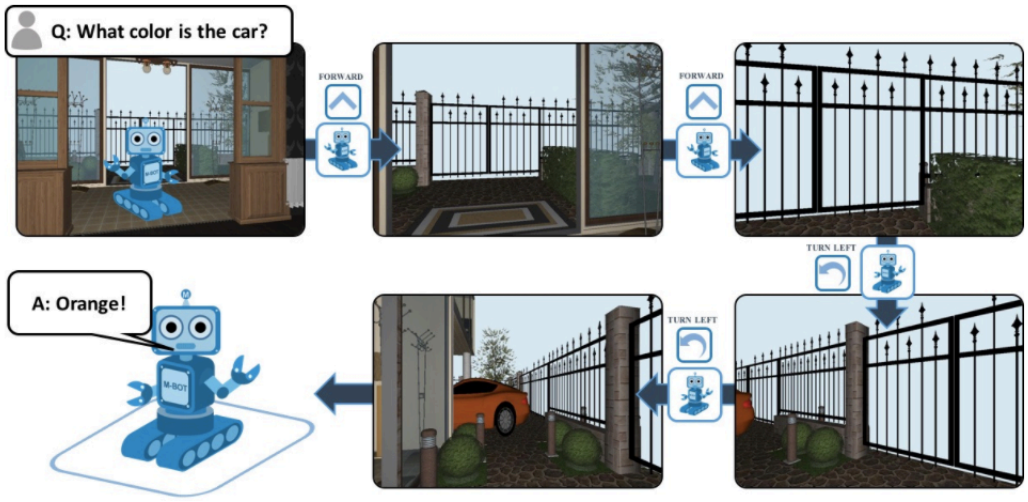


Figure 2: The Robot is asked a question at a start position. It needs to look around, collect information and decide on the next step to take. When it recognizes the car, it stops and processes the scene to answer the question

.

Embodied Question Answering (Das et al. (2018)) [1] is a new interactive task presented as one of the tasks within the Habitat Platform (Savva et al., 2019; Szot et al., 2021)[2]. The idea of the task is to allocate an agent at a random position in a 3D environment and ask it a question. To answer the question, the agent must intelligently explore the environment, collect information, and successfully navigate to the entity in question. EQA system navigates based on common reasoning, through an egocentric view, more or less imitating humans, it should be able to answer itself the common questions of ”where am I?”, ”where to go next?” and if asked a question about the car, as seen in 2, it should be able to reason that cars are usually situated outside or in the garage and look for the exit. Once it navigates successfully to a point where it recognizes the car, the robot should stop and answer the question.

[1] Link to the official page of EQA. It also includes other published papers about the task `https://embodiedqa.org/`.

[2] Github link to the Habitat Platform. Information and code about EQA and the other tasks within Habitat can be found there `https://github.com/facebookresearch/habitat-lab`.

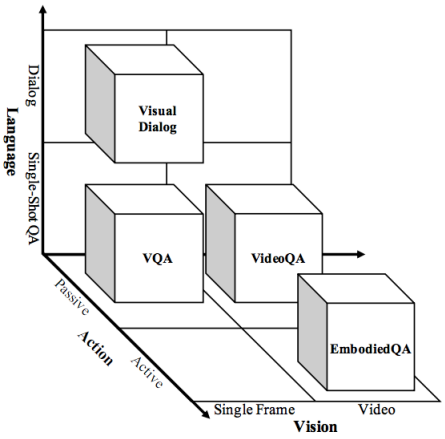


Figure 3: EQA in relation to other vision&language multi-modalities

In figure 3 we see where EQA stands concerning other vision-and-language multi-modalities discussed earlier. In the language domain, we see single-shot and dialogue. VQA is a typical example of a single shot interaction, where the system is designed to take a single shot question and a visual scene and output an answer. On the other side of the language domain (dialogue), we see Visual Dialogue, where the interaction within a visual context is continuous. On the vision domain, we see that VQA Visual dialogue is distinguished from VedioQA and EQA by the visual input type. The robot in EQA is continuously moving while navigating, so it inputs the vision similar to videos. Finally, EQA is distinguished from the rest of the multi-modalities on the action domain by being active. Hence, action here refers to executions of commands in a physical space. EQA is action-active by its navigational functionality. The rest of the modalities are passive with no functionalities of physical action execution.

The novelty of this system is that it presumably solves the problem of navigating and performing tasks in unseen environments. Many of the earlier studies that deal with navigation, such as Kruijff et al. (2007),Lauria et al. (2001) require the system to have a localized map of the environment to be able to navigate in it. The problem of localization in robotic navigation is known as Simultaneous Localization and Map Building(SLAM) problem. SLAM is a problem where a robot should map an unknown environment without a GPS or local map. Simultaneous localization is when a robot discovers it is surrounding and simultaneously construct a map while aware of its changing location. This means that the robot should extract information from its surroundings and learn the map as it goes Grisetti et al. (2010) Dissanayake et al. (2001) Zhang et al. (2018).

The answering system in the robot consists of two core components. The first is navigation, and the second is Visual Question Answering. In principle, the task should be performed in conjunction between the Nav and the VQA model. The navigation should lead the robot to a correct viewpoint then freeze its move. The VQA model should then take static image frames of the scene from the viewpoint where the Nav stopped and answer the question. However, the system's design allows it to exclusively perform either navigation or visual question answering on baseline models. The ability to train and evaluate either of the modules is possible due to two different training setups.

Training setups

The first setup is a connected system with training in Reinforcement Learning setup. The training of the robot in RL happens based on the answer-based evaluation. The robot is rewarded if it completes the whole task using the two components connected. The basis of evaluation in the RL setup is the answer prediction. The system is rewarded if it answers the question correctly, and to answer the question cor-

rectly, it needs to navigate to the right place and stop at a good view position so that the VQA system could have a relative and informative visual scene in order to answer the question. However, the researches in Das et al. (2018) elaborates that the system performs poorly when trained combined in RL. The navigation in the RL setup tends to position itself inaccurately at the stop-goal, which leads to passing distorted images to the VQA model. ”Noisy or absent views” would confuse the question-answering model Das et al. (2018). For the mentioned reason, there is no available RL-based system available for developers.

The second setup is a system with the Nav and VQA components trained separately and differently. The navigation is trained in the 'Imitation Learning' setup, and VQA is trained in Supervised Learning.Hussein et al. (2017) describes Imitation Learning as learning with a teacher, where a robotic system has to mimic the steps taken by its tutor. Imitation Learning is considered an effective solution, in particular, for navigational problems as its step-to-step learning restricts the freedom of systems; We see IL popular, for example, in navigational systems of ground vehicles Silver et al. (2008). The available Nav and VQA models that are available for training and evaluation in the habitat platform are the baseline models. The details of the training and the data used in each component will be described in more detail in the coming sections. The main point we attempt to convey here is that the answering system being researched is one with Nav and VQA components trained and tested differently.

Data

The dataset for the EQA task is called ”EQA-MP3D,” and it is a synthetic dataset generated automatically.[3] The EQA-MP3D task dataset is applicable for navigation and VQA, meaning that training the navigation and VQA use the same dataset. We refer to each question-answer in the EQA dataset as an ”Episode” because each QA sample includes a complete trainable navigation episode. We could describe the QA episode as a function executed in a 3D environment to yield an answer.

The 3D environments used in the task are indoor environments from the Matterport 3D(MP3D) dataset Wijmans et al. (2019).[4] The MP3D is 3D constructed scene dataset which contains 90 segmented houses. The EQA trains the robot in 57 MP3D environments and tests the robot in 10 other unseen MP3D environments.

Data in the Navigation training In each QA episode, the information mainly used for navigation is a question, an ID for the 3D environment, a unique starting position, a destination goal, and a path to the destination. The mentioned navigational information, excluding environment ID and question, are all represented in frequencies of coordinates. The starting position indicates where the agent should be spawned relative to the given environment. The path is the shortest path that the agent would take to reach the goal, consisting of steps and rotations. The shortest path is data used for Imitation Learning as the robots have to imitate the steps found in it. The goal is the stop point that marks the end of the episode. The stop point of the navigation is the viewpoint of the entity in question.

Data in VQA training In each QA episode, the information used for training the VQA model is an ID for the 3D environment, a question, ground truth answer, and the position of view. The mentioned information is automatically taken in a code part of the Habitat platform and reconstructed into a conventional VQA dataset, QA pair, and a visual scene. The visual scenes are extracted using the view positions given

[3] The dataset can be found on the Github page of the Habitat Platform, attached in the main page in the section' Task Data-sets' `https://github.com/facebookresearch/habitat-lab`.

[4] The GitHub reop of the Matterport 3D `https://github.com/niessner/Matterport`.

in each QA episode in the corresponding 3D environment represented by the ID.

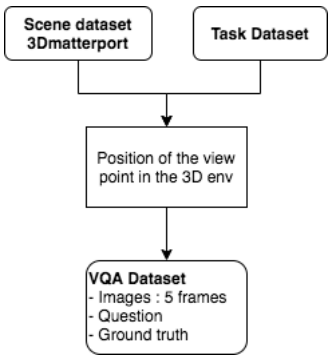


Figure 4: Locations of viewpoints of the entity taken from an EQA episode to extract a visual scene. The visual scene is then constructed with a QA pair to form a VQA sample of 5 frames of images, question and ground truth

.

The extracted scenes for VQA consist of 5 frames images taken from the viewpoint where the navigator is supposed to stop. In figure 4 we see an illustration of the structuring of the VQA dataset using the EQA task dataset (EQA-MP3D) and the scenes in Matterport 3D. The resulted VQA for VQA training is a Question-Answer pair with a visual scene.

Data-set size & Question types The question-answer data set contains three types of questions. Each question type is generated in a string template. The templates are as the following:

- color_room template: ”what color is <obj> in <room>?”In these questions, the agent needs to find the room in question, look for the object, and answer the question. For the agent to be successful at reaching its target, it needs to know the difference between rooms, and objects, by implicitly recognizing that a certain room is a living room and not a bathroom and such.

- color template: ”what color is <obj>”. The difference between ”color” type and ”color room” is that no room is specified in the ”color” type of question. In the ”color” type, the agent needs to figure out where to look by itself. For example, ”what color is the fridge?” the robot needs to implicitly figure that the fridges are usually in the kitchen and navigate to the kitchen to answer the question. In other cases, the object could be in the vicinity of the robot’s starting point so that all it needs to do is to look around.

- location template: ”What <room> is the <obj> located in”.

In EQA-MP3D, each object in a question is unique to the room. The latter means that for an object to be selected for a question, there needs to be only one instance of that object existent in the room. The reason for this is to avoid ambiguity and not to confuse the agent if there happen to be more instances of the same object in the room.

There is a total of 11496 question episodes in the train split and 1950 question episodes in the ”Val” split. As seen in figure 5, in the train split, there are 1830 episodes of ”color” type, 8031 episodes of ”color room,” and ”1635” of location type. For the validation split, there are 1335 ”color room” questions, 345 ”color” questions, and 270 ”location” questions.

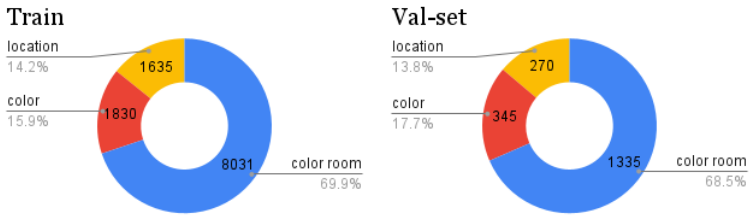


Figure 5: Number of question-answers represented by their types in the Train and Validation set

However, the number of unique question-answers is different from the number of episodes—a unique question-answer as a question-answer of the same strings and visual scene. For every unique question-answer and goal (scene), there are 15 different starting positions and shortest paths for the robot to train on for navigation. This means every unique QA in VQA is repeated 15 times. For example, in the validation set, the number of unique questions (same QA and goal-scene) of "color" type is 23, we multiply it with 15 (the number of starting positions for every unique goal), and we get 345, the number of episodes for "color" type in Val-set as seen 5. In the train set, the number of unique visual-question-answer for "color_room" is 536, for "color" is 122, for "location" is 109. In the Val-set, the number of unique visual-question-answer for "color_room" is 89, for "color" is 23, for "location" is 18.

Data Bias In all color questions (color & color_room) in the train set, we observe a total of 153 unique textual references. A 'reference,' in this example, is a string that can refer to a specific entity in a specific or non-specific space. For example, 'sofa' in questions like "what color is the sofa?" is one reference. "sofa in the living room," like in color_room questions "what color is the sofa in the living room?", is a second reference. "sofa in the bedroom" like "what color is the sofa in the bedroom?" would be a third and different reference. In order to gain insight into the data, we normalized all the color questions by reducing them into questions of a reference type and collect the number of answer choices found for each reference type in all color questions.[5]

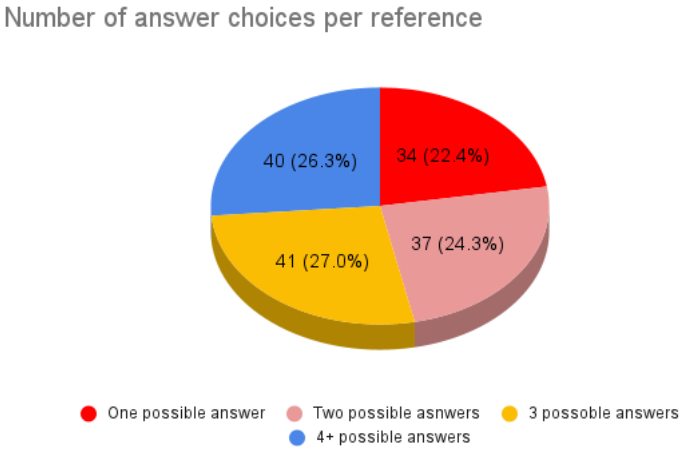


Figure 6: The color questions in the training data sorted by textual reference. In total we find in all color questions 153 references. 22.4 percent of the references have one color answer as the only choice.

