A balanced Counting Visual Question Answering dataset

Aya Nuseir[1], Moritz Vannahme, and Marc Ebner[1]

University of Greifswald, Institute of Mathematics and Computer Science, Germany,
{s-aynuse,marc.ebner}@uni-greifswald.de,moritzav98@gmail.com

Abstract. One of the goals of artificial intelligence is to create a machine that can answer arbitrary questions about an image. This field is known as visual question answering (VQA). A Counting VQA is a subfield that can answer counting questions about an image. The current existing benchmarks for Visual Question Answering especially for the counting problem suffer from bias issues. To solve this issue we developed a synthetic VQA dataset specialized for the counting problem. We have created a generator that automatically combines various 3D models into a checkerboard pattern. The generated image is similar to a Where-is-Waldo problem. Using this generator we created an extensive dataset to help in analysing the VQA general network architecture and the VQAv2 dataset. The main characteristic of this dataset is that it is a balanced dataset; where each object has identical or at least similar odds of showing up any number of times. .[1]

Keywords: Visual Question Answering, counting, synthetic, bias, balanced.

1 Introduction

Visual question answering is recognized as multimodal machine learning, that combines computer vision and natural language processing. The standard structure of VQA is based on using images and questions as inputs and producing answers as outputs. The progress made on computer vision and natural language processing was utilized recently in VQA. Regardless, VQA still struggles with the counting task, which requires understanding the characteristics of the desired object, finding it multiple times in the image then adding all instances up.

Currently, counting tasks like the other VQA tasks have several limitations, the most common one is the problem of language bias. The available models may generate or infer the answer based on the language data without the need for visual data. Consequently, that will affect the performance of the VQA model badly. An example is the question 'What colour is the banana?' Even without ever having seen the picture, this question can be answered with relative certainty with 'yellow'.

[1] https://github.com/ASNuseir/A-balanced-Counting-Visual-Question-Answering-dataset

Theoretically, the banana question could be answered with many other colours besides 'green', 'yellow', and 'brown'. However, such answers are highly unlikely. This leads to a misrepresentation of the ability of an artificial neural network to determine colors when compared to a baseline based using random guessing across all colours known to the network. The same issue arises with counting questions, where we found most of the generated answers range between the numbers 1-4. In this paper, we present a synthetic dataset analysing the ability of VQA systems in the context of counting problems, inspired by the work of the CLEVR dataset [4], the idea is to generate the images and questions automatically. The main characteristic of our dataset is that all possible answers for all possible counting questions occur equally often.

In section 2, we discuss some of the existing visual question-answering datasets. Section 3, describes the general structure of how our dataset was generated. Then, in section 4, we represent the image generation process. In section 5, we explain the question-and-answer generation process. Finally, the last section describes two VQA models we adopted in our experiments.

Fig. 1: A sample image, objects to be counted/found are circled.

2 Related Work

Various VQA datasets have been developed [12,13,4,5,6], and the purpose behind this huge number of VQA datasets is to increase the abilities of the VQA model

to be used in real-world settings like fine-grained recognition, object detection, counting commonsense reasoning [3].

The VQAv1 dataset was created by Antol et al. and developed together with with the VQA network [5]. Later Goyal et al [8] worked on the Antol et al dataset to make it a balanced dataset by collecting complementary images; where every question is associated with a pair of similar images that result in two different answers to the question. Both versions of the previously mentioned datasets took their images from the COCO dataset [1]. It contains a wide range of photos of various situations and different objects. The questions and answers were collected via Amazon Mechanical Turk. Initiated by the work of Zhang et al.[2], the goal of improvement was to record a better balance of answers. In the first VQA dataset [5], there was a very large number of questions, that had only one possible or at least had one strong dominating answer. In the course of the development of the VQAv2 dataset, an attempt was made to fix this. The focus of this correction was yes/no questions. The largest visual question-answering dataset has been created by Krishna et al. [6]. This dataset contains about 108,249 images from YFCC100M and the COCO dataset. It has two types of question/answer pairs for each image: freeform question/answers that are based on the entire image and region-based question/answers that are based on selected regions of the image. The dataset includes six different types of questions for each image: what, where, how, when, who, and why.

Trott et al. [7] have created a data set to evaluate counting systems. Questions and answers were selected from the union of the data set VQA 2.0 [8] and the Visual Genome QA [6]. A ground-truth answer for all questions is a number between 0 and 20.

Acharya et al. [9] have created the TallyQA dataset. This data set is the world's largest dataset for open-ended counting. This data set is split into a training set and two test sets. Questions were gathered using Amazon's Mechanical Turk (AMT) but also imported from other datasets. Their training set contains 249,318 questions and 132,981 images. A simple test set contains 22,991 questions and 18,411 images. The second test set includes 15,598 questions and 14,051 images.

