Inductive Biases for Low Data VQA: A Data Augmentation Approach

Narjes Askarian
Dept. of Data Science and AI
Monash University

Ehsan Abbasnejad
Australian Institute for Machine Learning
The Univ. of Adelaide

Ingrid Zukerman and **Wray Buntine** and **Gholamreza Haffari**
Dept. of Data Science and AI
Monash University

Abstract

Visual question answering (VQA) is the problem of understanding rich image contexts and answering complex natural language questions about them. VQA models have recently achieved remarkable results when training on large-scale labeled datasets. However, annotating large amounts of data is not feasible in many domains. In this paper, we address the problem of VQA in low labeled data regime, which is under-explored in the literature. We take a data augmentation approach to enlarge the initial small labeled data in order to inject proper inductive biases to the VQA model. We encode the additional inductive biases in the questions by producing new ones taking advantage of the image annotations. Our results show up to 34% accuracy improvements compared to the baselines trained on only the initial labeled data.

1 Introduction

VQA is the task of answering questions about visual contexts. It has attracted significant attention and achieved impressive results [11, 26, 2]. This success is partly due to large-scale labeled datasets [10, 4, 27, 13]. Relying on large-scale labeled datasets is not realistic in many settings due to the infeasibility of collecting such data. In addition, the objective of VQA in a sense is rather ambitious as there are potentially infinite number of questions to be asked about an image. As such, we argue that VQA is generally a low-data problem. In the absence of sufficient data, current VQA systems do not maintain their high performance (e.g., see Figure 1).

We consider VQA in low labeled data scenarios. We investigate the features of a VQA task which necessitate a lot of labeled training data. One of those features is understanding complex questions about rich visual contexts. VQA datasets mostly contain complex questions where a learner requires identifying multiple objects and understanding their relationships. Understanding a question and capturing an image scene is a lot easier when the learner has access to a large amount of labeled data. The model eventually captures the complexity after seeing a large variety of data when training on a large-scale dataset. However, complex relationships are challenging to learn from small datasets.

In this paper, we improve the generalisation of VQA models by injecting inductive biases so that the model can explicitly have access to them in a data efficient manner. The inductive biases are particularly of high significance to the questions since they heavily impact the answers in VQA. An inductive bias that a typical learner acquires by training on natural language tasks is related to the inherent compositionality of the human language, e.g. a complex sentence can be understood by understanding its simpler chunks. The resulting chunks are normally easier to capture the meaning, and provide a powerful foundation for understanding complex sentences. Inspired by the fact that a complex question can be learned on the basis of the basic concepts, we hypothesized that augmenting the training set of complex questions with simpler questions will help the model. The notion of simplicity of a question can be defined based on different criteria, including syntactic and semantic dimensions. In the VQA context, we consider simplicity as the number of reasoning steps required for answering a question. Thus, the simplest possible question requires identifying a single object and reasoning about it. We particularly include simpler questions that if learned could lead to better representations in the VQA model.

We take a data augmentation approach and enlarge the initial small training set by automatically generating simple question-answer pairs for images. We hypothesise basic concepts can be learned from simple questions, enabling the model to better learn the structure of more complex questions.

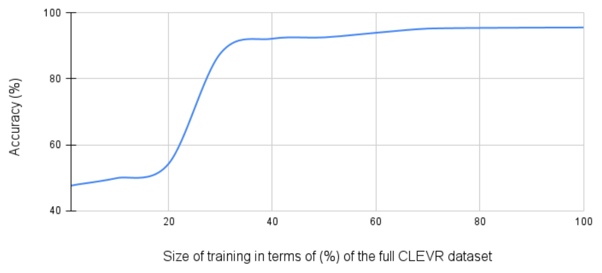

Figure 1. Accuracy of vanilla training of the *execution engine* on CLEVR `val` where trained on different-sized random subsets of the CLEVR `train` set.

Data augmentation strategies have proven to be particularly useful in a variety of computer vision applications, including image classifications [17]. Not only they can be helpful to overcome the problem of insufficient labeled data, they are also used to reduce overfitting and class imbalance problems [23]. Current data augmentation techniques use data warping or oversampling to increase the size of the training dataset [20, 23]. Data warping is a technique for transforming data while maintaining its labels. Typically the examples are transformed by geometric and color transformations, random erasing, neural style transfer, and adversarial training.

Data augmentation in VQA is under-explored due to the challenge of correctly preserving the semantic relation of the <*image, questions, answer*> triplet during transformation. Geometric transform, and random cropping of the image cannot guarantee to preserve the answer. For instance, the answer to *"what color is the thing on the left side of the cube?"* may be flipped if the image is vertically transformed. Random cropping can result in missing the number of objects when counting to answer a *how many* question.

Our proposed data augmentation method automatically generates simple questions. Our method only requires having access to shallow annotations of an image scene and does not use any additional labeled data. The annotations give some information about the appearance of the objects in the image. The answers to the questions are also automatically generated at no cost in human effort. The method is generic and is applicable to any VQA task given the scene information available in many current VQA datasets. The experimental results and analysis demonstrate that our method is effective in improving VQA performance, and significantly improves the performance by up to 34% accuracy compared to training on only the initial labeled data.

