3DVQA: Visual Question Answering for 3D Environments

Yasaman Etesam, Leon Kochiev, Angel X. Chang
yetesam, lkochiev, angelx@sfu.ca
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
https://3dlg-hcvc.github.io/3DVQA/

***Abstract*—Visual Question Answering (VQA) is a widely studied problem in computer vision and natural language processing. However, current approaches to VQA have been investigated primarily in the 2D image domain. We study VQA in the 3D domain, with our input being point clouds of real-world 3D scenes, instead of 2D images. We believe that this 3D data modality provide richer spatial relation information that is of interest in the VQA task. In this paper, we introduce the 3DVQA-ScanNet dataset, the first VQA dataset in 3D, and we investigate the performance of a spectrum of baseline approaches on the 3D VQA task.**

***Keywords*-3D Visual Question Answering; Visual Question Answering; 3DVQA, 3D;**

I. Introduction

The Visual Turing Test [14] was proposed as a test of an AI agent's ability to perceive and reason, and requires integration of visual and language capabilities. Motivated by this challenge, the Visual Question Answering (VQA) [5, 56] problem has been widely studied in the vision-and-language community. VQA tasks span a spectrum from visual reasoning in abstract 'blocks world' settings [20], to answering questions about complex real scenes while incorporating common-sense knowledge [56].

However, this work on VQA is limited to reasoning in 2D and cannot measure an AI agent's ability to reason in 3D environments. There is work on VQA for 360° panoramas [10] but this setting is still subject to the limitations of visual reasoning only within a 2D image domain. Recently, embodied question answering (EQA) [12, 15] was proposed to investigate an agent's ability to answer questions in a 3D environment where it can move, act, and perceive. The EQA setting is quite complex as it couples navigation, interaction, perception, and reasoning. This makes it difficult to disentangle whether failure to provide correct answers is due to the inability to move or act correctly, or an inability to model and reason about the 3D environment. Moreover, dataset biases may imply the agent does not even need perception to provide a correct response [3].

In this paper, we investigate the ability of a model to answer questions given a 3D environment represented as a 3D point cloud. The 3D point cloud provides 3D structure information, and allows us to focus on spatial relations and size. We hope to pave the way for future work on reasoning over 3D representations and grounding to language.

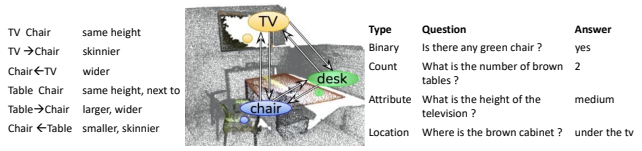

Figure 1: We introduce VQA in the 3D setting. We take as input a 3D point cloud (center), and construct a dataset of questions and answers (right) using scene-graphs for indoor scenes from ScanNet. A partial scene graph for three objects in the scene is shown (center) with nodes representing objects and edges representing relations between objects (left).

To this end, we create a 3D VQA dataset using 3D reconstructions of real environments from ScanNet [11], and systematically compare the performance of models on answering a range of VQA questions. To support generation of questions based on spatial relations, we construct a 3D scene graph for each ScanNet scene. Using the 3D scene graph, we create a set of questions and answers (see Figure 1). We use synthetically generated questions to control the complexity of the questions and to control the aspects of vision-language reasoning we study. This paradigm follows prior work such as CLEVR [20] and GQA [19] which also programmatically generate question-answer pairs. Compared with CLEVR [20], which consists of a few simple shapes (cube, sphere, and cylinder), our dataset consists of various real-world objects over 500 categories. Our aim is similar to GQA [19] (to provide a VQA dataset for investigating reasoning and compositionality in real-world scenarios), but we focus on 3D indoor environments instead of 2D images.

We consider four kinds of questions: *counting*, *query attribute*, *location*, and *binary (yes/no)* questions. We use attributes and spatial relations to select focus objects for which questions and answers are generated. We compare and analyze the performance of several models on this dataset. Our study includes a simple attentive model using LSTM and different feature encodings for point clouds, as well as a graph-based model using the Neural State Machine (NSM) [17]. We compare these models to 'language-only', 'vision-only', and random baselines. We find the accuracy of our models depends on the difficulty of the question and complexity of the scene and show that using a 3D object detector improves 3DVQA performance. We also find that images are more helpful for color understanding, and point

2022 19th Conference on Robots and Vision (CRV) | 978-1-6654-9774-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CRV55824.2022.00038

978-1-6654-9774-9/22/$31.00 ©2022 IEEE
DOI 10.1109/CRV55824.2022.00038

clouds helps with questions about sizes and spatial relations.

In summary, we introduce the '3D VQA' task in which the VQA is performed on 3D point clouds instead of 2D images. We create the 3DVQA-ScanNet dataset with a range of visual reasoning questions and answers for 3D point clouds of real-world scenes derived from ScanNet. As part of the 3DVQA-ScanNet dataset we also contribute a 3D scene-graph for each ScanNet scene. Lastly, we compare how well current models and simpler baselines can address the 3D VQA task. Our work is among the first to investigate the VQA task in 3D scenes.[1]. Concurrent with our work, other groups have started to develop visual-question answering for 3D [53, 6, 52], indicating growing interest in this area.