What makes our dataset differ from the previous counting dataset mentioned above([7] and [9]), is that we adopted the binary question(yes/no answer) plus the "How many?" and developed our own images artificially. The reasons for adopting binary question type are: relevant semantic information is available in the question and this type of question is easier to evaluate [2].

3 Generator Structure

In this section, the focus is made on explaining the general structure of the dataset generator. The generated dataset is composed of three parts: Images, questions and answers. Each question is associated with one answer (and vice-versa), but one image may be associated with multiple question-answer pairs. The generator is structured using multiple nested loops, which iterate (from the

outside in) over the objects that are to be counted, the number of times that object occurs in the image, the number of different compositions and the number of camera positions. The structure is shown in Figure 2

```
Initialization
O ← Number of objects −1
C ← Number of constellations −1
for i ← 0 to O do
    for j ← 0 to 10 do
        for k ← 0 to C do
            Generate next constellation
            for l ← 0 to 2 do
                if StaticCamera == True then
                    l ← 3
                else
                    Rotate camera
                end if
                Render image
                Capture image
                Write question(s)
                Write annotation(s)
            end for
        end for
    end for
end for
```

Fig. 2: Structure of the main loop

The structure of the dataset is defined by the structure of the loop given in Figure 2. Image files are named according to the scheme: IOOOAACCP.png, the pattern is dataset-ID (1 digit), Object-ID (3 digits), number of object occurrences (2 digits), composition number (2 digits) and camera position (1 digit). All values are indexed from 0.

The number of possible camera positions and compositions can be modified by program inputs. For each image, several question/answer pairs are generated which are explained in section 5.

4 Image Generation

The process of generating images is based on creating a chessboard structure of tiles and distributing objects to the individual tiles. The resulting chessboard with objects placed on top of it is illuminated and photographed to obtain the image. This process is divided into four categories:

- The objects which the network has to count (one object per image).
- The ground or floor they stand on.

– The composition of the individual object+tile pairs to the chessboard.
– The camera and light control.

4.1 Object

All objects are provided in the form of 3D computer graphics files (all formats accepted by OpenSceneGraph are accepted). They were taken from the SketchUp[14] model library as part of this project. Most objects are models created by SketchUpLabs. All objects are rescaled so that they have the same size.

Objects can be coloured randomly using the following colours: Red (1,0,0), Green (0.5,1.0), Blue (0,0,1) Purple (0,1,1), White (1,1,1) and Gray/Black (0.5,0.5,0.5). All RGB values have the range $[0, 1]$.

The colouring of objects is built in to exclude the possibility that the neural network is able to identify the object based on colour alone. If colour could be used as a cue for object identification, then location, identification and counting of objects would become significantly easier. Another option to achieve the same effect is to colour the floor tiles in the colours of the objects which is discussed in the next section. Objects can also be rotated and scaled. All object modifiers are shown in Figure 3.

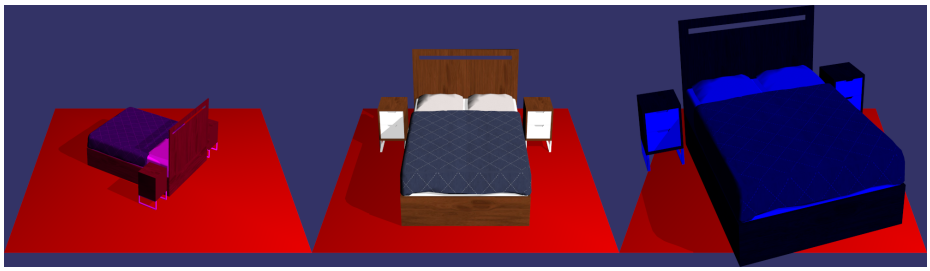

Fig. 3: An object shown with different scaling, rotation and colour

4.2 Floor

The floor is composed of individual tiles. Each object is positioned on one tile. The tiles are arranged in a chess-board-pattern. For most purposes, the size of the object is chosen such that the length of the tile matches the diameter of the object's bounding sphere. The camera position is chosen such that the completed chessboard has a constant size in the image.

Two approaches were used for the colouring of the tiles: In the first approach, the tiles were laid out in a checkerboard pattern with white (1, 1, 1) and orange (1,0.5, 0). The colour choice was made to provide a strong contrast. In the second approach, the floor tiles can be coloured randomly in the characteristic colours

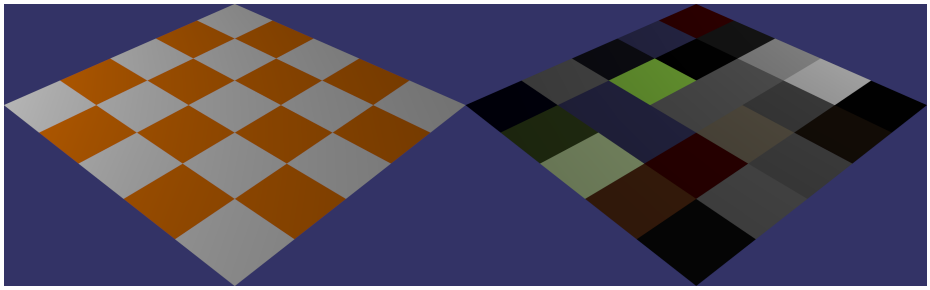

Fig. 4: Comparison of two-color and multicolor floor.

of the objects that are shown. This renders the object color useless for object identification because the same color can be found at random locations in the image. However, to use this mode, the characteristic object colours must first be extracted and made available to the generator.