2 Related Work

2.1 Data Augmentation

It is widely accepted that using larger datasets in training yields stronger DNN models. However, in many domains such as medical applications, limited datasets are of common challenges due to the manual effort of collecting and annotating data. One of the main problems of training on insufficient data particularly in deep learning is overfitting. Many methods try to solve this problem by focusing on the model's architecture [9] or regularization methods [24, 12]. In contrast to them, data augmentation tackles overfitting from the root, *i.e.* by manipulating the training set. This lies on the idea that data augmentation can extract more information from the original dataset. It generally consists of artificially increasing the size and the diversity of training examples by automatically creating additional "augmented" data based on the available data. Besides the problem of limited datasets, data augmenting has also been considered in the class imbalance distribution problem.

Data augmentation has received most of its attention in Computer Vision (CV). Many widely-used augmentation techniques are introduced to improve the generalization ability of Convolutional Neural Networks (CNN) models in vision tasks such as image classification, object detection, and image segmentation. These techniques are most applicable to images but not other types of data. In other areas such as text processing or more specifically Natural Language Processing (NLP), data augmentation has not been well explored. Although VQA is a multi-modal problem involves in both image and text, due to some restrictions it cannot easily benefit the data augmentation advances in both image and text modalities. The ensuing sections elaborate on the related studies on data augmentation in CV and NLP as a foreground for VQA data augmentation while we briefly explore the data augmentation studies in VQA later.

2.2 Data Augmentation in VQA

Data augmentation for VQA is covered in fewer works, for example [15, 19, 5]. Ray *et al.* [19] leverages knowledge in the Visual Genome dataset [16] to create QA pairs that quantitatively evaluate the consistency of a VQA model. The idea is that if a model answers *"red"* to *"what color is the ball?"*, it should answer *"yes"* if asked *"is the ball red?"* to correctly preserve the notions of entailment. They automatically create a set of logically consistent QA pairs from a source QA pair and also collect a human-annotated set of consistent QA pairs based on common-sense, *e.g. "is the ball the same color as a tomato?"*.

Shah *et al.* [22] presents a cyclic-consistent training strategy in which the model is trained to predict a consistent answer for a source question and its rephrased version. To enhance the model's robustness against semantic visual changes, their method uses a GAN-based re-synthesis methodology to automatically eliminate items. Agarwal *et al.* [1] use data augmentation to enhance the model's robustness against semantic visual modifications. They employ a GAN-based re-synthesis approach to automatically eliminate

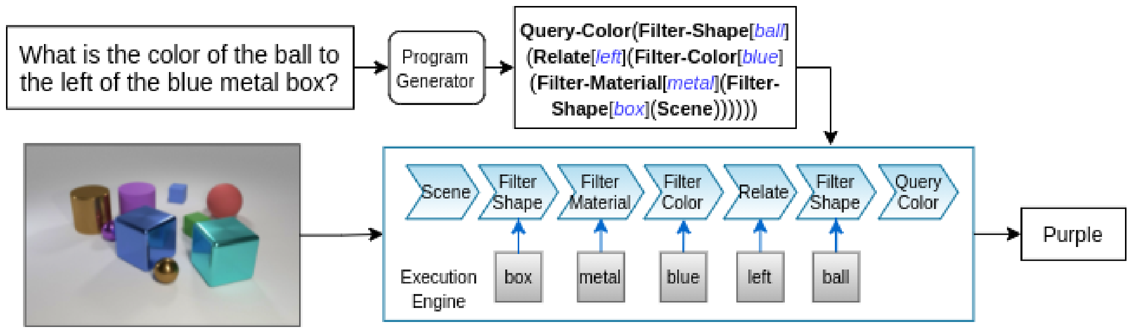
What is the color of the ball to the left of the blue metal box?
Program Generator
Query-Color(Filter-Shape[ball](Relate[left](Filter-Color[blue](Filter-Material[metal](Filter-Shape[box](Scene))))))
Scene
Filter Shape
Filter Material
Filter Color
Relate
Filter Shape
Query Color
box
metal
blue
left
ball
Execution Engine
Purple

Figure 2. An overview of a modular VQA system consisting of a program generator and an execution engine.

the objects.

2.3 Using Scene Information in VQA

Although many studies use scene graphs in answering questions [26, 11, 6], a few works attempt to exploit such information for generating questions. [5] proposed to generate contrast questions for the GQA dataset. Contrast sets proposed in [8] aim to perturb a small subset of the test instances in a meaningful way typically change the ground truth label in order to evaluate a model's true capabilities. For instance, a contrasting example for QA pair *"is there a fence near the puddle? yes"* is automatically generated using the scene graph as *"is there a wall near the puddle? no"*. [19] consider scene graphs in creating consistent Question-Answer (QA) pairs that can be derived based on simple notions of logic. For instance, they create a QA like *"is the sofa black? No"* according to the relational triplet of <*sofa, is, white*> derived from the scene graph and using a simple logic like *"a white thing is not black."* Similar to the above work we automatically generate the QA pairs using scene knowledge but in contrast to them, our proposed method does not rely on the relationships in a scene graph or any logic, it rather uses the attributes of the objects in images which can be easily generated by an object detection model.