[5]Link to the statistical analysis of the data in a notebook `https://github.com/Al-arug/EQA`.

.

We find that 22.4 percent of the references have one color answer as the only choice of answer, as seen in figure 6. This means that 22.4 percent, around 2208 of all the 9861 ”color” & ”color_room” QA episodes in the train-set, have one possible color as an answer. This means, in these cases, the model does not train on disambiguation any classes of color for the reference in the question. Instead, these data samples would tell the model that there is only one possible answer to memorize for this reference. (Appendices contain all the textual references found in the question)

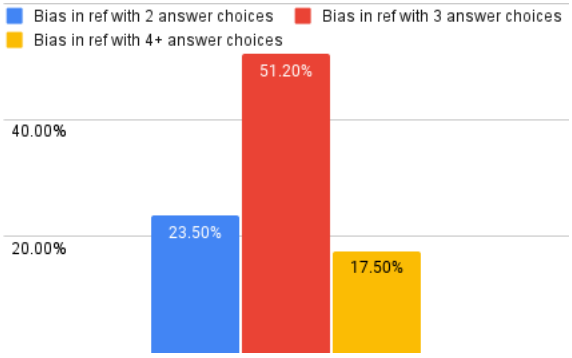


Figure 7

The second type of bias is the dominance of one color over the other choices in the question-answers with multiple color choices. In 7 we categorize references per answer-choice. References that has two answer choice in one category, references with three answer choices in one category, and references with 4 answer choices in a different category. The bias for each category is determined differently. In the references with category two answers, a reference is considered to contain biased QA if the answer in 75% of the instances of the answer is the same . For the categories 3 answer choice and 4 answer choice, biased is considered if one answer made up 50 percent of the answers in each reference. In total we get that 23.5% of the references with two answer choices are biased. 51.2% of the references with 3 answer choice are biased, and 17.5% of the references with 4 or more possible answers are biased.

Navigation Model

Habitat’s navigation is referred to as PACMAN. It consists of two core components, planner and controller. The planner takes inputs from the vision and language model, and the encoding of hidden-layer and action of the previous time-step then outputs action-decision.

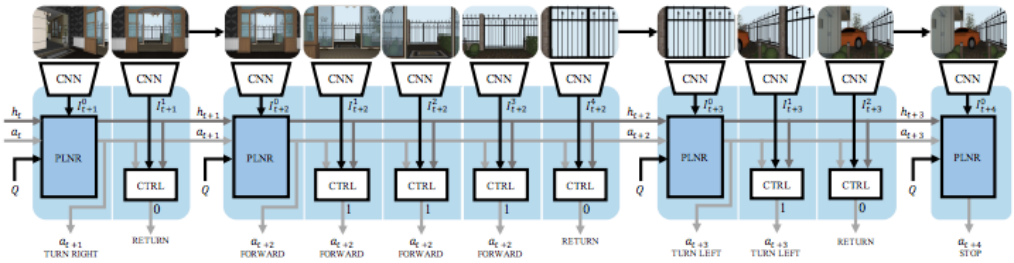


Figure 8

The controller takes the previous hidden state and action decision and executes the action. As seen in 8, visual input is passed to the control then the controller classifies the following decision of two possible

decisions. Either to repeat the last action given by the planner or to return to the planner. The controller can repeat the same action maximum of five times then it automatically returns to the planner.

Visualization of the navigation is in figure (1). T stands for the planner's time-steps, t = 1,2,3...., and N(t), n = 0,1,2,3.. denotes the controllers time-steps. The denotations of symbols explained clearer in the quotation :

"I_t^ndenote the encoding of the observed image at t-th planner-time and n-th controller-time. The planner is instantiated as an LSTM. Thus, it maintains a hidden state h^t (updated only at planner timesteps), and samples action $a_t \in \{forward,\ turn-left,\ turn-right,\ stop\}$ "p(6)

For example, the first step-decision from the planner is denoted as such:

$$a_t, h_t \leftarrow PLNR\left(h_{t-1}, I_t^o, Q, a_{t-1}\right)$$

The planner computes the next step-action a_{t+1} from input of the previous hidden layer (h_{t-1}), question encoding (Q), the previous action a_{t-1}, and the image input given to the PlNR (tI_t^o).The planner selects the action a_{t+1} and update the hidden state h_{t+1} then passes the control to the controller.

The controller decides to either repeat the action or return control to the planner. The controller's classification is based on the current hidden-stateh_t and current action a_t and the image observation from the planner + the image given at the controller's time-step. The denotation of the classification is as such:

$$\{0,1\} \ni\ c_n^t \leftarrow CTRL\left(h_t, a_t, I_t^n\right)$$

"if $c_n^t = 1$ then the action a_t repeats. Else $c_n^t = 0$ or a max of 5 controller-times been reached, control is returned to the planner"p(6). The h_t a_t coming from the planner act as an intent. The controller, initiated as "feed-forward multi-layer perceptron with 1 hidden layer",repeats and controls the action in order to align I_t^n with intent given by the planner.

VQA Model

The VQA model is a CNN-LSTM architecture. The CNN encodes 224x224 RGB images with a "multi-task pixel-to-pixel prediction framework" (p6) encoding. The structure of the CNN4 5x5 Conv, Batch-Norm, ReLU, 2x2 Max-Pool blocks, and they produce a fixed-size representation. "The range of depth values for every pixel lies in the range r0, 1s, and the segmentation is done over 191 classes" (p.11) Das et al. (2018). The "lstm" is a 2-layer LSTM with 128d hidden layers.

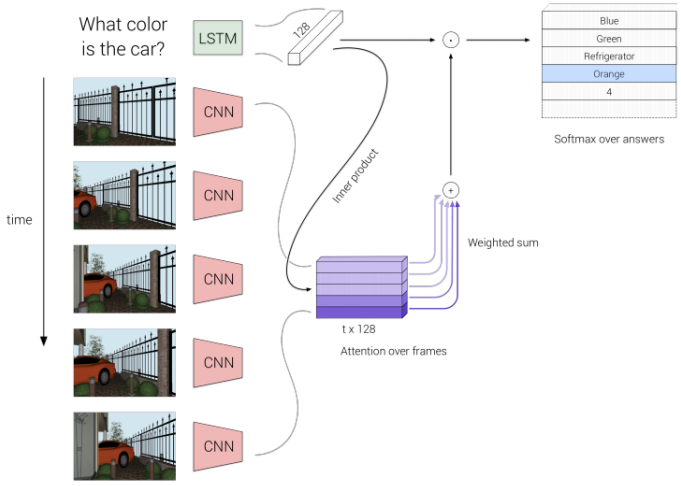


Figure 9: Architecture of the VQA model consist of and LSTM for language encoding, CNN for vision. The system is trained to combine the two with attention

The CNN extracts features from five images(5 frames) scene, and the LSTM encodes the textual features of the question. Combining the visual and linguistic features is done through computing similarity via dot product and concatenation. First, the similarity between each image and the question features is computed via dot product. A softmax converts the question and image similarity into attention wights then the question encoding is concatenated with them. The concatenated features are then classified in a softmax, where the answer probability is distributed over 172 answers.(Das et al. (2018),p6)

Problem

In an experiment we conducted on the VQA model, we observed that the system tends to answer the questions relying mainly on the textual input in the questions (bias)[6] The idea of the experiment was to give the model a random image instead of the original scene and see if it affects its predictions. The results showed that the system gave correct answers despite the absence of the corresponding scene required to answer the question. In such a case, the system's performance would typically have worsened, not improved, as the required visual information to answer the question is missing. The correct answering by the system was demonstrated in an overall increase in the performance score. Its ability to answer correctly demonstrates its reliance on the language model to predict the answer.

The system's ability to predict answers correctly in the experiment indicates a lack of visual grounding. We draw this conclusion from the observation that vision did not influence the predictions. This means that the system, in training, has not learned a scheme for word-meaning in association with vision. Grounding language in vision is when we connect the "high-level" symbolic representations such as language to a "low-level" non-symbolic representation such as the sensory (visual) features. The ability to ground language in vision is essential for any task requiring "seeing" and attending answer. If a robot successfully learns to align and combine the two types of representations, one could say that the computer understands what it sees (visual grounding). When a system fails to achieve such a connection, we define the problem as the "Symbol system problem" (Harnad (1990)) or 'lack of visual grounding.'

We presume that the lack of visual grounding is attributed to bias in the dataset. Earlier in this text, we

[6]Link to the experiment "Testing VQA's reliance on vision "`https://github.com/Al-arug/Habitat-Project`.

reviewed textual biases within the EQA dataset. Having biases in the dataset would hinder the learning process, as it gives the model a way to learn to avoid combining vision and language by giving correct answers without actually learning to combine the two types of data.

We also observe that the type of questions asked are simplistic and can be considered unnatural. The existent color questions in the dataset are not the type of questions that a human would naturally ask. The limited types of questions found in the dataset seem to be meant to simplify the robot's task with a primary focus on navigation.

Problem in a context

(Selvaraju et al. (2020),Goyal et al. (2017)) and other research within the VQA point out the problem where models learn biases in training and manage to give good results in the testing.Johnson et al. (2017) elaborate that the underlying issue here is that the model answers by memorizing prior textual information. For example, a neural network might answer the question "What covers the ground?" correctly by answering "snow," "not because it understands the scene but because biased datasets often ask questions about the ground when it is snow-covered." Fukui et al. (2016) clarify that the models' answer-cheating is demonstrated when a VQA system primarily relies on the language model and ignores the visual information. Such a learning problem is crucial because it makes it challenging to evaluate the model's improvementsAgrawal et al. (2018).

When a system cheats its way into answering the questions, it shows a lack of visual groundingGoyal et al. (2017). Visual grounding (understanding the meaning of words about vision)is crucial because we want the systems to understand the reasoning steps that humans would logically take to answer a question Agrawal et al. (2016),Zhang et al. (2016), Fukui et al. (2016). For the system to be able to reason its way to predict an answer, it must first capture the full meaning. Selvaraju et al. (2020) explains that learning to reason would require the systems to make inferences at "multiple levels of abstraction." For example, "is the banana ripe?" where it would instantly answer "no." Answering this question would require the system to rely on perception to answer sub-questions such as where is the object? What are its shape, size, and color? Then reason that the "yellow" color indicates ripeness.Selvaraju et al. (2020)

Research Questions

- How can we extend the dataset with more sophisticated and natural questions? (A useful robot should answer a variety of questions.)

  Adding new questions could help test the system's capabilities, but more importantly, we consider it a step to enhance the system's cognition. The VQA system that we are improving is part of a robotic system that should ideally be helpful for human use. Social robot's usability is very dependent on its exhibition of human intelligence Fong et al. (2003).

- How does the VQA system perform with the new question types?

- Does asking questions of spatial and size types improve the system's attention to vision? (Evaluating it based on the performance on color questions)

Methods and materials (proof reading required)

Overview

This section describes the methods and materials sources we use for question generation and the training&evaluation of the EQA system. The primary method relies on using utilities provided by the habitat platform. The name 'Habitat' is derived from the notion of learning within and from an environment(Savva et al. (2019)). The utilities in the platform provide necessary arrangements for the EQA task, such as simulating and working in a 3D environment, insinuating a robot with a specific configuration, preparing data sets, and setting up models for training. Materials are mainly used for question generation. The materials consist of data sets that contain semantic annotations and other geometric information essential for generating trainable episodes for navigation and VQA.

The two components providing utilities in the Habitat Platform are referred to as 'Habitat-sim'[7], and 'Habitat-Lab'[8]. Habitat Simulator is a 3D simulator with multiple functionalities, such as facilitating configurable sensors and robots in 3D environments. The habitat lab is a library that contains multiple tasks that can be performed in the environment. The lab provides different models and training setups. In the following sections, we describe the two components and their usage in this project.

The material we use for question generation is extracted from EQA-MP3D and Matterport3D. The environments in MP3D contain semantic annotations necessary for generating questions and the general linguistic understanding of the space. In addition, the annotations come with geometric information about the entities in the house, such as coordinate locations. The geometric information is essential for understanding the space geometrically for navigation as well as for question making. For example, asking a question about a spatial relation between two objects requires knowing where they are located. The task dataset, EQA-MP3D, provides data for navigational training and information about the objects in the questions. Following the Habitat Lab and Simulator description, we outline the relevant material in the two data sets and explain some of the concepts necessary for understanding the usability of the extracted data.

Habitat Simulator

Habitat-sim simulates 3D environments assimilating real-world settings. The environments are based on constructing either synthetic or real-world based scenes. Szot et al. (2021) describes a simulator as a system of two parts, physics engine, and renderer. The physics engine generates physical phenomena such as gravity and the physical state of the environment throughout the simulation, and the renderer completes the physics engine's work by outlining objects, colors, and borders. When constructing the scene, habitat-sim can do environment state manipulation by changing the layout of objects Savva et al. (2019). We observe a change of object's layout, for example, in the names of objects in Matterport3D extracted from habitat-sim and the ones found in the MP3D annotation files.

Habitat-sim has efficient GPU usage and can simulate different environment data sets. The simulation is displayed on GPU devices, which usually would require big storage of GPU to display a simulation of houses. The simulation setups in (Szot et al. (2021), Savva et al. (2019)) allows for smaller storage GPU's to perform the simulation. The latter makes it possible for an unfamiliar user of 3D simulation to simulate on their machines with reasonable speed. For the data set part, the simulator is designed with generalization for simulating different 3D Detests. In addition to MP3D, it supports 3D simulating for

[7]The GitHub repo for Habitat simulator `https://github.com/facebookresearch/habitat-sim`
[8]The GitHub repo for Habitat Lab `https://github.com/Al-arug/habitat-lab`

the following 3D datasets: GIBSON Xia et al. (2018),ReplicaStraub et al. (2019).