II. Related work

2D VQA The introduction of visual question answering with images (2D VQA) [5, 56] has lead to the development of various models [32, 42, 51, 4] and datasets for VQA focusing on reasoning [20], common sense [54]. Work in VQA has investigated the use of various types of attention [32, 42, 51, 24, 25], different language encoders and visual backbones [45], models for reasoning [21, 16, 18, 17], incorporating knowledge bases [55, 49, 34], and pretraining with transformers [33, 43]. The focus of our work is not to develop a new architecture for VQA, but to study VQA in the 3D domain, we use simple attention based models adapted to use 3D features.

The introduction of VQA has spurred work in QA for figures and charts [13, 22, 23], document images [35], videos [44, 30]. This work considers question answering in 2D and does not explore 3D relations. In embodied settings [12, 15], the agent is moving in a 3D environments, but uses as input egocentric 2D frames and does use have explicit 3D representation. In contrast, we develop a 3D VQA dataset that allows us to investigate 3D reasoning.

Synthetic data for visual reasoning CLEVR [20] popularized the use of generating controlled language to study reasoning and compositionality. Following CLEVR, Hudson and Manning [19] used scene graphs from Visual Genome[29] to generate questions for real-world images with more diverse vocabulary and visual input. Similarly, templates were used to generate questions for VQA in panoramas [10], figures [13, 22], and IQA/EQA [15, 12, 50].

3D and language There has been increasing interest in connecting language to 3D representations such as disambiguating 3D objects [1, 2], localizing 3D objects in scenes [37, 28, 7], as well as captioning in 3D [8], and generating 3D shapes from language [9].

Concurrent with our work, recent work has started to consider 3D VQA [53, 6, 52]. Both Ye et al. [53] and Azuma et al. [6] build their question answering datasets on top of ScanNet, with Ye et al. [53] collecting questions and answers from crowdworkers. Azuma et al. [6] generate questions and answers automatically using a transformer-based language model [41], and further refine the questions and answers so they are grounded to the 3D scenes using crowdworkers. Yan et al. [52] leverages the 3RScan [47] dataset with annotated 3D semantic scene graphs [48] to construct templated questions and answers. We follow a similar approach as Yan et al. [52] using ScanNet scenes and construct scene graphs for ScanNet [11]. All three works use transformers [46]-based methods to tackle the problem. In this work, we aim to provide baselines with simpler models.

[1]Part of this work is described in Kochiev [27].

III. Dataset

We introduce the 3DVQA-ScanNet dataset, built from 3D reconstructions of real-world environments from ScanNet. To create the dataset, we create a 3D scene graph for each unique ScanNet scene. The 3D scene graph consists of objects at the nodes, and relationships between the objects. Using the 3D scene graph, we generate question/answer pairs using templates (see supplement for details).

By generating questions from the scene graph, we have control over the complexity of our dataset. We define the question-difficulty as the number of times we need to look at the scene graph to determine the correct answer. As a result, we have the opportunity to study the effect of complexity level on the performance of different models. We also investigate how each of the attributes and spatial relations affect different models.

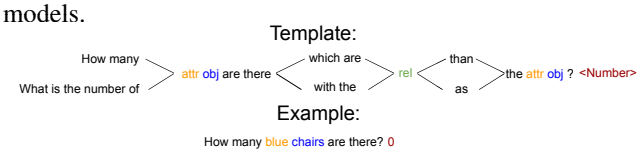

Figure 2: Question template and answer.

A. Scene graph generation

We build scene graphs using annotated objects from ScanNet and their oriented bounding boxes (OBB). We extract four attributes for each object: color, lightness, height, and size (volume). See supplement for more details.

For the color, we assume that each object can be described with a single color. To obtain the color name for an object, we classify the color of each point using a nearest neighbor match to a list of predefined RGB colors. To account for variation of human visual sensitivity to different parts of the visible spectrum and the common use of gamma-corrected color values in cameras, we follow Li et al. [31] and measure the distance between two RGB colors as: $D(RGB, rgb) = \sqrt{((R-r)\cdot 0.3)^2 + ((G-g)\cdot 0.59)^2 + ((B-b)\cdot 0.11)^2}$ Using this distance, we match against a list of predefined colors commonly used in indoor scenes. We take the majority vote of the points to obtain the final color name.

We compute object lightness by converting the RGB color for each point to HSL and taking the mean of lightness

Table I: Dataset statistics. We use the train/val/test split of scenes from ScanNet v1.

	Train	Val	Test	Total
# scenes	494	71	142	707
# questions	326520	53395	104444	484359
ave # tokens	13.256	13.232	13.348	13.273
vocab size	930	638	782	961
vocab in train	-	626	758	-
# unique questions	308893	52357	101618	454144
# question in train	-	3859	7639	-
question difficulty	4.0501	4.0851	4.0690	4.0581

(L) for all points of the object. For size attributes (height, volume), we use the axes length of the OBBs to estimate the height (z) and the bounding box volume (size).[2]

Since these are gradeable attributes, we then convert each into an appropriate coarse adjective by grouping the attribute values into three levels (below average, within average, and above average). We take the mean and standard deviation for each measured attribute within each object category, and consider instances that are within one standard deviation of the mean to be 'within average', and others to be below or above average. We then choose the appropriate adjective (short vs tall, small vs large) for describing the attribute.

After determining the attributes for each object, we establish relationships between the objects. We consider spatial relations and relations comparing attributes between the objects. Since we are operating in a view-agnostic manner, we include only view-independent spatial relationships (`on`, `under`, `above`, `support`, `next to`, and `between`). For comparative relations, we use the attributes to determine if two objects are: `same color`, `lighter`, `darker`, `same category`, `same volume`, `larger`, `smaller`, `same height`, `taller`, `shorter`, `same width`, `skinnier`, and `wider`. We also include the `between` relation based on whether the bounding box of one object is in between the bounding box of two other objects. Based on this information, the scene graph for each scene is generated (see fig. 1).