4.3 Scene Composition

For each image, we have one object that has to be counted. The remaining objects are not to be counted. The number of tiles per axis is limited to positive odd integers. The camera is centered on the midpoint of the central tile. The objects are either distributed randomly or with a fixed position. Each tile is assigned an object at random. It is possible to exclude specific objects. If n is the number to be counted then n different tiles are randomly chosen and the object is placed on the selected tiles. Optionally, the main object colors may be used to randomly color the floor tiles. Each object is rotated by a random value between 0 and 2π around the y-axis.

We also implemented the option of having fixed object positions for the objects that need to be counted. Instead of randomly distributing a certain number of objects at run time, we obtain a fixed distribution of objects for each number from 0 to 10 once. This distribution is then read in for each setup to be generated. It is important to note that the same seed needs to be passed to both the training and the validation set (identical sets are prevented through the use of the validation flag). If this is not done, object positions will differ. The distribution of the remaining objects i.e. objects not to be counted, is determined at random.

4.4 Camera and Lighting

Both the camera and the light source can be rotated around the y-axis on a circle with radius $r = 2 * b * d$, where b is the length of the board in tiles and d is the length of a tile. Both are placed at a height of $b * d$ above the chessboard and in the x-z-plane. They are each rotated around the y-Axis by a random amount between 0 and 2π whenever they are updated. The light is

always changed whenever a new composition is created. The image is rendered three times using different camera positions if the camera is not static. If the camera is set to a static position, then the camera is positioned such that it faces the front right corner.

The scene is illuminated using white light without any attenuation. The ambient component of the light is also white but its intensity can be modified. It defaults to 1. The specular component of the light is set to an intensity of 0. The ambient bias of shadow casting can also be modified. It defaults to $(0.8, 0.2)$, where $(1, 0)$ is no shadow and $(0, 1)$ means each shadow is pitch black.

5 Question and Answer Generation

The generated data set has three types of questions: counting questions, yes/no counting questions and yes/no presence questions (answer should be affirmative if the object is present and dissenting otherwise). There are two possible distributions of these questions. In the standard version, the counting question, as the main subject of this work, was generated for each image and the other two questions for every third image (once per setup with a moving camera, once per three setups with a fixed camera). Since the yes/no questions are generated in pairs which leads to 3 counting questions to 2 yes/no counting questions to 2 yes/no presence questions, i.e. a (3:2:2) split. Alternatively, it is possible to generate yes/no questions for each image. With that the split is 1:2:2. This approach records 2.1 times as many unique picture+question combinations as the standard version.

The purpose of adding non-counting questions is to force the network to evaluate the question beyond keyword detection, as well as to give the network an easier question so that it may focus on learning object recognition. In order to be able to generate the natural language questions, the file specifying the model locations must also specify the singular and plural of the object in natural language.

5.1 Counting Question

Counting questions have the form 'How many [objects] are in the image?'. The question type is 'How many' and the answer is given by the state of the main loop at the time of its generation.

5.2 Binary Counting Question

Binary Amount Questions have the form 'Are there [X] [objects] in the image?'. The question type is 'Are there'. These questions are generated as a yes/no pair. For both questions, the object is the same as that used in the counting question and thus set by the main loop. The number X is also given by the state of the loop, with the amount given by the loop being A: For the answer yes,

$$X = A \tag{1}$$

, for the answer no:

$$X = A + 6 \bmod 11 \tag{2}$$

This offset ensures that each amount+object combination is answered with yes or no with the same probability.

5.3 Binary Presence Questions

Binary Presence Questions have the form 'Is there a [object singular] in the image?'. The question type is 'Is there'. We resolve the ambiguity between 'exactly one' and 'at least one' in favour of the latter. The former is already covered by the binary counting question 'Are there 1 [objects] in the image?'. Unlike the other two question categories, we did not balance this one perfectly vis-a-vis the yes/no splits for individual objects. Instead, the generation process is as follows:

- If a question should be answered with 'yes', we pick a random position on the currently active composition and ask for the object at that position.
- If a question should be answered with 'no', then an object is selected from a set of objects that are not placed anywhere else. We generate a list of items at the initialization of the generator. These objects are excluded from the set of objects that are used as a distraction. One of these objects will be placed on the board.