Habitat simulator facilitates the employment of configurable sensors and agents. Configurations such as the location to spawn the agent and the type of sensors, and their position on the agent are the types of flexible settings given to the simulator to act upon. "Sensors" is a different name for referring to the CNN decoders where each decoder can be seen as a sensor of the following: 1) RGB reconstruction, 2) semantic segmentation, and 3) depth estimation. The latter sensors are used to obtain "object attributes (i.e., colors and textures), semantics (i.e., object categories), and environmental geometry (i.e., depth).". The agent can be configured with or more of the mentioned sensors depending on the task. For navigation, the agent is configured with "depth" and "RGB" sensors. Depth sensor is essential for the agent's capability to navigate. With a depth sensor, it could estimate distances and avoid colliding with obstacles. For VQA, the agent is configured to output only "RGB" images of 5 frames.

Habitat Lab

Habitat lab can be described as an API that facilitates task training in connection with the 3D simulator. In addition to initiating the simulator, Habitat-lab provides trainers, neural models, and data loaders. The lab has a hierarchical structure that acts given different configurations. For example, the steps for training/evaluating a task would be to prepare the data, initiate models and trainer, then simulate 3D environments. The information about the required model and the data for each task, such as paths to data and the models to use, can be manually given/changed in a task's configuration file.

In addition to facilitating the training of whole tasks, the lab can train models separately. Concerning EQA, the CNN model, in particular, is trainable independent of the other components using the lab platform.

Training and evaluating the navigation and VQA is possible on baseline models[9]. The text-image attention model is trained in connection with the pre-trained CNN[10] [11]. For navigation, the platform provides a training setup for Imitation Learning.

As mentioned earlier, for a VQA training and evaluation session, the lab facilitates preparing the VQA dataset in a conventional VQA format. Data preparation and setting up the models & trainer acts upon the paths and settings given in the VQA configuration file [12]. The configuration file also includes some manually instructed commands for configuring the robot, where the lab passes these instructions to the simulator. The simulator is initiated simultaneously while preparing the VQA dataset. The data loader of the lab takes each environment ID from each QA sample in the EQA dataset and extracts the image frames.

Besides facilitating EQA training, the lab contains training setups for other tasks. The trainable tasks in the platform are as the following Goal navigation where the robot has to navigate to a geometric point; object navigation where the system has to navigate to an object; pick-up task where the robot has to pick up an object and move it to a different location, language-vision task where the agent follows directional

[9] Instructions for training and evaluating the baseline models in the API `https://github.com/facebookresearch/habitat-lab/tree/master/habitat_baselines/il`

[10] File containing the VQA trainer `https://github.com/facebookresearch/habitat-lab/blob/master/habitat_baselines/il/trainers/vqa_trainer.py`

[11] File containing the VQA model `https://github.com/facebookresearch/habitat-lab/blob/master/habitat_baselines/il/models/models.py`

[12] Configiration files for baseline including VQA config `https://github.com/facebookresearch/habitat-lab/tree/master/habitat_baselines/config/eqa`

instructions. Each of the previously mentioned tasks has a separate data set.

Data and Data-sets

In this section, we review the materials used in generating QA episodes. The first part is a review of data structure and the relative semantic annotations found in MatterPort3D. The review of MP3D also includes an elaboration of geometric and viewpoints concepts necessary to understand the geometric annotations in MP3D and their usage. The second part of this section includes a review of the EQA-MP3D dataset structure and content.

EQA-MP3D is seen as a method and material source. The method of generating QA episodes relies on imitating EQA-MP3D. The imitation of EQA-MP3D ensures that the newly generated QA episodes are executable in the habitat platform by matching code requirements. Having the same structure for the generated question-answers as EQA-MP3D, a review on EQA-MP3D would also help imagine the QA form that this project achieves. In the same review, the navigational data we take as a material source for navigational training are highlighted and explained.

Semantic annotations in Matterport

Annotations In the Matter port annotations, each house environment has three files. The three files are x.house,x.ply, and x. We collect the annotations from the x.house files house.

Each house file (x.house) has eleven line-types of annotations.. The lines are marked by a capital letter as a marker; the first letter-marking to the last letter areas in this list [H, L, R, P, S, V, P, I, C, O, V]. Each letter-marker symbolizes a certain type of information. In this section, I will explain only the type of information that we use in this project.

The line representing an object's info in a house file begins with the string "O". The "O" lines contain information about the objects in the house. Every line that begins with an O letter consists of one object in the house with corresponding information about its geometry and location within a room and level floor. Each "O" line looks as such: [O object_index region_index category_index px py pz a0x a0y a0z a1x a1y a1z r0 r1 r2 0 0 0 0 0 0 0 0 0]

The object's data in the line seen above comes in a string form, and each section in the string represents different types of information. *Object_index*, the index of an object is what we refer to as the object ID. *region index* is the room ID. *category_index* is the object's index in the category map; this index is used to obtain the object's name from the category map.*px py pz* represent the center of the box in (x,y,z) axis. *r0 r1 r2* represent the radius of the object from the center on the (x,y,z).*a0x a0y a0z a1x a1y a1z* these are the rotation of the OOBB radius(OOBB and radius will be elaborated on in a coming section). Finally, the last "0" s in the line have no meaningful value and therefore are ignored.

Point of views, geometric data As seen in the previous section, the geometric information consists of elements as the location of an object, region, or level, defined by their center in a world coordinate system. Other information is the size of the entity given its radius from its starting location (center).

The camera views of the scenes are globally oriented Chang et al. (2017)(p3). A way to allocate an object is to find its location under global coordinates. Let's say the global coordinates start from the center of

[12] https://github.com/niessner/Matterport/blob/master/data_organization.md

a house where the center of the house is (0,0,0) on the (x,y,z); and let's say all the objects are spawned throughout the house's (x,y,z) axis where its distance defines the location of each object to the house center. When annotated, the objects are viewed through a camera. The description of their geometric location, thus, should consider the view-position of the camera.

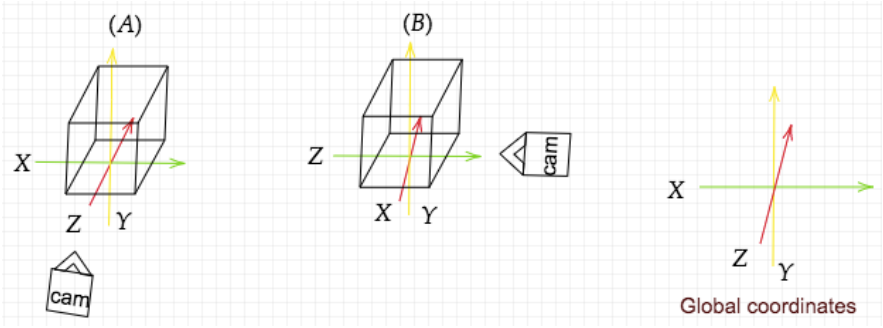


Figure 10: The camera in graph A views the objects from global perspective(readers perspective). The view of the camera in graph B is rotated. The rotation is resembled in the the axis's representation

In graph (A) in 10, we see that the camera-view of coordinates aligns with the global coordinates. The (x,y,z) that go through each object in graph(A) and graph (B) are the view of the axis in reference to the camera. However, if the camera is positioned to the right of the object from our view, as in graph (B), then we say that the camera view of coordinates is not aligned with the global view. We notice in graph (B) that from the camera view, the "global X" is "Y" and vice versa.

Some geometric calculations cannot be performed if the location measurements are not relative to each other. For example, if we want to calculate the distance between objects, the locations must be consistent with one reference point. The camera position is changing, and if the camera's position references the location of an object, we would get locations relative to the changing position of the camera in timespan.

To globalize the view's orientation, measures such as top-down view of a map or calculating the camera's rotation from the global center. While the global locations are crucial for measuring the distance, other point-views are also crucial for other purposes. There are three essential coordinate systems to know when working in a 3D environment:

1. World coordinates(global): World coordinates(global): The coordinate system that starts at the center of the world; a house in our example. Its distance then decides the center of an object in this coordinate system to the center of the world.

camera-view coordinates: The coordinates from the camera's views. The center of this coordinate system is the position of the camera. Its distance to the camera defines the center of the object in this world.

3. Local view: The center of the local view is the object itself.

The center of all these views is (0,0,0). We described above that the world coordinate system allows us to measure the distance between objects in a world map. The camera view is useful if a robot is expected to navigate an environment and describe spatial relations between objects such as "next to", "above". The local view could tell about the size of an object. In particular, the (x,y,z) from a local point of view tell about how far the object stretches from its center where the center is (0,0,0). The local view can be referred to as "radius".

matterport 3D provides the views described above. Next, We discuss the usage of the object's location in global coordinates and the local view in detail.

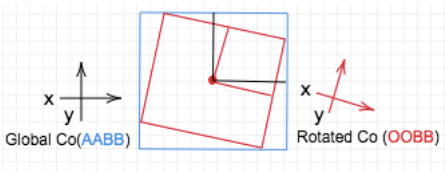


Figure 11: 2D AABB represented in blue square with its axis aligning with the world view of axis. 2D OOBB represented in red square has its axis rotated from the global view

Object's locations, geometric data In a 3D environment, objects could be represented by bounding boxes(covers) referred to as "Axis-Aligned Bounding Box" (AABB) and "Object-Oriented Bounding Box" (OOBB).

The AABB and its center are aligned with the view of the world coordinates, while the OOBB is oriented with the box and more likely to be rotated from the global view. In figure 11 we see a demonstration of the two boxes in 2d squares. The red square represents the 'Object-Oriented Bounding Box' (OOBB), where its (x,y) radius connecting the center to the sides is colored in red. The blue square represents the 'Axis Aligned Bounding box' (AABB), where its (x,y) radius connecting the center to the sides is colored in blue. The notable difference between the two boxes is the direction of their radius. The radius of the AABB in black is aligned with the direction of the global coordinates on the right part in 11. The radius of OOBB, on the other hand, has its coordinate direction rotated from the global coordinates illustrated in the red coordinates on the right side of the graph.

The difference in the rotation of the coordinates is important for determining the correct calculation for estimating the locations of the box's sides. In order to obtain where the sides/corners of the box are located in the global coordinates, we would estimate how far the radius-es stretch from the center and in what direction.

The estimation for the AABB sides is straightforward since the AABB's radius stretches in the same direction as the global coordinates. For example, subtracting the length of the AABB radius on the y-axis from the center would give us a location point of the lower horizontal line of the blue box. Adding the (y) length from the center point would give a location point on the upper side of the blue box. Hence- The center and the AABB radius are both globally aligned(pointing in the same direction) so adding or subtracting them would give the correct globally defined position of the AABB side.

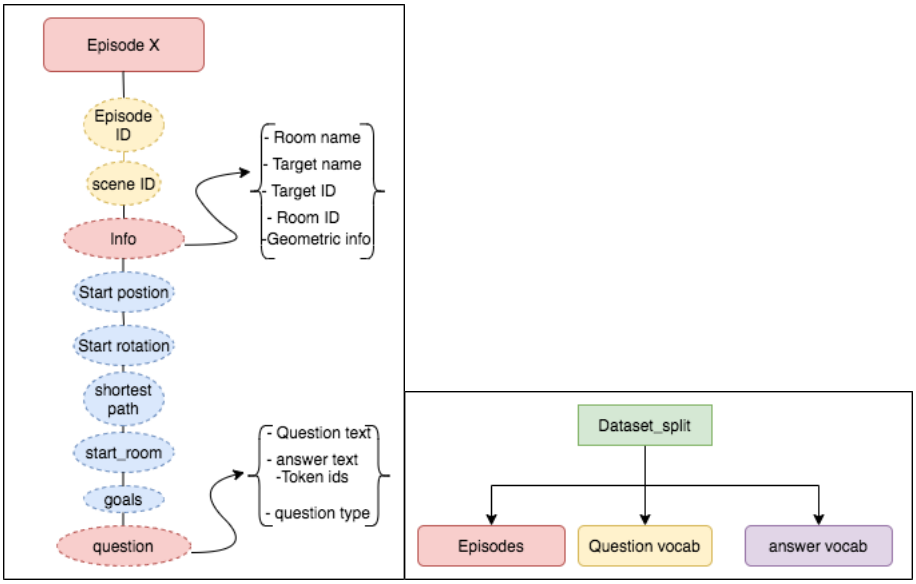
Even though the centers of OOBB and AABB are positioned in the same location, the direction of their axis is different. For estimating the sides of the OOBB from the center, one should consider the rotation of the coordinates. Adding or subtracting the OOBB (x,y) radius from the center, as done for the AABB in the previous example, would likely not end in a correct position on any OOBB sides since the directions of the coordinates(slope) are different. Estimating a globally defined point on any OOBB sides would require calculating the rotation or the slop of the radius from the center (the direction).

Using the AABB of an object is suitable for a direct allocation of the geometric locations of objects. At the same time, OOBB could give a more accurate estimation of the size of an object. With AABB, the borders of an object could be found with straightforward calculation, which makes it less complex. For example, allocating two objects and calculating the distance between any side would only require knowing their radius and centers. The OOBB, on the other hand, is more enclosed in the object since

it's more oriented in the object's local view. The enclosing of the OOBB on the object makes the space between the box sides and the object's edges much smaller compared to the space that could be found in an object defined in an AABB. The radius of the OOBB can provide a more accurate measurement of the object's volume. For size estimation, the rotation is unnecessary because calculating the volume does not require knowing any coordinate positions in a global map.

EQA (Task Dataset)

Our method for generating questions relies on imitating the structure of the EQA-mp3d dataset. In this section, we give a review of the EQA-mp3d structure and content. The EQA-mp3d also provides a primary material source. The relevant material consists of the navigational data required for constructing a trainable episode. In addition, an episode includes essential information about the target object in a question, such as its ID and room ID. This object's info is important as it would make it possible to acess more metadata about the spcefic object from the annotation.



(a) (b)

Figure 12: (A) represents the structure of one QA episode. The arrow and curly brackets branching from "info" and "questions" show the content of each of these two categories. (B) represents the most top layer of a split("train" or "val")

Structure .

Figure 12 (B) shows the top structure of the val and test. *Episodes* contains all QA episodes in a data-set split. *Question vocab* and *answer vocab* contain the same elements as dictionary keys. The elements are: [word list,stoi,itos,num vocab,pad token].