B. Question and answer generation

Based on the scene graphs, we generate four types of questions using templates: *counting*, *query attribute*, *location*, and *binary (yes/no)*. The four types of questions correspond to different answer types. Using the templates, we can generate questions of varying complexity with different lexical surface forms. We define the question difficulty as the number of times that the scene graph needs to be consulted to process the question (i.e. the number of object names, attributes, and relations that is referenced in the question). We generate questions ranging from difficulty level 1 to 5.

[2]These bounding boxes do not necessarily reflect the true size of the object, but provide coarse scale.

Figure 2 shows an example template with generated question and answer pair. *Counting* questions are designed to assess the ability of a model to pick out and count the number of objects matching a set of attributes and relations. The *query attribute* questions ask about the color, height, or size of an object. Answers to *location* questions are designed to be in the form of a short phrase, indicating the relative spatial location of the focus object relative to another object (or two other objects for the "between" relation). For *binary (yes/no)* questions, we generate questions that require checking the existence of a specified object, comparison of counts, and checking the relations. *Location* questions are the most challenging as the space of answers is the largest. In contrast, *binary (yes/no)* questions have just 2 answer options ('yes', 'no') and are the easiest to answer.

To determine the answer, we traverse the scene graph to identify nodes in the graph matching the focus objects (category and attributes). If there are multiple objects that match the reference, we consider the question to be ambiguous and discard the question. We then consider the relationship between the matched nodes to determine the appropriate answer. For *query attribute*, *location*, and *check relation*, we also ensure that the target object is non-ambiguous.

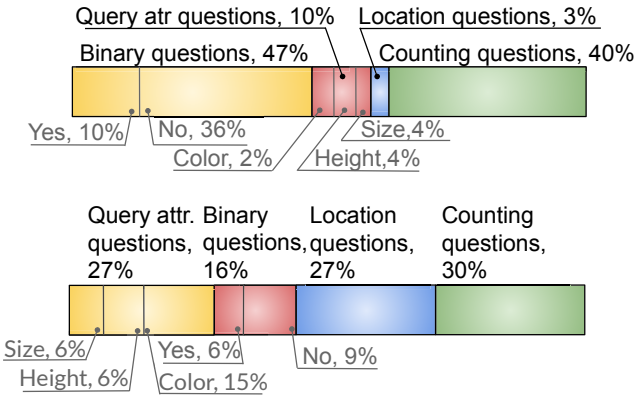

Figure 3: Question distribution for balanced (bottom) and full (top) dataset.

1) Dataset statistics and analysis: In total, we generated more than 40M questions for 707 scenes, from which we sampled around 0.5M questions. As noted above, we discard any questions that refer to ambiguous objects.

Dataset balancing. We balance our dataset between the different types of questions. Since binary questions are easier, we aim for less binary questions than other question categories, resulting in a split of 16% binary questions and $27-30\%$ for the other question types (see fig. 3).

Dataset statistics. Table I shows the statistics of our final dataset after balancing. We split our dataset into train/val/test split based on scenes following the split used in ScanNet v1. This ensures that evaluation is done on unseen scenes.

Human assessment. To ensure that the generated questions

and answers agree with human judgement, we sampled 153 questions from 4 scenes. Two of the authors wrote answers to the questions (following the designed answer templates). Our generated answers matched at least one human response 84.31% of the time and matched both 66.01% of the time. Humans were able to answer *binary (yes/no)* and *counting* questions with high accuracy (91.89%). For *query attribute* questions, color was easier for humans than height/size questions. Answers to *location* questions had the most variation, matching generated answers less frequently due to the free-form nature of the answer.

IV. Models

1) 3D encoding and object detection: We compare two different approaches to obtain vector representations of the 3D point clouds.

PointNet++. We use the point cloud P as input to a PointNet++ [40] which uses a series of set abstraction (SA) modules that hierarchically group and sample the points. In each jth SA layer, PointNet [38] is used to encode each input point $q_i \in \mathbb{R}^{C_{j-1}+3}$ into features $f_i \in \mathbb{R}^{C_j+3}$, where 3 is for the 3-dimensional (x, y, z) position of the point, and C_{j-1} and C_j are the input and output feature dimensions. We use the standard PointNet++ architecture with 4 SA layers to encode our colored point cloud (with $C_0 = 3$). At the end of the process we obtain an encoded visual representation for the 3D scene $V \in \mathbb{R}^{N_2 \times (C_2+3)}$ with $N_2 = 1024, C_2 = 13$. Note that for our visual representation we have a reduced set of representative points with encoded features as well as the original position. We pre-train PointNet++ for semantic segmentation over 20 different object categories in ScanNet.[3]

VoteNet. We also consider obtaining object-level features using VoteNet [39], a 3D object detection network. VoteNet uses PointNet++ as a backbone and provides object proposals for 18 object categories.[4] VoteNet takes the output features from PointNet++ and produces K object proposals with m sampled points for each proposal. With VoteNet, our visual representation of the scene is $V \in \mathbb{R}^{K \times m \times (C+3)}$, where C is the embedded feature dimension. For our experiments, we use $K = 256$, $C = 256$ and $m = 16$. Using VoteNet, we capture visual information at the object level.