This method does not guarantee a perfectly balanced data set but almost achieves this as the data set gets larger. If uncounted objects are deactivated, this question type will not be generated.

From Table 1 we can see clearly the distribution for questions of type "Is there" and "Are there".The distribution is the same for both question types but the difference is a kind of split that is adopted. However, the "How many" question answers distribution for both splits is equal to 3366 and distributed equally between the values between 0 to 10 (306 for each numbered answer)

Table 1: The distribution of the binary answers for both splits

Question Type	**1:2:2 Split**		**3:2:2 Split**	
	yes	no	yes	no
Is there	3366	3366	1122	1122
Are there	3366	3366	1122	1122

6 VQA methods

Through this paper, we intend to study the consequence of the unbiased dataset on VQA models by developing a synthetic balance VQA. Our experiments are based on testing the most well-known available VQA models; the models that

are based on using the typical VQA model without attention, and the attention mechanism.

Starting with the [5] models; They built three models to test their dataset, for the images two procedures were used. Both approaches were based on using the VGG algorithm, the first used the activations from the last hidden layer as 4096-dim image embedding. The second procedure normalized the activations from the last VGG hidden layer. For the Questions three strategies were adopted; the first was based on using a Bag of words, and the second used LSTM with one hidden layer to obtain 1024-dim embedding. Every word in the question is encoded with 300-dim embedding by a fully connected layer and tanh non-linearity which is then fed to the LSTM. The third method used LSTM with two hidden layers to obtain 2048-dim embedding. They combined the images with questions using Multi-Layer Perceptron (MLP). In this work, we tested the third model.

The second model is[10], they proposed their hierarchical question-image co-attention model. Their hierarchical model is composed of two novel components question hierarchy and co-attention. The question hierarchy generates three level representations: word level, phrase level and question level embeddings. For the co-attention two mechanisms were implemented; parallel co-attention and alternating co-attention. During our experiments to examine these two models with the balanced dataset, we adopted the following configurations in creating our dataset: a coloured object with a two-colour floor using the 1:2:2 split. The result shows that the performance of the [5]. model outperform the performance of [10]; where the accuracy of the [5] model is equal to 45.09% and the accuracy of the [10] is equal to 25.76%. Even though Lu et al.[10] adopted model can attend to different regions of the image plus different fragments of the question but fails to focus only on the specific or targeted part of an image and textual data. In consequence, this model was unable to represent the correct attention.

7 Conclusion

We have developed a generator to automatically generate counting questions for visual question answering. The generator can be used to generate extensive and balanced datasets, which is often not the case for real-world datasets. For this work, the generator was developed in a way that can generate such datasets automatically based on various parameters. We also took care that characteristic object colours may not be used for object recognition because such colors also occur elsewhere in the image. Thus the generator and this work offer a good starting point to further one's work on analyzing systemic errors in Visual Question Answering, in particular in the area of counting.

References

1. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In Computer

Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13 (pp. 740-755). Springer International Publishing.
2. Zhang, P., Goyal, Y., Summers-Stay, D., Batra, D. and 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).
3. Cadene, R., Dancette, C., Cord, M. and Parikh, D., 2019. Rubi: Reducing unimodal biases for visual question answering. Advances in neural information processing systems, 32.
4. Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C. and 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).
5. Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L. and Parikh, D., 2015. Vqa: Visual question answering. In Proceedings of the IEEE international conference on computer vision (pp. 2425-2433).
6. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A. and Bernstein, M.S., 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. International journal of computer vision, 123, pp.32-73.
7. Trott, A., Xiong, C. and Socher, R., 2017. Interpretable counting for visual question answering. arXiv preprint arXiv:1712.08697.
8. Goyal, Y., Khot, T., Summers-Stay, D., Batra, D. and 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).
9. Acharya, M., Kafle, K. and Kanan, C., 2019, July. TallyQA: Answering complex counting questions. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 8076-8084).
10. Lu, Jiasen and Yang, Jianwei and Batra, Dhruv and Parikh, Devi, 2016. Hierarchical question-image co-attention for visual question answering. Advances in neural information processing systems
11. Teney, D., Anderson, P., He, X. and Van Den Hengel, A., 2018. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4223-4232).
12. Gao, H., Mao, J., Zhou, J., Huang, Z., Wang, L. and Xu, W., 2015. Are you talking to a machine? dataset and methods for multilingual image question. Advances in neural information processing systems, 28.
13. Zhu, Y., Groth, O., Bernstein, M. and Fei-Fei, L., 2016. Visual7w: Grounded question answering in images. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4995-5004).
14. SketchUpLabs,2021,https://3dwarehouse.sketchup.com/collection/5bc34341-a87c-47cf-9a9f-0a6f3db6c916/Library-Models