"Question vocab" and "answer vocab" in the "train" and "Val" are identical to each other. When using each split of the dataset, the answer-tokens that are considered are the ones contained within the episodes instead of the word lists mentioned above.

Each question-answer is an episode that consists of multiple layer information. The structure of an episode is as seen in 12 (A). We describe the elements of an episode in the following:

House ID: The house ID given by the house ids in MatterPort3D.
Episode ID: The episode index in the range of the split's length.
Info: This element contains all the information about the object and room in a question. The inner layers of "info" include the following:

Information about the target object is the first layer within "info" and its elements are listed below:

centroid: The center of the object's Axis-aligned bounding box(AABB). The center coordinates are in 3D on the (x,y,z) axis and defined with the global view.

radi: This is the radius of the Axis-Aligned-bounding box of the object. The AABB radius is in 3D on the (x,y,z) axis.

level: The level-floor number in the house where the object is located.

room-id, *obj Id*,*room name*, *object name name*: Room ID, room name and object ID as given by semantic annotation in Matterport3D. Many of the objects are re-named, mostly names in hyponymes changed to hypernym category such as: round-sofa, l-shaped sofa changed to their hypernym category "sofa". The information about the room (*room-id*,and *room name*) are found in the second layer within "info"

The elements that are marked in blue in figure 12 are **navigation data** used for training the navigator:

Start position: The start positions are all unique. For each unique question in the data set, there is fifteen different starting position.

Rotations: These are the rotations that the agent has to do while navigating. It stands as supplementary information for the shortest path.

Shortest Path: The shortest path is the data used for Imitation Learning by training the robot to follow the steps in the path with short rewards. The path consists of steps in frequencies that mark the shortest way to reach the question's object.

Goals: Goals are the destinations that the agent should reach in navigation. The goals stand for the viewpoint where the robot can see the target object. Each viewpoint consists of a geometric position and the rotation toward the target object respective to the position.

Task One- Question Generation (Final review required)

Overview Extending Dataset

Extending the dataset is in a form of asking more questions about the objects found in EQA-MP3D. The new generated questions are size and spatial questions, and they are constructed in an episode form. The episodes, as mentioned in previous sections, are executable functions when inserted in an environment they yield an answer. In order to make a question-answer an executable episode-function, we copy the navigational & object data of every episode in EQA-MP3D and generate new question for each unit of the nav data.

The questions' strings are automatically generated in templates. For each template of a question-type, there are two variants of the template. One variant with a room specified in the question and the other without specified room. The one with room specified are for generating a question of an EQA-MP3d episode of color_room type, such that a question like "what color is the table in the room?" would have a corresponding size question "how big is the table in the livning room?". Templates with unspecified room are for generating new question of the episodes of "color" type in EQA-MP3D, such as "what color is the table?", and the new question would be "how big is the table?".

The templates for size questions are as the following:

size_obj : 'how big <AUX> the <OBJ> ?
size_room:'how big <AUX> the <OBJ> in the <ROOM>?'

Size question-episodes do not include the insertion of new object's info. Size question relies on turning a color question into a size one without the addition of other meta-data into the newly transformed episode. The object's info such as, center and radius of its AABB are taken as they are from an EQA-MP3D episode into a new episode.

The spatial questions are of three relational types. A spatial question can either ask if there is an object "next to", "on" or "close to" other object. For each relational type there two variants of templates. The templates are the following:

'<AUX> there <ARTICLE> <OBJ1> close to the <OBJ> in the <ROOM>?'/
'<AUX> there <ARTICLE> <OBJ1> next to the <OBJ> in the <ROOM>?'
'<AUX> there <ARTICLE> <OBJ1> on the <OBJ> in the <ROOM>?
'<AUX> there <ARTICLE> <OBJ1> close to the <OBJ>?',
'<AUX> there <ARTICLE> <OBJ1> next to the <OBJ> ?',
'<AUX> there <ARTICLE> <OBJ1> on the <OBJ>?'

Spatial questions include the addition of a new object's info into an episode. The new object is the object used to question about a spatial relation with the already existent object in an EQA-MP3D episode. For example, a generated question as "is there a chair next to the table?" where "table" is an object in an EQA-MP3D episode and chair is a new object found within a relation to the table, and therefore, it has its data inserted in the new episode.

work organization The work structure of generating question consist of two components. The first component is a parser that does data extraction and processing, and has the functionality of acting as a calculator for measuring spatial relations. The second component is the question-answer episode generator.

This section begins with describing the parser, then the question-generator and ends with presenting results. Description of the parser is split in two parts; The first part shows the process of extracting semantic annotations, and the second part views how the spatial relations are estimated. The section, thereafter, moves to describing the workflow of the generator, and the criteria in which the episodes of each question type is constructed. Finally the section ends with results showing the number of questions generated for each question type, and the answer distribution of the questions in the extended data-set. The results section ends with a discussion around the semantics of the questions asked. The discussion raises questions about the precision of the conveyed meaning in the question-answers, and how might the meaning be perceived.

Data parser

The data parser is initially used to parse the semantic annotations and process geometric data and save it for the use of generating answers. The second functionality of the parser is to act as spatial relation estimator used simultaneously while generating questions. These two functionalities are are divided in two components. We begin with describing the first component, annotation extractor, which includes two different experiments/ways of extraction. The description of the second component, spatial estimator, includes the measurements in which spatial relations were determined for pairs of objects.

The first experiment for extracting semantic annotation is through extraction from 'house files' of the MP3D data-set. The second experiment uses Habitat Simulator and sensors. The annotations extracted from the sensors in the Habitat's simulated environments provide additional computed information and slightly different raw data from MP3D annotations; In particular, some object names are different, but the rest of information, such as object ids and location-centers, is consistent with the annotation of the MP3D.

In the existing generated question-answers data-set we use the data extracted through the Habitat semantic sensors. The main reason for choosing Habitat's simulator's sensors is because they provide a computed geometric information of the object's Axis Oriented Bounding Box. The MatterPort files include only the radius of objects' OOBBs. An additional important reason for this choice of extraction is that some of objects names output-ed by the sensors are aligning with the names found in the original EQA-mp3d data-set. For example, object names in MP3D such as l-shaped sofa and rounded-sofa are transformed, in Habitat's sensor, into their Hypernym category 'sofa'. Choosing object names that are aligning with names found EQA-mp3d, is helpful for having the overall data consistent with each other when we emerge our generated questions with EQA-mp3d.

Direct annotation extraction from MP3D files

We extract two types of raw information from each object's line of annotation found in the house files in Matter-port. Lines mentioned erlier: [px py pz a0x a0y a0z a1x a1y a1z r0 r1 r2 0 0 0 0 0 0 0 0 0]

First we take the obj and room indexes (ids). Second is the [px py pz] and where we categorize it as the center of the OOBB of the box. Third is the [r0 r1 r2] (radius of the OOBB).

we structure the data in a form that the annotation of a house begins with the first level in it, followed by the rooms and objects in each room as: house 1 [level1:room1[bedroom]:(obj1:bed,obj2:..),room2:(obj..),, level2:.....]. The extracted data is then saved into a file.

Annotation extraction using Habitat Simulator

Our final choice for extracting semantic annotations is Habitat's simulator. Our annotation's parser of the houses uses the sensors with configuration provided by the habitat platform [13]. The configurations include the settings such as the scene, the height and width of the sensors, and the types of sensors to include. color sensor, semantic sensor and depth sensors are used.

Once we simulate the environment, the sensors output the annotations of a house(scene) as an object. We iterate through the object to obtain information about the levels, rooms, and the objects in the rooms. We freeze the simulator once the annotations' object of one environment is outputted, then repeat the process for the other environments.

In addition to the semantic annotation, we extract the center, radius of each of the AABB and OOBB of the objects. The radius size of the AABB is computed within the simulator and is streamed out with objects annotations. The radius of the OOBB is used for finding the objects' sizes. The center and radius of the AABB are processed into an information useful for a method of estimating spatial relations among objects.

Calculating the Min and Max values of AABB corners The center and radius of the AABB are used to find position points on the edges of the object. Knowing the borders of an object's AABB provide a way to determine a spacial relation between objects given a criteria of distances between the corners of two objects . The first Information we obtain from the center and radius of the object is two corners of the AABB. Figure 13 illustrates an AABB in 3D as the AABBs we get with the objects annotation. Each of R_x, R_y, R_z is 1d radius on the x,y,z axis, where the x is the length, y is the width, and z is the height. The radius in 3d would be the line/vector from the center (C) to either min or mix.

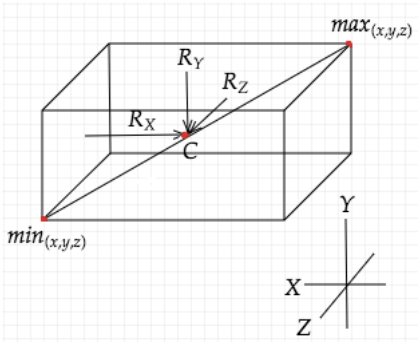


Figure 13: Min and Max of an Axis Orients Bounding Box. R_x is the radius on the x-axis, R_y is the radius on the y-axis, R_z is the radius on the z axis. The line from C to Max is the radius in 3D which is also equal to the line from C to Min. The line from Min to Max is the 3D diameter. Radius is half of the diameter.

The first calculation is finding the 'min' and 'max' points of a bounding box given an object's center and its 3D radius. The min represents the corner point in the minus direction from the center in all the axis, and max is the corner on the positive direction from the center in all axis. The AABB radius extracted from the habitat simulator is in diameter form as the line from min to max. The radius would be half the extent of the diameter so we get the 3D radius by simply dividing the the diameter by two.

[13] https://aihabitat.org/docs/habitat-lab/habitat-sim-demo.html#scene-semantic-annotations.

$Radius = D_(x, y, z)/2$.

The 'min' is the position point stretched from the center by the length of the radius on the negative direction of all the axis, and 'max' is the point on the positive direction of the center, at the end of the radius length. Calculating the min and max, is therefore, done by subtracting or adding the 3D radius from the 3D center. The center is a point at one ends of radius and the radius is a vector, if we add the length of the vector to the center point we get the point at the end of it's length "Max", and if we subtract the radius length from the center point we get point at the end of the radius length in the minus direction which is the "Min". Below, C denotes the center point and $\vec{R}$ denotes the radius as a vector.

$$Min\ point\ =\ C\ -\ \vec{R}\ =\ (x_1 - x_2, y_1 - y_2, z_1 - z_2)$$

$$Max\ point\ =\ C\ +\ \vec{R}\ =\ (x_1 + x_2, y_1 + y_2, z_1 + z_2)$$

The min and max as corners of the box could be used to estimate distances between objects. For example, in 3D game design they are often used for collision detection. From the Min and Max one could the values of all the other corners, as the values of the other corners range between the [Min,Max] on all the axis. In the following sections we describe in detail how the Min and Max used in a technique for finding spatial relations between objects.

For every object in the annotation we find the Min and Max of its AABB and extract the radius of its OOBB. The radius of OOBB is given with the data extracted in the simulator. We use the OOBB radius for calculating the sizes of the objects. We consider the size as the volume of the box which is the length multiplied with the height and width. In our case the length is the x delimiter, width is the y delimiter and height is z delimiter. The calculated volume of a box is $X\ x\ Y\ x\ Z$.

Storing the annotations We structure the annotations and save them in a file. The structure of the data consist of a dictionary storing the data in a hierarchical way. At the top part is the house id, then rooms in the house, then the objects in the house. Each object stored by id, contains the Min and max value of its AABB, radius of the OOBB, its name & ID , room name, and the level id where the room is located. Structuring and processing the data and having it stored in files allows to access all the objects in a room through the scene id and room id, which accelerates the question generating process.

We store the calculated volume of each object in all the houses and store it in a second file. The volumes of objects are stored by their object category. In the volumes file, we find the volumes of all the objects in all of the houses stored in a dictionary, each key represents a category such as 'sofa' with values of the volumes of this object type. The point here is to obtain data on the sizes of each object type. We use this information for finding ground truth answers about for the size questions. Defining answers for size questions is elaborated in detail in further sections.

Spatial relation estimator

The functionality of the estimator is to take a group of objects and pair them according to spatial relations. We give the estimator all the objects in a room to find pairs that are 'on', 'next to' or 'close' to each other. If the mentioned relations are founded between two objects, the pairs are organized in a dictionary, one key for each spatial relation. This information is used for generating positive spatial questions.

The first measure for determining the mentioned spatial relations is by calculating the distance between the corners of AABBs of two objects. In the processed annotations the objects are initially represented by two corners, 'min' and 'max', as seen previous section. The other corner points of the box can be

extrapolated from the 'min' and 'max', as certain dimensions overlap.

Extrapolating AABB corners from Min & Max Points In figure 14 illustrates the eight corners, the view point of the cube is rotated to the right for the sake of viewing all the points in the cube. If we move our point of view directly in front of the cube as if we are facing the square GHED, the points A and H would seem to be lying on a straight line. Lying on the same straight line, for example, means the point A and H are located on the same points in the x-axis, and so one for the other parallel points.