2) Answer prediction: To get the answers from visual and textual representations, we use two approaches:

Softmax classifier. After concatenating the visual and text vectors, we use a simple classifier consisting of linear, ReLU, and Softmax layers on the combined vector to produce the final answer among all possible answers (fig. 4).

Seq2seq. As shown in fig. 4, we also use seq2seq decoder consisting of an LSTM to predict answers word by word.

[3]These categories are floor, wall, cabinet, bed, chair, sofa, table, door, window, bookshelf, picture, counter, desk, curtain, refrigerator, bathtub, shower curtain, toilet, sink, and otherfurniture.

[4]Excludes floor and wall from the above categories.

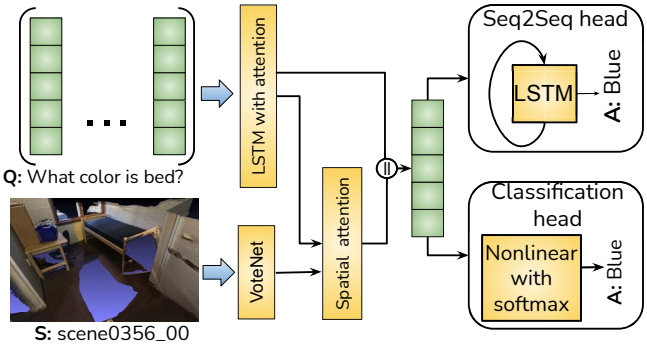

Figure 4: Our 3D fused attention model takes in as input: i) a question and encodes it using a LSTM language encoder (top); and ii) a 3D point cloud and encodes it using PointNet++ or VoteNet (bottom). We fuse the two inputs using language and spatial attention, and feed it to our answer prediction module. We compare two different answer prediction modules: i) a simple classifier consisting of a linear layer, ReLU, and softmax; and ii) a seq2seq decoder which generates the answer word by word.

This is useful for multi-word answers in response to location questions. The input is a concatenation of the encoded question and 3D features which is set as the initial hidden state of the decoder LSTM. We use teacher-forcing to train the network. During decoding, we use attention over the input sequence (see supplement for details).

A. 3D Fused Attention

In our fused attention model, we adapt the multimodal low-rank bilinear attention network (MLB) proposed by Kim et al. [24] to 3D VQA. For question encoding, we use a single-layer LSTM with GloVE [36] embeddings of size 300. The input to the fused attention model is the final hidden state $h_n \in \mathbb{R}^{d_l}$ of the LSTM as the question encoding $\mathbf{q}$, and visual features $V \in \mathbb{R}^{K \times d_v}$ extracted by the 3D encoder (PointNet++ or VoteNet). For PointNet++ we use $K = N_2 = 1024$ and for VoteNet we use $K = N = 256$.

We then apply language-guided spatial attention on the 3D features V to obtain attention-weighted visual features. To compute the attention, we first project the visual V and language features h_n into a common space by passing each of them through fully connected layers f_v and f_q with $f_v(V) = \sigma(\mathbf{W}_v V^T)$, $f_l(\mathbf{q}) = \sigma(\mathbf{W}_l \mathbf{q})$ where $\mathbf{W}_v \in \mathbb{R}^{d \times d_v}$, $\mathbf{W}_v \in \mathbb{R}^{d \times d_l}$ and σ is a non-linear activation. We use $\sigma = \mathrm{ReLu}$ as we experimentally found ReLu worked better than $\tanh$. We compute attention over the K visual features v_i to obtain the attended visual features $v_{\text{att}} = \sum_{i=1}^{K} \alpha_i v_i$ with $\alpha = \mathrm{softmax}(\mathbf{g})$ where $\mathbf{g}$ are the attention scores. The attention scores are computed by taking the low-rank bilinear approximation $\mathbf{g} = \mathbf{w}_a(f_v(V) \circ f_q(h_n \cdot \mathbb{1}^T))$ where $\circ$ is the Hadamard product (i.e. element-wise multiplication), $\mathbb{1} \in \mathbb{R}^K$ is a column vector of ones, and $\mathbf{w}_a \in \mathbb{R}^d$ are learned weights.

Note that bias terms are omitted for simplicity. Finally, we concatenate the visual and language representations and pass them to our answer prediction module (section IV-2).

B. Neural State Machine

As an alternative to the above 3D fused attention model, we adapt the NSM model from Hudson and Manning [17] to the 3D domain. The NSM predicts answers using attention-based reasoning over the scene graphs. For the NSM, we use the same language encoder and answer classification module as for other models. The visual input to the NSM is a scene graph, which we extract from the point cloud by using VoteNet to identify objects and heuristics to determine relations between the objects (see supplement for details).

V. Experiments

A. Implementation details

All our models are implemented in PyTorch. We use the official VoteNet implementation[5] and a PyTorch port of PointNet++[6]. We train our models on a workstation with a Core i9-9900K CPU and RTX 2080 Ti GPU. The stopping criterion was a change of less than 0.0001 in validation set accuracy between epochs. We used dropout on the last network layers. For the 3D fused model, we used ADAM [26] with an initial learning rate of 0.001, decaying the learning rate by half every two epochs, and a dropout rate of 0.7. We trained the network for up to 30 epochs, with the training stopping when the validation accuracy stabilizes (increase less than 0.0001), typically after 10 epochs. For NSM, we trained it using ADAM with a learning rate of 0.0003, decaying the learning rate by half every epoch. We used a last layer dropout rate of 0.15 and trained up to 6 epochs.