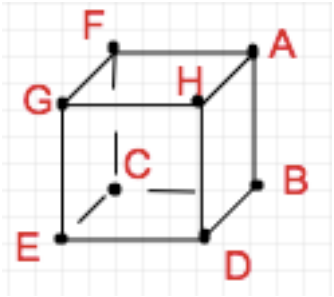


Figure 14: Corners of Axis-Aligned Bounding Box. The box viewed here from upward rotated to the right position in relation to the front of the box. The correct global viewpoint of the box would be by imagining our viewpoint straight facing the square GHED, where A&H would be on a straight line(simiar points on the x-axis), and as such for all other parallel points. From the illustrated corners A would be the 'Max' and 'E' would be the min

We get the rest of points from the Min and Max of an AABB for the reason that AABB's are not rotated and aligning with the global view. To express it better, we image the global point of view of the AABBS as a view facing a group of adjusted and not rotated boxes. When the axis are aligned, the values of the corners would overlap where the 3D values on the (x,y,z) would be either the 'Max' or 'Min' in each dimension. For our example in figure 14, the point A represents the "Max" corner point, and the point E represents the "Min" corner. We can extrapolate, from the 'Min' & 'Max', the six other corners as such:
$A = (x_{max}, Y_{max}, Z_{max}), F = (x_{min}, Y_{max}, Z_{max}), H = (x_{max}, Y_{min}, Z_{max})$,
$B = (x_{max}, Y_{max}, Z_{min}), D = (x_{max}, Y_{min}, Z_{min}), C = (x_{min}, Y_{max}, Z_{min})$,
$G = (x_{min}, Y_{min}, Z_{max}), E = (x_{min}, Y_{min}, Z_{min})$

Measuring the distance between AABB corners The first criteria for determining a potential pair with spatial relation is the distance between their corners. The Euclidean distance between two corner points; denoted as the distance between p and q, where P is one corner of an object and q is corner of other. n denotes an Euclidean space, q_i&p_i are the Euclidean vectors of the corners where the base i stand for the dimensions of the vector. The formula can be described as the square root of the sum of the square of the subtractions of q and p at every i-dimension.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

The corners, depending on the type of relation we want to extract, can be represented as points in 1d, 2d, or 3d. For example, to filter pair of objects where one is 'on' the other, we would check how close they on 3D axis but then we want to know the distance on z-axis(the height) in particular; Therefore we would

calculate the euclidean distance for the corners in 1D as such: $\sqrt{(z_1 - z_2)^2}$. Knowing the distance on the x and y axis would be an indicative of corners next to each other in this case we measure the distance of 2d corners and such:$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. We would represent the corners in 3D if we would want to measure how close two corners to each other in general, not on a specified axis. The Euclidean distance between two 3D points would be as such: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$

Classifying relations between the sidelines of AABBs If a pair of objects are close in distance, we distinguish the possible spatial relations they have depending on the overlap of their sides of their AABBs. For example, close boxes that might be 'on' each other, the vertical line (Z(max),Z(min)) of one object should be contained within the vertical line of the other, otherwise, this would mean the one is inside the other instead of being on it. This measuring criteria is intended to check for possible spatial relations among the pairs with close distance. This technique is used in different instances while determining if an object is "next to" or "on"

Figure 15: Text to be added

The calculation if one side is contained within the other relies on defined criteria. The calculation is done in a specific function and can take as input corner points on one axis or more. In the drawing 15 'A' represents two lines on the 'X' axis where the top part is not contained within the other, an in 'B' the top is contained within the lower line. In this example we determine the 'contain' relation by taking the 'min' represented by the the orange dot and the 'max' doted in red.

If the min of the upper line is greater than the min of the lower line and the max of the upper line is less than the max of the lower line then the upper line is contained within the lower. Hence the 'min' of the upper line is greater than the 'min' of the lower line because it's more to the right in the positive direction of the 'x' axis. If the previous condition is not satisfied, then the lines are otherwise not contained.

The operations above are done over different axis for every relation. Below we specify how each of the 'on', 'next to', and 'close' relation is determined between the objects.

On First step is choosing pairs of objects that are closest to each other vertically(on the Z axis). The distance should be less than a 0.5 millimeter thresh hold. The Euclidean distance here is calculated between the 1D points on the Z axis only. Two corner values from each box and any corner match the distance required are picked out.

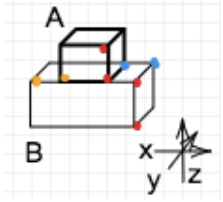


Figure 16: Text to be added

The second step consist of a group of conditions that the pair of objects need to meet in order to be

considered on each other. The first condition is that the vertical sides, the line from Zmin to Zmax, of the boxes are not contained within each other. The Zmin-Zmax lines of every object box are the lines between the two red dots in box A ans B in the illustration 16. Otherwise if the lines on the Z axis are contained it would mean one object is inside the other.

The second condition is that the horizontal line of one of the boxes is contained within each other.The horizontal lines in the illustration are from the Orange to the red points in each box. The lines on the y axis from the red to the blue points should also be contained.

The final step is deciding which object is on the top of the other. The pair of objects are passed to a function that see which Zmin-Zmax has greater value. The object on the top should be in the upward positive direction.

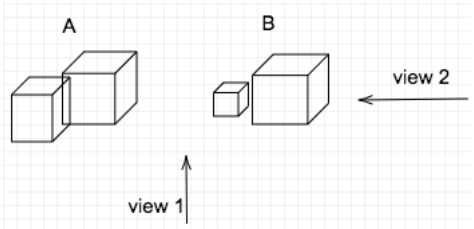


Figure 17: To be replaced by new figure

next to A pair of objects next to each other have to have a distance not greater than 0.1 meters on the X and Y axis. The distance here is calculated for 2D corner points, this means the distance is calculated for four corners (min and max front and back).

The pairs should not have contained sides in neither the x nor the y axis. In this condition a pair of objects next to each other would like illustration A in 17. This is might a bit different from what we consider next to each other as humans. We might imagine a typical next to pair as illustration B seen from view 1.

However, the choice of having 'next to' pairs not contained with each other is due to considerations of the view point. From view 1 the pair(B) seem next to each other but from view 2 they would not. In pair(B) from view 2, one object would be behind the other and likely hidden. So if view 1 is the global view and we pair the objects as in (B), seen as next to in view 1, the robot might enter the scene from view 2 and it would be wrong to refer to the pair as next to each other. However, if the 'next to' pairs are assigned as in illustration (A), the pair would be still visible in whichever view, and positioned proper enough to be referred to as next to each other.

Finally the pair must have their lines contained on the Z axis. Otherwise, the two objects might satisfy the first condition on the (x,y) but be distant on the z axis, such as one object in the ceiling and the other on the floor.

close to A pair close to each other are a pair who has any of their 3D corners close to each other within a max distance of 0.2 meter. No other conditions required for the pairing of objects close to each other. It can be a close object on, above, below or next to.

Second Module - question generation (To be reviewed)

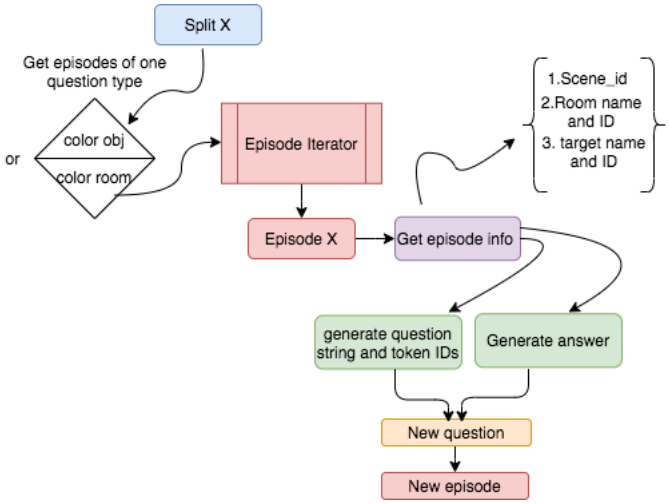


Figure 18: Split generator

Our question generator generates questions of two types, size and spatial. In order to run the generator, the arguments required are the type of question, path to the val and train splits.

The question generator generates questions of one type at the time. Questions with the string ”room” in is considered a different type from a question that refers to objects without a room. For example, the question ”How big is the table?” has the type ”Size”, and the question ”How big the table in the living-room?” is of type ”Size room”. In order to generate size questions with and without reference to a room, therefore, requires ruining the code, one separate time for each type.

The split generator is the core component in the code. It takes a split either train/val and a question-type as arguments. The functionality of the split generator is to turn an EQA v1 split of episodes into a new split of new episodes. In figure 18 we see an illustration of the workflow of the split generator. The five general steps is filtering(uncolored rotated square), iterator(red rectangle), episode parser (pink rectangle), QA generator (the green rectangles), and episode wrapper (the bottom yellow rectangle- inputs QA and outputs episode)

The filter returns a set of question-episodes of one type only. The returned set of questions of a type is dependent on the question type given to generate. For example, if the input is to generate questions of ’size-room’, ’how big is the sofa in the room’, we take only the questions of ”color-room” type.

The filtered set is passed to an iterator. Each iteration passes one episode from EQA-v1 to a parser. The parser function in the iterator extract information, such as the object name and id, scene ID and room ID, from the EQA-V1 episode.

The parsed information is passed to a QA generator. The QA generator is better described as a group functions of the split-generator that are conditioned differently dependent on the question type. The answer generation function is, however, a different function for each question type.

The general idea for generating QA of any type relies on two straightforward steps. First, generating a ground-truth answer for the given question, which is the most important stage in the generation process as it requires calculating values from the data in the houses. Second step generating a question string and token ID’s.

The final step consists of inserting the new question with the corresponding geometric information, and structuring them into an executable function. We call a QA sample an episode when the section of the episode seen in figure 12 are filled with the new QA and the other the corresponding information.

Our question generation can be described as generating one question for every “shortest path” there is. The idea of transforming an episode from EQA into a new one is based on using the starting position and the ’shortest path’ found in them. Having more questions for each shortest path is equivalent to having multiple questions about the same scene.

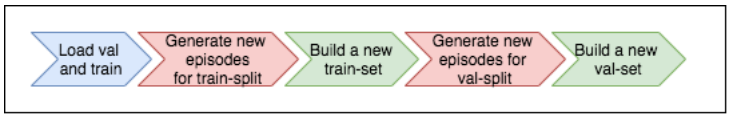


Figure 19: Split generator (The top part of the code)

The top most part of the code (the data-set generator), illustrated in figure19 , passes the train and val to the split generator at different time-stamps. The reason for generating the two splits in two different stages is to keep track of the number of question-answers generated for each split. Emerging the two splits and splitting them randomly at the end might create and imbalance between the answers in each split. The current code controls the distribution of answers in the train and val sets. Otherwise, leaving the type of answers uncontrolled would leave a bias towards one answer over the other.

Once a split of episodes is generated it’s passed to loader function. The loader function inserts the answer and question vocab to finalize the data-set in the form seen in figure 2.2

In the coming two sub-section we describe in detail how the QA generators for the size and spatial questions work.

size-questions

Size questions are generated through three steps. The first step is generating a ground truth answer about the size of the the target-object found in EQA-V1 episode. There three possible answers are Big, Small, and Medium. The second step is generating a question string and token ids. The final step consist of filling the question-answer in an episode form, with shortest path and the rest of object’s info from the original EQA-v1 episode, as described in the previous section.

Size answer

The size answer is generated in a function referred to as ”GetsizeAnswer”. This function takes as an argument the target-object’s name and the size of its box and returns an answer about its size. The function calculates the volume of the target object in a similar way as the rest of the sizes of the objects. Volume of the OOBB = W x L x H. The next step in the function is to compare the size of the target to the sizes of the objects of its type.

The relative size is determined by its deviation from the standard of its type. As mentioned earlier the sizes of all objects are stored by type in a file. We pick the volumes of the object’s type and calculate the mean size and the standard deviation of all the the sizes from the mean. The standard deviation denoted below:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

The answer is 'small' if the objects' size is smaller the mean size of its type minus the standard deviation, 'big' if the size is larger the median + the standard deviation, and middle if the size of the object is within the standard deviation added and subtracted from median.

We control the answers' distribution. We observe that a majority of objects have a medium size given the standard of their type. In order to avoid bias towards the 'medium' answer, we restrict the number of QA with medium answer. We keep track of how many QA with medium answers has been generated and when the number reaches a limit we generate None QA that are later filtered out. The limit varies depending on the question type and the split (train or val), and is based on our observation of the answer distribution in the splits. Note that we refer to 'question type' in this example if either the question to be generated is size question with string 'room' such as "color-room" or without.

spatial-questions

Generating spatial question takes more complex steps and longer time than generating size questions. Generating a spatial QA requires a coordination with the spatial relation extractor. In addition, spatial questions include the addition of an extra object to the question string, and the insertion of the new object's information into the QA episode.

Searching for spatial relations of the target object in an EQA episode is the first step taken. We pass the scene an room id to the 'relation' extractor to obtain pairs of objects, within a room, with a spatial relation between them. The relation extractor returns three types of relational : next, on, or close, if existent within a room. Else it returns a category with empty values.

The decision of generating a question of one of the mentioned relational categories is dependent on the existent of an object with a spatial relation to the target object. The process of executing a generation command of a question of a spatial type is illustrated in figure 20. If there is an object 'on' the target or a target is on another object, we generate one questions, and similar case if there is an object next to the target object. If there is no 'on' or 'next' relation or either of them is non existent, the criteria for checking if there is a 'close' object is satisfied. If none of the conditions are satisfied a QA with no 'answer' of a random spatial type is generated.

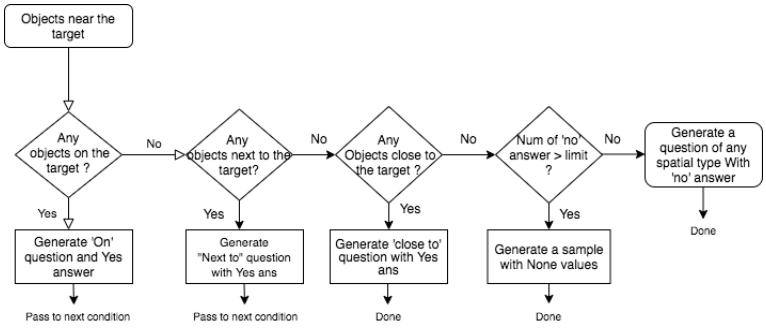


Figure 20: Decision tree for generating different types of spatial questions

A QA with positive spatial answer has a 'yes' answer, and 'no' if a relation is non existent. The decision tree as seen 20 leverages positive QA for the reason that we observe that the no-relation instances

outnumber the positive ones. The final condition, we even control the number of QA with 'no' answer by generating a None QA if the number of generated QA with no answer reaches a limit. The QAs' with None values are later filtered out.

Within this decision structure, for each 'shortest path' in an EQA episode, there is a possibility for generating from one to two spatial questions of different spatial type.

The process of generating a spatial question includes the addition of information about two objects. An episode/question generator, a group of functions, adjust itself to a spatial question generation if certain arguments are given to it. These arguments are seen in the input section in the illustration in fig 21. Such as potential object type, spatial question type, and all object in a room

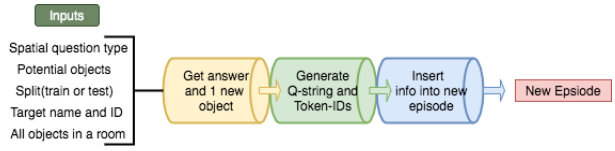


Figure 21: Structure of spatial questions generator

All the inputs seen in 21 are required to generate an answer from a function called "GetSpatialAnswer". All objects in a room are needed for generating "no" answer. In case of generating a "no" answer the "GetSpatialAnswer" function picks a random object fro the houses that is not in the room. The reason of excluding objects in the room from the selection of a random object for a negative QA is to help us in the validation process, such as we would know if the robot answer 'yes' to a QA with 'no' as ground truth that it's due to bias rather than he robot recognizing the object in the scene.

A selected object of potential objects and the type of spatial question are required arguments for generating spatial question string and token ids.

The last part in blue is conditioned by the type of answer, if it's 'yes' or 'no'. If the answer is yes, geometric information of the target object's pair is passed to it to insert it in the episode. If the answer is 'no', no additional information is added to the episode beside the information of the target object found at the end of the shortest path.

Results

Total number of generated questions

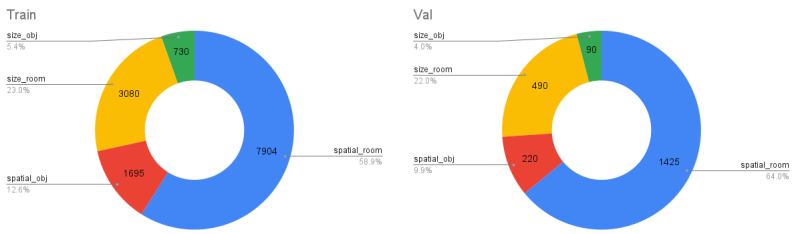


Figure 22

We generate a total of 13 409 question for train and 2 335 questions for validation. In figure 22 questions of size_room and spatial_room refer to questions that contains a reference to a room, such as 'How big is the bed in the bedroom?'. Questions of spatial_obj or size_obj type are questions with a reference to

object only, such as ”Is there a chair next to the table?”.

Answers distribution

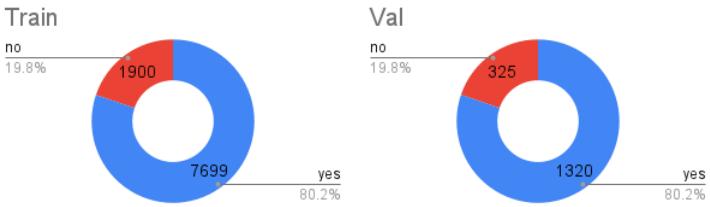


Figure 23

Answers distribution of spatial questions The majority of answers for the spatial questions are positive ”yes” as seen in figure 23. This imbalanced distribution is an intended outcome. The motivation behind this intention is based on an idea, formulated by Regier (1996), of learning of positive samples only. The argument behind this approach to learning is inspired from a cognitive theory of human’s first acquisition of language. The theory is based on the premise that humans tend to learn spatial relations from positive evidence instead of non existent instances.

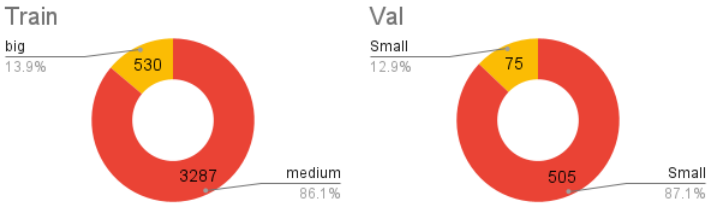


Figure 24

Answers distribution of size questions The outcomes of generating size questions resulted with zero samples of ”small” answer and a majority of ’medium’ answer as seen in figure 24. In the QA generator, we intended to limit the question-answers with ”Medium” answer based on an observation of their dominance. However, limiting the ’medium’ answers more than the presented numbers would have resulted in a very few question-answers of size type. An insignificant proportion of size questions was insufficient for training the model . We decided to keep the size questions with their imbalanced distribution, despite knowing that this linguistic bias might hinder the learning outcomes.

Our question-answer generation is very dependent on the objects and the shortest paths found in EQA-V1 data-set. The geometric information such as starting positions and shortest paths are required material for extracting scenes training and testing the VQA model. To be only dependent on the objects found in the EQA restricts our choice over the QA that we could generate. The most prominent limitation we see in the generated questions is the inability to control the answer distribution for size questions. None of the target object’s found in EQA V1 turned to be of a small size, based on the criteria we establish for describing an object’s size.

Discussion - (two paragraphs to be added)

The description of sizes is paradoxical. A well known paradox in philosophy, the sorites paradox, uses the description of a ”pile of sand” to display the dilemma of describing the size of an entity. The paradox,

described by Fisher (2000),is stated as such: is a grain of sand a pile of sand? No. does adding one or 5 or 20 grains make a pile of the sand?, the answer is still no. We can make inference that adding grains of sand does not make a pile. However, this inference is inaccurate because we might conclude that adding 10 million grains does not make a pile of sand, which is false. This is the reason why it is a paradox. A paradox is when the premises entail a logical inference but a conclusion we draw is false (Fisher (2000), Sainsbury (2009)).

Sorties paradox can be expressed more clearly in prepositional logic, *modus ponens*. *Modes ponent* is a form deductive argument for making an inference. Its rule is based on conditionality of the truthfulness of a statement, if P is true then Q is true. The inference we make from making a pile is that, if "one grain of sand is no pile" is true, then "2 grains of sand is no pile" is also true. We denote the predicate 'no pile' as *P* and a grain of sand as *a* and the number of grains as *n*; one grain of sand makes no pile is the denoted as such P_{a_1}, "two grains of sand is no pile", P_{a_2} , and our conclusion that adding any number of sands is no pile would be $P_{a_{n+}}$. The process in which we made the inference ($P_{a_{n+1}}$) adding more grains makes no pile is represented as such:

$$P_{a_1} \rightarrow P_{a_2} \rightarrow P_{a_3} \rightarrow P_{a_4} .... \rightarrow P_{a_{n+1}}$$

if one grain of sand is no pile then two grains of sand is no pile, if two grains is no pile then three grains no pile, if three grains is no pile then four grains is no pile, then we make the inference that adding any number of grains is no pile. The conclusion is any number of grains do not make a pile.

Paradoxical predicates are vague form of knowledge. The sorties paradox applies to all the adjectives and forms of expressions that lack a precise form of logical construct; Small, big, bold and expensive are examples of predicates equally paradoxical as the "pile" of sand Kennedy (2007). From an epistemic approach, some philosophers disregard these forms of language expressions as valid arguments/knowledge about the world. In epistemology the questions in interest are how do we know what we know?, in what way was a certain knowledge obtained?. For example, how do we know that a pile of sand is a pile of sand or that a chair is a big chair not a small one; note that the knowledge of 'big chair' or 'pile of sand' is here the 'meaning' of them, what does the symbol "big chair" mean.Williamson (2002) notes that vague predicts distinguished from other predicates by the absence of a precise/clear boundary lines that separates their meaning from its negation, for example, the boundary line of when a pile is a pile and when it's not, or the boundary between pretty and ugly, big and small. On the other hand The distinction between an animal and a tree has a clear borderline, one is inmate and the other is not.

Paradoxical expressions are rejected, in the epistemic view, based on the unfounded precise reasoning that logically defines them. They are considered vague primarily for the absence of higher meaning of measure. The induction to explain what a 'pile' means (in terms of properties of grains) proved to be paradoxical in classic logic, as seen in a previous example. The missing boundary of vagueness in classic logic is the distinguishing line of their truth-value, the one that separates True from false.

Williamson (2002) quotes J.L Austin (Austin & Warnock (1962)) on an argument regarding the usage of expressions such as 'accurate','precise', where J.L Austin disputes ,'If I measure a banana with a ruler, I may find it to be precisely 5 5/8 inches long. If I measure my ruler with bananas, I may find it to be exactly six bananas long, though I couldn't claim any great precision for my method of measurement'. One might say that our measurement of the banana is very precise, but to measure the truthful validity of our measurement is as accurate as measuring it with bananas. In the discussion found in (Williamson (2002)), vague expressions are seen in the epistemic view as a type of ignorance.

It is important to stress that 'vague' refers to the representation of the entity (the description) not the nature of it. Nature is logical, the world either exists or does not exist, the 'pile' exists regardless the vagueness of our measurement of its existence. Williamson (2002) on Bertrand Russel's reference to

vagueness, in (Russell (1923)), notes that Russel considers the issue of vagueness as a problem of symbol representation. Quine (2011) refers to the distinction between the symbol representation and the symbol referred to in the world as intention and extension. Intention is the expression/representation/language we use to refer to an existent entity in the world, extension is the entity with its inherited property in the world as it is. Crescent/full moon are intentions, the extension is the same moon object as it is in the world. Quine (2011) elaborates on Russel's assertion to vagueness as an issue with the intentions we use to describe the extension. Williamson (2002) points out the Russel means representation also in extra-linguistic properties, such as the perceptual representation we have of the extension.

Sorties paradox reveal vague expressions of an ordinary language through a logical system of an ideal language. Williamson (2002) on Russel's assertion, in (Russell (1923)), that the existence of vague expressions indicates that language as a whole is vague. An ideal language must have a logical representational unit for every extension Quine (2011). Russell's then invalidates classic logic as suitable to express ordinary language (Ordinary language has vague, general,ambiguous expressions). In the debate whether logic could be used to explain vague expressions, Williamson (2002) cites Max Black (Black (1937)), responding to Russel's invalidation of logic in regard to vagueness, that Russel's claim is an evasion of responsibility to describe natural language in systematic way. Black goes on in explaining that the incoherence of vague expressions in logic stems from a trouble that occurs by trying to logically formulate an expression of degree of truthiness without having explicit degrees of truth. For Black, some expressions in ordinary language cannot be confined to absolute truthiness or falseness, they instead could be neither true or false. For example, we can say a person is not tall but somewhat tall, or that adding 3 grains don't make a pile but adding thousands of them could, so 'adding grains of sand makes a pile' is neither false or true but partially true.

Supervaluation logic and solutions for expressing vagueness. Super valuation suggests a three valued truthiness. An expression can be either super-true, super-false or undefined. For a statement to be super-true or super-false a condition must apply to all of its valuations; valuation can be, for example, the *n* number of grains. A statement has "undefined" truth value if it has true valuations and false valuations. This three valued logic allows to distinguish precise expressions from 'borderline cases'(vague expressions). The obtaining of truth value of each valuation of the predicate is referred to as precisification. Precisification are referred to using quantifiers, existential quantifier $\exists$ to refer to a single valuation , and universal quantifiers $\forall$ all the valuations. The logical statement below denotes that there exists an *n* that's a border line for 'is no pile' where *n+1* is a pile. The statement draws a dividing line where at *n*th grain, every *n+1* after the line "is a pile" $\neg P(n_{+1})$

$$\exists n(P(n)\ \&\ \neg P(n_{+1})\ )$$

Despite the distinction of borderlines in super-valuations, a deficiency in meaning, referred to as highr-order vagueness, still appears. If the borderline *n*th , for example, is 8,000,000 grains of sand where all *n+1* "is a pile", but isn't 8,999,99 also a pile?. This a sort of paradox that found initially. Also the difference between very tall and tall would be still vague, the two predicates would fall after the dividing line with undefined difference.

Fuzzy logic and fuzzy sets proposes a solution of truth range that lies between [0,1]. This range of true values allows to map gradable matters, such as the difference between very short, tall, and very tall. Fuzzy sets maintains a feature of classical logic, that the absolute truth still lies in exclusion of a middle, [0,1],false or true. The range of turthness can be expressed in set theory using notions of intersection, union, complementation and subset which correspond to conjunction, disjunction and negation. In set theory we can denote that a 'definitely is a pile' is distinguished from 'is pile' in a way that 'definitely is a pile' is a subset of 'is pile' without claiming that a pile is a subset of definitely is a pile'. For example,

A is a set for 'definitely pile', and B is the set of 'is a pile', the relation can be expressed as $A \subset B$ (A is a proper subset of B), where all the elements in A are in B but B has more elements that could be for example, "is kind of a pile".

Sets can be formed in Boolean functions with conditions which makes it very flexible for usage in computer applications and programming languages. (Williamson (2002)) asserts that the development of fuzzy logic out of fuzzy sets allows for a replacement of [0,1] to any established range of number, for example we could say that *n* is a member of the 'is no pile' set (C) if its value is 0 or more and less than $10,000 : n \in CIf 0 \leq n < 10,00$.

Vague descriptions of sizes For size questions,the object is considered an element of a size set if its within the range assigned for the set. The range and borderlines of each set are determined in relation to a context. Establishing a range for a vague expression relative to a context is defined by (Raffman (1996) ,pp 181) as multi-range theory

"For any object O, vague predicate 'P', and total context TC: 'P' applies to O, relative to TC, just in case a competent speaker could judge O in TC and, were he to judge it in TC, he would apply 'P' to it."

The context in this project has been defined by the category of objects found in all the 3D environment; For example, the context of "big sofa" is all the sofas we find in MP3D. The standard deviation can be seen as the established boundaries taken from the contexts. The range of deviation of each object category is different and respective to its context, so that the size range of "big sofas" is different from the range of "big fireplace". The ranges stretch from the mean on the minus or plus as mentioned in section ..