B. Evaluation metrics

We evaluate our models using three metrics: *accuracy*, *validity* and *distribution* following the setup of Hudson and Manning [19]. *Validity* checks how often the predicted answer belongs to the valid set of answers for the given question. For instance, a valid prediction to a color question is a color name. The *distribution* metric uses the Chi-Square statistic to measure the difference of the distribution of the answers predicted by the model against the distribution of the ground truth answers. It measures whether the model is able to predict infrequent answers as well as the most common answers. Note that lower distribution values are better.

C. Baselines

We compare the described methods in Section IV against several baselines: random, majority, language-only, vision-only, and 2D-VQA approach with a bird-eye-view (BEV).

[5]https://github.com/facebookresearch/votenet

[6]https://github.com/daveredrum/Pointnet2.ScanNet

Random. We randomly select from all possible answers in the train set (**Rand**), or from possible answers conditioned on question type (**Rand(Q-type)**).

Majority. We select the most frequent answer from the training set is used. We condition this based on the question type (**Maj(Q-type)**) and the question (**Maj(Q)**).

Language-only. We also consider language-only baselines with a Bag-of-words (BOW) model and a sequence model based on an LSTM. For the BOW model, we consider both TFIDF and one-hot encoding versions. We compare the BOW model to the LSTM (with attention) with either a simple classifier (**LSTM+cls**) or a sequence decoder (**LSTM+seq**).

Vision-only. For the vision-only baselines, we use the visual representations obtained by **PointNet++ (PN++)** and **VoteNet (VN)** to predict answers using a simple classifier.

2D VQA Baseline. We also consider a 2D VQA baseline where instead of using the point clouds, we use a 2D top-down rendering of the scene as input. We use ResNet-18 on the 2D top-down rendering to get the visual representation. The concatenation of this vector and the language representation is passed through a classifier to predict the answer. Since we use the bird-eye-view (BEV) as input, we refer to the vision-only variant as **BEV** and the the language+vision model as **LSTM+BEV**.

D. Results

In our experiments, we follow the ScanNet v1 scene split, ensuring that the scenes are distinct in train, val, and test. Table III compares the performance of different models using the accuracy, validity and distribution metrics. The random baselines show that it is challenging to randomly guess the answer. The majority baselines show that there is some bias in the dataset (e.g., most binary questions are answered by 'no'). The importance of language is reflected in the strong performance of the language-only baseline (LSTM). In contrast, the low performance of the 3D point-cloud only baselines (PointNet++ and VoteNet), shows that since there are many questions for each scene, the text of the question is necessary for determining the answer. Models that combine information from the question and the visual modality, improve the performance only slightly. Incorporating VoteNet (LSTM+VN) improves performance by 0.9% while using the top-down view (LSTM+BEV) improves the performance by 0.5%. In our experiments, using PointNet++ (LSTM+PN++) did not improve performance.

Overall, the accuracy and validity are largely correlated across models, but the distribution is not. The lowest distribution is for the vision-only PointNet++ model which has the poor accuracy and validity. We also study how different methods of encoding and fusing point clouds affect the results (see Table IV). We find that using seq2seq helps the overall performance, but we choose to report the results for the classifer based prediction for consistency in Table III.

Table II: For the smaller dataset of only 18 categories, VoteNet (VN) outperforms 2DVQA and LSTM in all question types. We also see that 2DVQA performs well on color questions while VoteNet performs better on questions related to geometry.

	Val							Test						
Method	Acc	loc	count	query	y/n	clr	h/s	Acc	loc	count	query	y/n	clr	height/size
LSTM	37.71	6.29	46.20	**39.43**	**78.69**	33.59	28.38/36.27	38.25	6.47	47.72	**39.43**	**74.49**	40.58	**38.54**/37.89
LSTM + BEV(2D)	38.34	6.71	**47.69**	35.45	75.25	**35.93**	32.24/37.70	36.53	4.90	45.26	37.86	73.55	**43.10**	29.62/34.63
LSTM + VN	**39.12**	**10.82**	46.04	36.70	74.46	33.59	**39.38**/41.19	**38.88**	**9.45**	**48.68**	39.18	72.02	40.58	34.26/**40.75**
NSM (pred)	34.13	5.97	40.68	36.12	76.11	30.79	38.41/43.06	33.50	6.71	42.01	36.25	71.64	34.29	36.02/38.57
NSM (GT)	36.71	6.48	41.01	36.09	76.90	31.26	39.19/**44.26**	35.51	6.89	41.68	36.60	72.31	35.26	36.66/39.24