The followed approach attempted to reduce vagueness following the semantic conventions for dealing with vagueness. However, even if semantics manage to find logical means to express vagueness as clear as possible, in an epistemic view this clarity is still in question. Semantically, perhaps the burden of clarifying vagueness in natural language is cleared out if boundaries are established with some systematic logical construct. Carnap (1955) refers to vagueness as *intentional vagueness* if no specifications (boundaries) made in the intention. For Carnap, if specifications are made in the intention and vagueness occur then vagueness here is extensional; They mean that vagueness is in the extension itself rather than only a matter of vague expression in the linguistic meaning (intention). An example that higher-order-vagueness remains despite the specification of boundaries, as mentioned before, the pile of sand would perceptually seem the same if it's one unit above or below the borderline of range.

However, for Russel, vagueness would probably remain as an issue of representation. Even if the intention(meaning) is constructed measurably in accordance to some logical relation to the extension, for Russel vagueness still remains in the other types of representation. As mentioned earlier, Russell views 'representation' as including also a photographic symbol such as the perceptual imagery we have of an object. We could assume an object to be of a big size relative to its context, and the 'intention' would seem matching the contextual size if we view the object from a certain point. But what if we go further away from the object ? The object would seem smaller and smaller from the point where we described it as big. The previous point sums the main argument for Russell's view on vagueness as an issue of representation. However, in this regard, he considers vagueness as natural phenomenon or attributed to what he refers to as the 'law of physics'. According to Williamson (2002) Russel refers to the law of physics as 'the appearances of a thing at different places are less and less differentiated as we get further away from the thing'(p.68).

Aldina Translating this discussion into a practical sense, the validity of our method of connecting the meaning "big sofa"(intention) to the extension it refers to is vague in two regards. First vagueness is the

higher-order vagueness described by the vagueness appearing in one percefication above or below the drawn borderline. For example two big objects appearing similar in size where one would be classified as medium size for being within the standard deviation, and other would be classified as big for being only one size unit above the deviation. The second existing vagueness is the description of sizes relative to the distance of the robot from the object. Does the description of 'big chair' match of what we believe as 'big chair' seen from the distance of the robot to the object (the extention)?. No we don't know if each size category is appearing in a similar approximation relative to similar viewpoints from the distances that the robot stops at.

Vague colors and spatial relations Color descriptions are the typical examples of vagueness

It is plausible that we measure the sizes of the objects relative to themselves in separation of the intention, then we connect what we believe

Vagueness is a natural phenomenon. Although Russell confines vagueness to representations, he regards it as a natural phenomenon, because he regards representation as a natural phenomenon. He traces vagueness to what he calls a law of physics: 'the appearances of a thing at different places are less and less differentiated as we get further away from the thing'.71The appearances of a thing are its publicly observable physical effects. As they spread outwards, information is lost; different close-up appearances give rise to the same distant appearance, so the latter is vague as a representation of the former. In the case of perception, Russell treats our sensations as appearances of their stimuli. Sensations caused by different stimuli are identical, or at least indiscriminable – Russell is not sure which, but hopes that quantum physics will eventually settle the matter. He holds this feature of our physiology responsible for the ineliminable vagueness of our knowledge, including its higher-order vagueness. Russell's physical explanation of vagueness again confuses it with unspecificity

to that "If cases for which no specification has been made can occur, the vagueness is intentional. If such cases do occur, it is also extensional" (249)

First counter-argument for dealing with vagueness is that a fraction of number, subtracted or added, to the size value of an object can categorize it

of the extension in this regard, can be seen as

The generation of size questions is based on range of sets in a context. The range we pick is

For size questions,the object is considered an element of a size set if its within the range assigned for the set.

The range of category sets is contextually defined.

6He does not require the truth set to be the interval [0, 1], but allows it to be any set on which a certain kind of abstract mathematical structure is defined

Moreover, fuzzy intersection, union and complementation correspond to continuum-valued conjunction, disjunction and negation. The development of fuzzy logic out of fuzzy set theory was soon initiated by Joseph Goguen.26

Moreover, fuzzy intersection, union and complementation correspond to continuum-valued conjunction, disjunction and negation. The development of fuzzy logic out of fuzzy set theory was soon initiated by

Joseph Goguen.26

0 is total false and one is total true, and in between is degrees of truthiness. notions of intersection, union, complementation and subset

According to Black, vague language runs into trouble by trying to express matters of degree without being explicit about degrees. 'He is tall' treats tallness as though it were an all-or-nothing matter. The proposed remedy is a notation in which degrees are explicitly registered.

0 In the sense of Peirce's dictionary definition, the negation of a vague sentence is equally vague, for they share the same blurred boundary. In the present sense, the negation of a vague sentence is general rather than vague; the negation of the vague 'Some man is immortal', for instance, is equivalent to the general 'Every man is mortal'. p 40

Kennedy (2007)

A pile of sand and a 'big chair' are vague predicates. We can treat comparative adjectives such as big, small, tall, bold similar to pile. The vaguness of these

Topic sentence: we add spatial questions for the goal of .... deespite haing lingustic bias?

He traces vagueness to what he calls a law of physics: 'the appearances of a thing at different places are less and less differentiated as we get further away from the thing' russel.

if i take one step closer, is it big ? no. A second step closer, is it big no ?. However, the objects size in the world is the same, so our view does not change the

Evaluation depending on no answers

Our choice to include size and spatial questions is motivated by the belief of the importance of spatial language to cognition. (Landau & Jackendoff (1993)) in "spatial language and spatial cognition" states that the human first acquisition of linguistic names of objects in the physical world is associated with establishing a geometric representation of what defines them. In particular, the conceptual identification of an object might be defined within a spatial relation to other entities, and the image we mentally construct of a concrete noun of a physical property, may appear in the form of its shape.

The annswer

Our question-answer generation is very dependent on the "shortest paths" in EQA-V1 data-set and the objects they lead to.

Task Two- Question Asking (Text to be added)

Training

In the VQA model on the new questions with 50 epochs. In particular, train the LSTM with attention on the visual features, and use a pre-trained CNN for encoding visual features. The pre-trained CNN we use

is one proposed by the researchers in habitat-platform and could be found on EQA gihub page.[14]

We pick the model trained on the last epoch. The model has 1.10 average loss and 0.79 average accuracy.

Evaluating

The overall evaluation on the validation set with all the question types show an average accuracy of 0.61 and average loss 2.10. However, the accuracy score is by no means indicating a good system performance. As mentioned earlier, VQA system could cheat its ways by remembering answers and score high in accuracy results. These scores are attributed to a great extent to the bias we have in the size questions, where most of the answers to size questions are ”medium”. The absolute bias in the size question contribute to a higher score in the average accuracy of the overall validation.

The results of the size questions showed all the predictions to be of ’medium’ answer. In figure 25, the illustration of the predictions shows that all the answers to size questions been predicted as ”medium”. The results of the evaluation of the size questions are not surprising given the significant imbalance in their answer distribution.

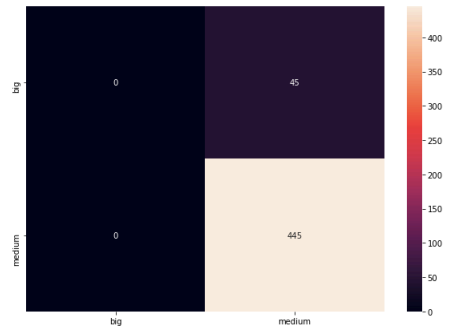
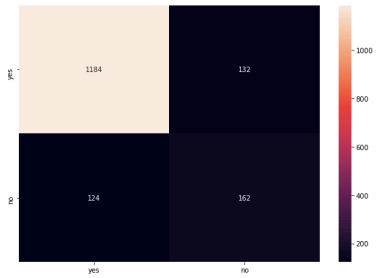


Figure 25: A map shows the number of predictions for each question type sorted by answer. All answers for size questions were predicted ”medium”. The row an column with big categories are zero

A reliable measure of evaluation for spatial questions is measuring the performance on the questions of ”no” answers. The bias to ’yes’ answers in spatial questions was an intended learning experiment, and it is predictable that the system would preform well in predicting the questions with ’yes’ answer. However, the extent in which the model’s predictions are based on exploiting bias could be measure by how many questions with ”no” answer have been predicted correctly. Otherwise, given the bias of the answers to ”yes”, an indicator of learning shortcomings would be the number of times the model answered ’yes’ to a question with ’no’ answer. A positive learning indicator is the number of times that the system predicted question with ’no’ answer correctly. The system’s prediction to questions with ’no’ answer, therefore, provides a good validation measurement to how well the VQA model learnt about spatial relations.

[14]Link to the code for running the VQA baseline model. The same page include an attachment to the pre-trianed-CNN ”https://github.com/facebookresearch/habitat-lab/tree/master/habitat_baselines/il#eqa-cnn-pretrain-model.



(a) label 1

	precision	recall	f1-score	support
no	0.55	0.57	0.56	286
yes	0.91	0.90	0.90	1316
accuracy			0.84	1602
macro avg	0.73	0.73	0.73	1602
weighted avg	0.84	0.84	0.84	1602

(b) label 2

Figure 26: A map shows the number of predictions for each question type sorted by answer.

In the section below we see the difference in the distribution of answer-predictions of the color questions between the original model and the model after being trained on the new questions.

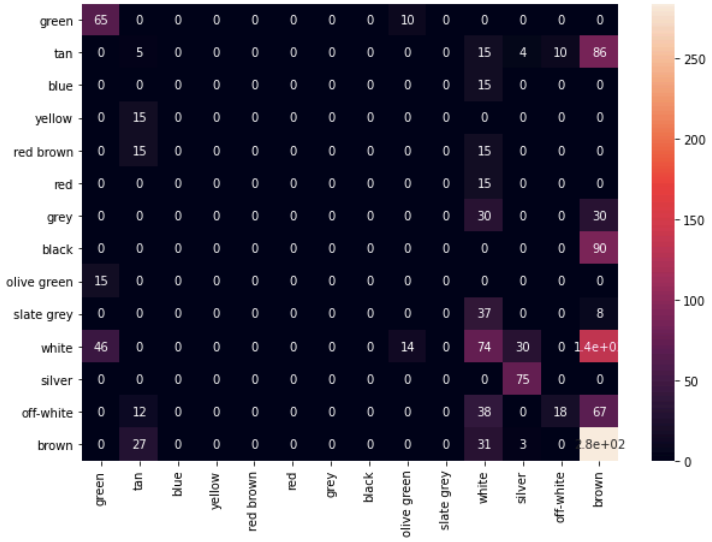


Figure 27: Predictions of color questions for the original model, untrained with new questions

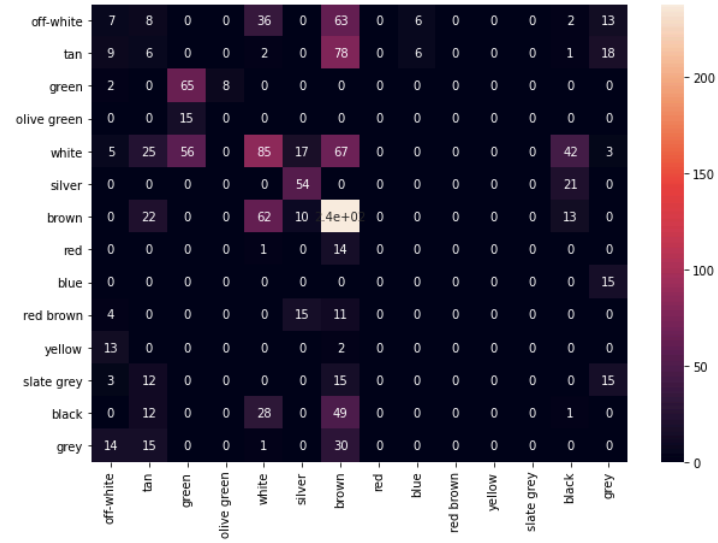


Figure 28: Predictions of color questions on the model trained with new questions

Discussion

We should treat neural models as a theory or brain that is applicable and functional for every task scenario Regier (1996)

Dobnik (2009)

Hcolor questions could get more complex as ”people employ compositional color descriptions to express meanings not covered by basic terms, such as greenish-blue” Monroe et al. (2016). It would be shallow to assume that color questions are simplistic, especially if we expect the system to answer colors beyond the basic color terms like ”green” and ”red.”

Monroe et al. (2017)

intersective compostionality: intersective compositionality is when two words which one is attributed to the meaning of the other ”brown bear” where it means a bear that is brown-[brown and bear]

non-intersective compostionality: non-intersective is one word does not modify the second, such as [Teddy bear]. ’Teddy bear’ cannot be mean a bear that is ’Teddy’, ’Teddy’ is not an attribute of a bear so not [Teddy + bear]. Teddy + bear is instead a different entity with a different perceptual meaning. Larsson (2017)

References

Agrawal, A., Batra, D., & Parikh, D. (2016). Analyzing the behavior of visual question answering models. *arXiv preprint arXiv:1606.07356*.

Agrawal, A., Batra, D., Parikh, D., & Kembhavi, A. (2018). Don’t just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4971–4980).

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., & Parikh, D. (2015). VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*.

Arik, S. O., Chrzanowski, M., Coates, A., Diamos, G., Gibiansky, A., Kang, Y., Li, X., Miller, J., Ng, A., Raiman, J., Sengupta, S., & Shoeybi, M. (2017). Deep voice: Real-time neural text-to-speech.

Austin, J. L. & Warnock, G. J. (1962). *Sense and sensibilia*, volume 83. Clarendon Press Oxford.

Barsalou, L. W. et al. (1999). Perceptual symbol systems. *Behavioral and brain sciences*, 22(4), 577–660.

Black, M. (1937). Vagueness. an exercise in logical analysis. *Philosophy of science*, 4(4), 427–455.

Carnap, R. (1955). Meaning and synonymy in natural languages. *Philosophical studies*, 6(3), 33–47.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., & Zhang, Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., & Batra, D. (2018). Embodied Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J. M. F., Parikh, D., & Batra, D. (2017). Visual dialog.

Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., & Csorba, M. (2001). A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, 17(3), 229–241.

Dobnik, S. (2009). *Teaching mobile robots to use spatial words*. PhD thesis, University of Oxford.