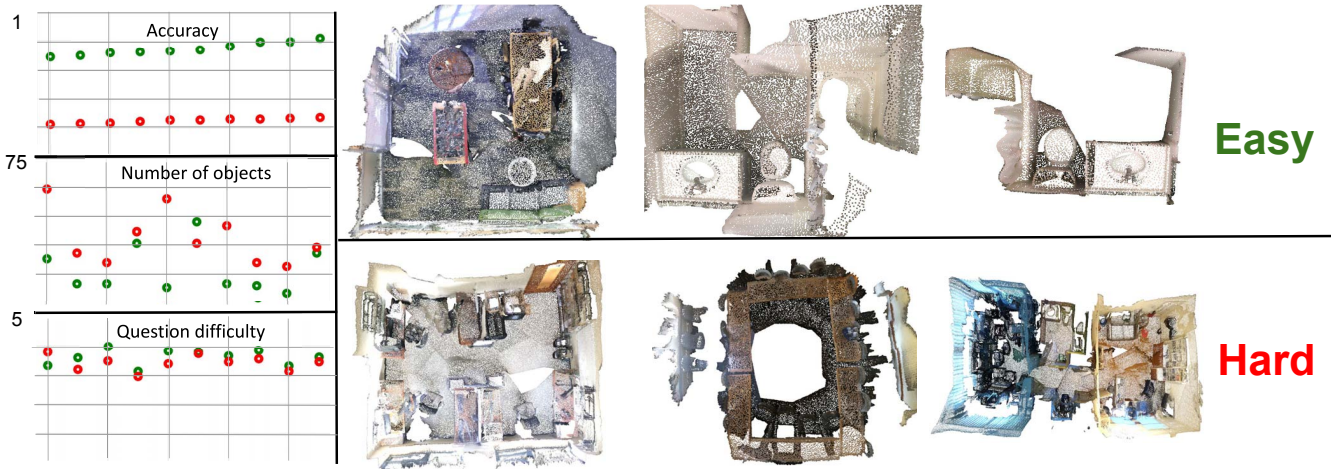

Figure 5: Example scenes with the best and worst accuracies for LSTM + VoteNet with attention on the test set. We also plot the scene complexity ("number of objects") and question difficulty for each of the scenes. We see that the most challenging scenes have a large number of objects. The question difficulty remains relatively constant across the scenes.

Table III: Performance of different models on 3DVQA-ScanNet. Accuracy, validity, distribution trends are most correlated across models. Note NSM (pred) uses predicted scene-graphs and NSM (GT) uses ground-truth scene graphs.

	Val			Test		
Method	Acc↑	Val↑	Dist↓	Acc↑	Val↑	Dist↓
Rand	0.03	3.56	133.58	0.05	5.27	267.88
Rand(Q-type)	11.08	44.58	2141.49	9.92	46.78	5564.31
Maj(Q)	2.50	6.64	120.01	2.42	6.81	237.60
Maj(Q-type)	27.44	87.72	2082.14	25.04	86.44	2113.33
LSTM	42.02	98.91	1874.36	41.44	98.72	1652.45
BEV(2D)	9.21	28.88	163.33	8.72	28.84	314.92
PointNet++	10.24	26.79	**88.62**	8.91	25.17	**183.67**
VoteNet	11.47	30.79	174.44	9.62	29.59	381.47
LSTM + BEV(2D)	42.52	96.94	1324.47	40.75	97.56	609.86
LSTM + PN++	40.22	96.56	429.53	38.39	96.75	624.98
LSTM + VN	**42.98**	**98.99**	542.28	**43.07**	**98.76**	608.99
NSM (pred)	37.12	96.04	275.48	37.24	95.79	312.60
NSM (GT)	41.32	95.68	252.32	41.44	96.91	234.40

Table IV: Ablations showing the impact of PointNet++ (PN++) vs VoteNet (VN) as the 3D feature encoder, use of attention (att), and answer prediction with a softmax classifier (cls) vs predicting answer word sequence (seq). We also compare end-to-end trained vs fixed PointNet++/VoteNet.

Method	lang	3D	fus	ans	e2e	accuracy
(a)	BOW (TFIDF)	-		cls	-	8.31
(b)	BOW (One Hot)	-		cls	-	27.03
(c)	LSTM	-		cls	-	42.02
(d)	LSTM	-		seq	-	42.81
(a)	-	PN++		cls	-	10.23
(b)	-	VN		cls	-	11.46
(a)	LSTM	PN++		cls	N	29.43
(b)	LSTM	PN++	att	cls	N	40.22
(c)	LSTM	VN		cls	N	40.38
(d)	LSTM	VN	att	cls	N	42.98
(e)	LSTM	PN++		cls	Y	39.70
(f)	LSTM	PN++	att	cls	Y	40.32
(a)	LSTM	PN++		seq	N	26.20
(b)	LSTM	PN++	att	seq	N	37.25
(c)	LSTM	VN		seq	N	38.97
(d)	LSTM	VN	att	seq	N	**44.27**

E. Analysis

To further analyze the performance of our models, we investigate how different inputs affect the results.

Visual and linguistic complexity We measure visual complexity of the scene using the number of objects in the scene and the linguistic complexity by the length of the question. As expected, the average accuracy decreases as the visual

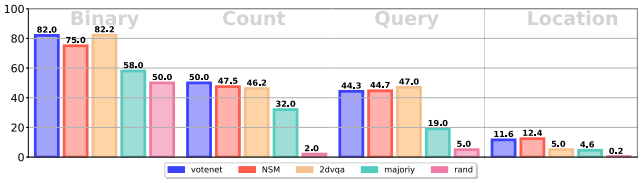

Figure 6: Accuracy for each question type. As expected, binary questions are relatively simple to answer while location questions are the most challenging.

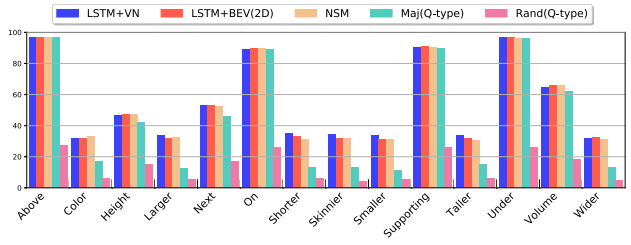

Figure 7: Accuracy for relation types. We find vertical spatial relations (above, under, support) to be the easiest while other relations are more challenging.

and linguistic complexity increases. We provide more details on the analysis in the supplement.