Fang, H., Gupta, S., Iandola, F., Srivastava, R., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J. C., Zitnick, C. L., & Zweig, G. (2015). From captions to visual concepts and back.

Fisher, P. (2000). Sorites paradox and vague geographies. *Fuzzy sets and systems*, 113(1), 7–18.

Fong, T., Nourbakhsh, I., & Dautenhahn, K. (2003). A survey of socially interactive robots. *Robotics and autonomous systems*, 42(3-4), 143–166.

Fukui, A., Park, D. H., Yang, D., Rohrbach, A., Darrell, T., & Rohrbach, M. (2016). Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*.

Gao, H., Mao, J., Zhou, J., Huang, Z., Wang, L., & Xu, W. (2015). Are you talking to a machine? dataset and methods for multilingual image question answering.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1), 142–158.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., & Parikh, D. (2017). Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6904–6913).

Grisetti, G., Kümmerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4), 31–43.

Hariharan, B., Arbeláez, P., Girshick, R., & Malik, J. (2015). Hypercolumns for object segmentation and fine-grained localization.

Harnad, S. (1990). The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3), 335–346.

Hussein, A., Gaber, M. M., Elyan, E., & Jayne, C. (2017). Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2), 1–35.

Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., & Girshick, R. (2017). Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2901–2910).

Kennedy, C. (2007). Vagueness and grammar: The semantics of relative and absolute gradable adjectives. *Linguistics and philosophy*, 30(1), 1–45.

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., Bernstein, M. S., & Li, F.-F. (2016). Visual genome: Connecting language and vision using crowdsourced dense image annotations.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12 (pp. 1097–1105). Red Hook, NY, USA: Curran Associates Inc.

Kruijff, G.-J. M., Zender, H., Jensfelt, P., & Christensen, H. I. (2007). Situated dialogue and spatial organization: What, where… and why? *International Journal of Advanced Robotic Systems*, 4(1), 16.

Lakoff, G. & Johnson, M. (2008). *Metaphors we live by*. University of Chicago press.

Landau, B. & Jackendoff, R. (1993). Whence and whither in spatial language and spatial cognition? *Behavioral and brain sciences*, 16(2), 255–265.

Larsson, S. (2015). Formal semantics for perceptual classification. *Journal of logic and computation*, 25(2), 335–369.

Larsson, S. (2017). Compositionality for perceptual classification. In *IWCS 2017 — 12th International Conference on Computational Semantics — Short papers*.

Lauria, S., Bugmann, G., Kyriacou, T., Bos, J., & Klein, A. (2001). Training personal robots using natural language instruction. *IEEE Intelligent systems*, 16(5), 38–45.

Liang, M. & Hu, X. (2015). Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3367–3375).

Lin, D., Fidler, S., Kong, C., & Urtasun, R. (2014). Visual semantic search: Retrieving videos via complex textual queries. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 2657–2664).

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2015). Microsoft coco: Common objects in context.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3431–3440).

Lowe, D. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2 (pp. 1150–1157 vol.2).

Matuszek, C., FitzGerald, N., Zettlemoyer, L., Bo, L., & Fox, D. (2012). A joint model of language and perception for grounded attribute learning.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.

Monroe, W., Goodman, N. D., & Potts, C. (2016). Learning to generate compositional color descriptions. *arXiv preprint arXiv:1606.03821*.

Monroe, W., Hawkins, R. X., Goodman, N. D., & Potts, C. (2017). Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 5, 325–338.

Mooney, R. J. (2008). Learning to connect language and perception. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)* (pp. 1598–1601). Chicago, IL. Senior Member Paper.

Nilsson, N. J. (2007). The physical symbol system hypothesis: Status and prospects. *50 Years of Artificial Intelligence*, (pp. 9–17).

Quine, W. V. O. (2011). *Two dogmas of empiricism*. Princeton University Press.

Raffman, D. (1996). Vagueness and context-relativity. *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition*, 81(2/3), 175–192.

Regier, T. (1996). *The human semantic potential: Spatial language and constrained connectionism*. MIT Press.

Ren, M., Kiros, R., & Zemel, R. (2015a). Exploring models and data for image question answering.

Ren, S., He, K., Girshick, R. B., & Sun, J. (2015b). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, 1137–1149.

Roy, D. (2005). Semiotic schemas: A framework for grounding language in action and perception. *Artificial Intelligence*, 167(1-2), 170–205.

Russell, B. (1923). Vagueness. *The Australasian Journal of Psychology and Philosophy*, 1(2), 84–92.

Russell, S. J. & Norvig, P. (1995). *Artificial intelligence - a modern approach: the intelligent agent book*. Prentice Hall series in artificial intelligence. Prentice Hall.

Sainsbury, R. M. (2009). *Paradoxes*. Cambridge University Press.

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., & Batra, D. (2019). Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Selvaraju, R. R., Tendulkar, P., Parikh, D., Horvitz, E., Ribeiro, M. T., Nushi, B., & Kamar, E. (2020). Squinting at vqa models: Introspecting vqa models with sub-questions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10003–10011).

Silver, D., Bagnell, J., & Stentz, A. (2008). High performance outdoor navigation from overhead data using imitation learning. *Robotics: Science and Systems IV, Zurich, Switzerland*, 1.

Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Skocaj, D., Kristan, M., Vrecko, A., Mahnic, M., Janícek, M., Kruijff, G., Hanheide, M., Hawes, N., Keller, T., Zillich, M., & Zhou, K. (2011). A system for interactive learning in dialogue with a tutor. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 3387–3394).

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., et al. (2019). The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*.

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., & Batra, D. (2021). Habitat 2.0: Training home assistants to rearrange their habitat. *arXiv preprint arXiv:2106.14405*.

Turney, P. D. & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37, 141–188.

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator.

Wang, H., Zhang, Y., Yu, X., & Solari, F. (2020). An overview of image caption generation methods. *Intell. Neuroscience*, 2020.

Wijmans, E., Datta, S., Maksymets, O., Das, A., Gkioxari, G., Lee, S., Essa, I., Parikh, D., & Batra, D. (2019). Embodied Question Answering in Photorealistic Environments with Point Cloud Perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Williamson, T. (2002). *Vagueness*. Routledge.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J. R., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144.

Xia, F., Zamir, A. R., He, Z., Sax, A., Malik, J., & Savarese, S. (2018). Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 9068–9079).

Yu, L., Park, E., Berg, A. C., & Berg, T. L. (2015). Visual madlibs: Fill in the blank image generation and question answering.

Zhang, C., Platt, J., & Viola, P. (2005). Multiple instance boosting for object detection. *Advances in neural information processing systems*, 18, 1417–1424.

Zhang, L., Wei, L., Shen, P., Wei, W., Zhu, G., & Song, J. (2018). Semantic slam based on object detection and improved octomap. *IEEE Access*, 6, 75545–75559.

Zhang, P., Goyal, Y., Summers-Stay, D., Batra, D., & Parikh, D. (2016). Yin and yang: Balancing and answering binary visual questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5014–5022).

Zhu, Y., Fathi, A., & Fei-Fei, L. (2014). Reasoning about object affordances in a knowledge base representation. In *European conference on computer vision* (pp. 408–424).: Springer.

Zhu, Y., Groth, O., Bernstein, M., & Fei-Fei, L. (2016). Visual7w: Grounded question answering in images.

Zhu, Y., Zhang, C., Ré, C., & Fei-Fei, L. (2015). Building a large-scale multimodal knowledge base system for answering visual queries. *arXiv preprint arXiv:1507.05670*.

Zitnick, C. L., Parikh, D., & Vanderwende, L. (2013). Learning the visual interpretation of sentences. In *2013 IEEE International Conference on Computer Vision* (pp. 1681–1688).

Appendices

datasets

The 3D Scenes and the QA dataset mentioned in Das et al. (2018), are called SUNCG(3D houses) and ”EQA V1” (QA). The EQA V1 is a synthetic dataset generated automatically, and constructed based on the setting of the 3D houses in SUNCG. SUNCG is no longer available. Das et al. (2018) changed the SUNCG 3D setting to MatterPort 3D (MP3D). MatterPort 3D is a reconstruction of 3D houses in (SUNCG) scene dataset. The latter also implies that the inital ”EQA V1” is not applicable for MP3D.

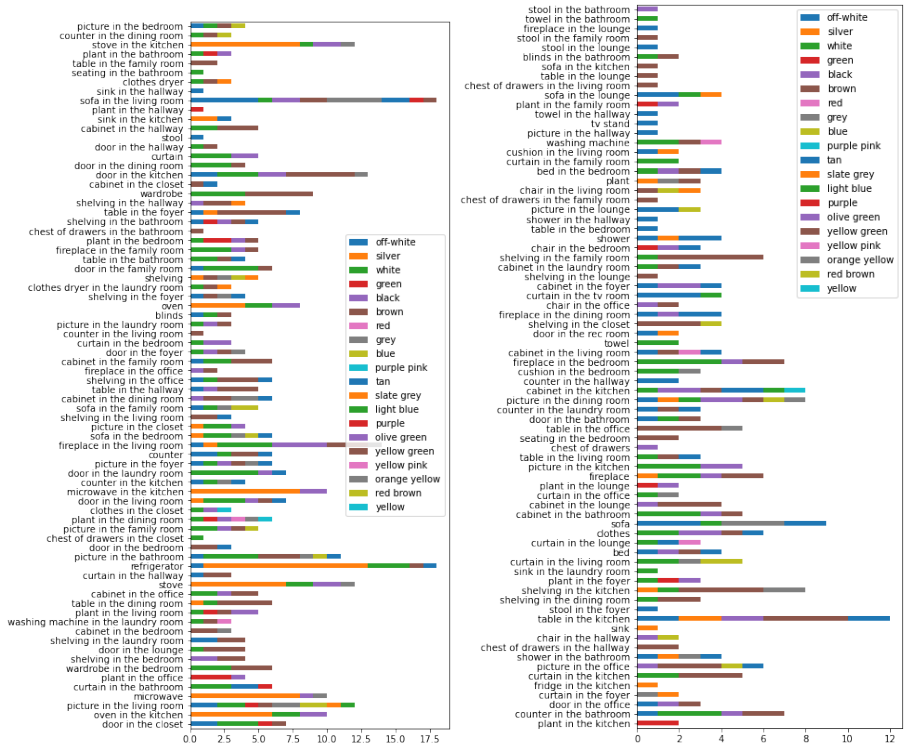
The new QA dataset for Matterport 3D is available but not the code that generated it. The EQA-mp3d v1 is also a synthetic dataset generated automatically and can be found at this footnote reference [15]. For generating questions for SUNCG, a code published at this reference[16]. However, there is no code for generating QA for MP3D.

A few of the differences between the question dataset for SUNCG (EQA-SUNCG) and MP3D(EQA-MP3d) are mentioned in Wijmans et al. (2019). However, not in all the information in Wijmans et al. (2019) seems to match with EQA-MP3D that we have. In Wijmans et al. (2019) page(4) it’s stated that the number of scene used from MP3D is 76. The dataset we downloaded from ”facebookai/habitat” repo on github uses a total 67 scene of 90 scenes available in MatterPort3D. 57 of the 67 scenes are used for questions in the train-set and 10 in the the enviroment. Note that the latter implies that the robot is tested on different scenes from the scenes it has been trained in.

List of textual references with number of answer choices

.

[15] https://github.com/facebookresearch/habitat-lab
[16] https://github.com/facebookresearch/EmbodiedQA



(a) (b)

Figure 29: Each row is a textual reference. The number of answer choices for each textual reference is represented by the colorful blocks. One block = 1 answer choice and so on. The colors of the bars are not representative of the named color answer so they should be disragarded

.

Habitat-lab(EQA evaluation)

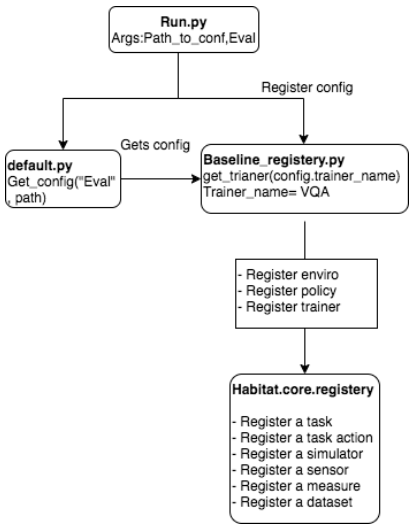


Figure 30: Example of Habitat lab processing the configurations to implement validation for the VQA model

Figure 30 resembles a map of the code structure when the habitat lab module is initiated to preform validation task for the VQA. Each task has its own configurations and in this example the task is 'VQA evaluation'. As seen in the figure 30, the module takes hierarchical steps in which each step is executed in accordance to the configuration of the given task. In the most down box of the structure we see parts of the commands directed for the simulator, such as insinuating an environment and sensors in the agent. Other commands include registering a data-set which takes part in lab module.

The configurations are processed into commands in Habitat-lab before being passed to the simulator. Habitat lab is the second core component of the system. In addition to giving commands to the simulator, the Habitat Lab module acts as a pipeline that prepares the data-set of the corresponding task. The habitat-lab module,in other words, is the coordinator that informs the simulator of the required setting, and the data loader and processor that prepares the data for either training or testing.

y in science and engineering (from aerospace to zoology). In the context of embodied AI, simulators help overcome the aforementioned challenges – they can run orders of magnitude faster than real-time and can be parallelized over a cluster; training in simulation is safe, cheap, and enables fair comparison and benchmarking of progress in a concerted communitywide effort. Once a promising approach has been developed and tested in simulation, it can be transferred to physical platforms that operate in the real world

. The name 'Habitat' is derived from the notion of learning within and from an environment. Imitating our natural habitat, the Habitat platform facilitates spawning an agent in a simulated environments with the possibility of teaching the robot to preform different tasks.

The perquisites needed to test or train an agent for a certain task in a given environment, are facilitated by a core component called Habitat Simulator. Habitat Simulator is responsible for simulating an environment and insinuating a robot in it. The simulator acts depending on the configurations given to it.