What are the easiest and hardest scenes? Figure 5 show 10 scenes for which the LSTM + VoteNet (att) model performed the best and worst. We see that some scenes are simpler (with only 5 objects), while other scenes have up to 59 objects). The question difficulty is mostly constant across the scenes.

Are some question types more challenging? Figure 6 shows the accuracy for each question type. Some questions are more challenging than others, with *binary (yes/no)* questions being the easiest (with a chance accuracy of 50%) and *location* questions being the hardest (with a chance accuracy of 0.018% over 5586 answers). Not surprisingly, random has low performance on all question types. NSM and LSTM+VN have a high performance on location questions compared to other methods. In contrast, there are only two possible answers for the binary questions, so it much easier for the model to learn how to answer them.

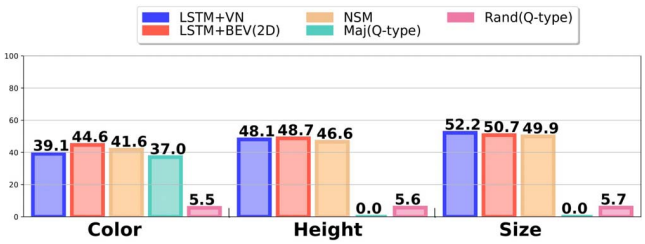

Figure 8: Accuracy for attribute types. We find color to be more challenging than height/size. This is likely due to the noisy colors of the scan.

Are some object categories more challenging? The distribution of objects follows a long-tail distribution, with frequently represented objects having stable accuracy across all methods, and rare object categories displaying high variance in accuracy (see supplement for detailed analysis).

Are some relations more challenging? Figure 7 shows that questions with vertical spatial relations have high accuracies while other relations (shorter, taller, larger, smaller, wider, skinnier) have the minimum accuracy. This is likely due to the limitations of the object detection.

Query attribute question accuracy. Figure 8 shows the accuracy of different methods for different query attributes. From Figure 8, we see that the 2D-VQA model using the top-down view is able to answer color questions more accurately, and that the models using the 3D point cloud are better at answering questions involving size.

Reduced category analysis. Since VoteNet is trained with a small set of 18 categories, we conduct an experiment to study the performance of our models on a reduced dataset of only 18 categories (see supplement - we keep only questions that refer to objects within the 18 categories and in the question use the coarse category as the word). Table II indicates that on the restricted dataset, the 2D method is better at answering color questions while 3D features help with questions about height and size, and in the *location* questions where spatial information is needed to determine the answer. This reaffirms our observations from the larger dataset.

VI. Conclusion

In this paper, we propose 3DVQA, which extends the traditional 2D VQA to 3D point clouds. We created a synthetic question-answer dataset based on real-world scans from ScanNet and conducted a series of experiments on baseline models. Currently, our work is limited to synthetically generated questions and answers, with noisy reconstructions (e,g, missing surfaces and parts, low resolution, poor lighting, etc). Nevertheless, we believe our work can be a good starting point for future researchers who are interested in 3DVQA.

Acknowledgment

We thank Weilian Song, Ekaterina Fedluyk and Gleb Kumichev for fruitful discussions. This work was supported by the Canada CIFAR AI Chair program and an NSERC Discovery Grant. This research was enabled in part by support provided by WestGrid (https://www.westgrid.ca/) and Compute Canada (www.computecanada.ca).

References

[1] P. Achlioptas, J. Fan, R. Hawkins, N. Goodman, and L. J. Guibas, "ShapeGlot: Learning language for shape differentiation," in *ICCV*, 2019.

[2] P. Achlioptas, A. Abdelreheem, F. Xia, M. Elhoseiny, and L. Guibas, "Referit3D: Neural listeners for fine-grained 3D object identification in real-world scenes," in *ECCV*, 2020.

[3] A. Anand, E. Belilovsky, K. Kastner, H. Larochelle, and A. Courville, "Blindfold baselines for embodied QA," *arXiv preprint arXiv:1811.05013*, 2018.

[4] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *CVPR*, 2018.

[5] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "VQA: Visual question answering," in *ICCV*, 2015.

[6] D. Azuma, T. Miyanishi, S. Kurita, and M. Kawanabe, "ScanQA: 3D question answering for spatial scene understanding," *arXiv preprint arXiv:2112.10482*, 2021.

[7] D. Z. Chen, A. X. Chang, and M. Nießner, "ScanRefer: 3D object localization in RGB-D scans using natural language," in *ECCV*, 2020.

[8] D. Z. Chen, A. Gholami, M. Nießner, and A. X. Chang, "Scan2Cap: Context-aware dense captioning in RGB-D scans," in *CVPR*, 2021.

[9] K. Chen, C. B. Choy, M. Savva, A. X. Chang, T. Funkhouser, and S. Savarese, "Text2shape: Generating shapes from natural language by learning joint embeddings," in *ACCV*, 2018.

[10] S.-H. Chou, W.-L. Chao, W.-S. Lai, M. Sun, and M.-H. Yang, "Visual question answering on 360deg images," in *WACV*, 2020.

[11] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *CVPR*, 2017.

[12] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *CVPR*, 2018.

[13] S. Ebrahimi Kahou, V. Michalski, A. Atkinson, A. Kadar, A. Trischler, and Y. Bengio, "FigureQA: An annotated figure dataset for visual reasoning," *arXiv preprint arXiv:1710.07300*, 2017.

[14] D. Geman, S. Geman, N. Hallonquist, and L. Younes, "Visual Turing test for computer vision systems," *Proceedings of the National Academy of Sciences*, vol. 112, no. 12, pp. 3618–3623, 2015.

[15] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi, "IQA: Visual question answering in interactive environments," in *CVPR*, 2018.

[16] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko, "Learning to reason: End-to-end module networks for visual question answering," in *ICCV*, 2017.

[17] D. Hudson and C. D. Manning, "Learning by abstraction: The neural state machine," in *NeurIPS*, 2019.

[18] D. A. Hudson and C. D. Manning, "Compositional attention networks for machine reasoning," in *ICLR*, 2018.

[19] ——, "GQA: A new dataset for real-world visual reasoning and compositional question answering," in *CVPR*, 2019.

[20] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning," in *CVPR*, 2017.

[21] J. Johnson, B. Hariharan, L. Van Der Maaten, J. Hoffman, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Inferring and executing programs for visual reasoning," in *ICCV*, 2017.

[22] K. Kafle, B. Price, S. Cohen, and C. Kanan, "DVQA: Understanding data visualizations via question answering," in *CVPR*, 2018.

[23] D. H. Kim, E. Hoque, and M. Agrawala, "Answering questions about charts and generating visual explanations," in *Conference on Human Factors in Computing Systems*, 2020.

[24] J.-H. Kim, K.-W. On, W. Lim, J. Kim, J.-W. Ha, and B.-T. Zhang, "Hadamard product for low-rank bilinear pooling," in *ICLR*, 2017.

[25] J.-H. Kim, J. Jun, and B.-T. Zhang, "Bilinear attention networks," in *NeurIPS*, 2018.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[27] L. Kochiev, "Neural state machine for 2D and 3D visual question answering," Master's thesis, Simon Fraser University, 2021.

[28] C. Kong, D. Lin, M. Bansal, R. Urtasun, and S. Fidler, "What are you talking about? text-to-image coreference," in *CVPR*, 2014.

[29] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International journal of computer vision*, vol. 123, no. 1, pp. 32–73, 2017.

[30] J. Lei, L. Yu, M. Bansal, and T. L. Berg, "TVQA: Localized, compositional video question answering," *arXiv preprint arXiv:1809.01696*, 2018.

[31] Z.-N. Li, M. S. Drew, and J. Liu, *Fundamentals of multimedia*. Springer, 2004.

[32] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," in *NeurIPS*, 2016.

[33] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *NeurIPS*, 2019.

[34] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi, "OK-VQA: A visual question answering benchmark requiring external knowledge," in *CVPR*, 2019.

[35] M. Mathew, D. Karatzas, and C. Jawahar, "DocVQA: A dataset for VQA on document images," in *WACV*, 2021.

[36] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[37] M. Prabhudesai, H.-Y. F. Tung, S. A. Javed, M. Sieb, A. W. Harley, and K. Fragkiadaki, "Embodied language grounding with implicit 3D visual feature representations," in *CVPR*, 2020.

[38] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *CVPR*, 2017.

[39] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *ICCV*, 2019.

[40] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *NeurIPS*, 2017.

[41] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[42] K. J. Shih, S. Singh, and D. Hoiem, "Where to look: Focus regions for visual question answering," in *CVPR*, 2016.

[43] H. Tan and M. Bansal, "LXMERT: Learning cross-modality encoder representations from transformers," in *EMNLP*, 2019.

[44] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler, "MovieQA: Understanding stories in movies through question-answering," in *CVPR*, 2016.

[45] D. Teney, P. Anderson, X. He, and A. Van Den Hengel, "Tips and tricks for visual question answering: Learnings from the 2017 challenge," in *CVPR*, 2018.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[47] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner, "RIO: 3D object instance re-localization in changing indoor environments," in *ICCV*, 2019.

[48] J. Wald, H. Dhamo, N. Navab, and F. Tombari, "Learning 3D semantic scene graphs from 3D indoor reconstructions," in *CVPR*, 2020.

[49] P. Wang, Q. Wu, C. Shen, A. v. d. Hengel, and A. Dick, "Explicit knowledge-based reasoning for visual question answering," *arXiv preprint arXiv:1511.02570*, 2015.

[50] E. Wijmans, S. Datta, O. Maksymets, A. Das, G. Gkioxari, S. Lee, I. Essa, D. Parikh, and D. Batra, "Embodied question answering in photorealistic environments with point cloud perception," in *CVPR*, 2019.

[51] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *ECCV*, 2016.

[52] X. Yan, Z. Yuan, Y. Du, Y. Liao, Y. Guo, Z. Li, and S. Cui, "CLEVR3D: Compositional language and elementary visual reasoning for question answering in 3D real-world scenes," *arXiv preprint arXiv:2112.11691*, 2021.

[53] S. Ye, D. Chen, S. Han, and J. Liao, "3D question answering," *arXiv preprint arXiv:2112.08359*, 2021.

[54] R. Zellers, Y. Bisk, A. Farhadi, and Y. Choi, "From recognition to cognition: Visual commonsense reasoning," in *CVPR*, June 2019.

[55] Y. Zhu, C. Zhang, C. Ré, and L. Fei-Fei, "Building a large-scale multimodal knowledge base system for answering visual queries," *arXiv preprint arXiv:1507.05670*, 2015.

[56] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei, "Visual7w: Grounded question answering in images," in *CVPR*, 2016.