3 VQA Model

Current VQA approaches mostly obtain the embedding of questions and images and map them into a cross-modal common space in which the answers are predicted. Such approaches treat a question holistically thus the reasoning process is unclear [25, 18, 21].

In contrast, modular approaches plan the reasoning process by semantically parsing the question [3, 14]. A modular VQA model generally consists of two main components: (1) a *program generator* $\mathcal{G}$ that takes a question and generates the reasoning plan called program p, (2) an *execution engine* $\mathcal{E}$ that use the program as a layout to combine the modules. Modules are small neural networks treated single-task functions that are able to fulfill complex jobs when combining into a larger network. The resulting network predicts the answer by processing the input image. Figure 2 depicts an overview of the modular VQA system in this work.

In this paper, we use the modular model proposed by [14]. The authors demonstrate that the *program generator* can produce accurate programs when training on less than 4% of the possible programs. Therefore, we disregard $\mathcal{G}$ and focus on evaluating the *execution engine* in a low data setting. We train $\mathcal{E}$ on different training sets where each set is a subset of the full dataset. Note that we use the ground truth program and image pair as the input to $\mathcal{E}$ in all experiments. The results shown in Figure 1 reveal that the accuracy of predicting answers dramatically drops when training on small sets.

4 Our Data Augmentation Approach

This section explains our approach for automatically generating simple questions using superficial information from the image scene. We use only the basic attributes of the objects present in an image including size, color, material, and shape; and disregard object's coordination and spatial relationship between objects.

4.1 The Notion of Simplicity of Questions

The first objective of this work is to generate simple questions. As discussed earlier, simplicity is an abstract and relative concept that can be defined in different ways. One may view it from the linguistic point and incorporate lexical and semantic criteria such as the length of the questions. In order to focus on simple questions, we inevitably must formulate simplicity as a measurable criterion. Based on the general focus of this work which is reasoning using compositionality, we translate simplicity as the number of reasoning steps required for answering a question. In this sense, a question

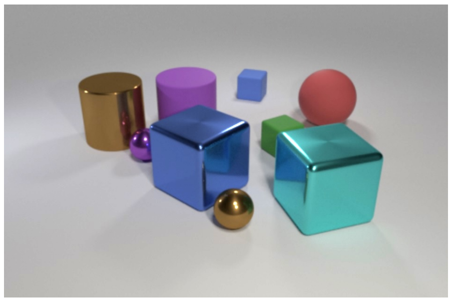
Figure 3. The question *"what color is the object behind the blue cube?"* is ambiguous since there are two blue cube in the image. To create an unambiguous question, the target object must be uniquely identifiable by the referring expression in the question. A unique attribute combination assures that the expression refers to a unique object such as *"red"* or *"red sphere"* that both refer to the red sphere at the top right corner.

with a longer reasoning chain is considered a more complex question relative to other questions with a shorter reasoning chain.

Since we aim to generate simple questions, let us first specify the characteristics of the simplest possible question that can be created. According to the above definition, the simplest question is the one with the shortest reasoning chain. One of the most important aspects of comprehending questions is identifying the target object. In many cases, an object is detectable through its spatial relationship with other objects, *e.g*. *"the sphere behind the green box"* in Figure 3. Such cases involve locating multiple objects and therefore the reasoning chain would be more than one. As a result, a demand for finding a single object in the image can be taken as the first character of the simplest questions.

A second factor that makes a reasoning chain longer, even if it requires detecting a single object, is a long referring expression to the target object. The referring expression is the phrase by which we can uniquely locate the object in the image. Every major word in a referring expression will be translated to a filtering step in the reasoning chain. For instance, the expression *"red sphere"* is converted to *"1) filtering the red color pixels, 2) filtering the sphere shapes on top of the result of the first step"*. To keep a question as simple as possible, the referring expression needs to be short.

4.2 The Notion of Ambiguity of Questions

The objective here is to generate questions automatically without any use of human effort. Generally speaking, human intervention in question generation may be required for two main purposes: either monitoring the quality of generated questions or producing answers. Question quality is a broad concept that embraces meaningfulness, unambiguity in addition to linguistic factors such as grammar and fluency. Every work may narrowly define question quality according to its task and aims.

In our VQA setting, question quality narrows down to unambiguity. Since our method is a template-based question generation approach, other aspects of question quality are automatically addressed in the templates. Therefore, human intervention may be needed to filter out the ambiguous questions. According to our definition, ***ambiguity*** happens when the target object cannot be uniquely identified by the attributes stated in the question. For example, in Figure 3 the question *"what color is the object behind the blue cube?"* is ambiguous due to two reasons. Firstly the expression *"the blue cube"* can refer to either the large shiny blue cube or the small matt cube at the back. Secondly, if supposedly *"the big shiny blue cube"* was the only *"blue cube"* in the image, the questions would still be ambiguous because there is more than one object behind it whose color could be the answer. The reader should note that the task is a question-answering task, not a dialog in which the agent can ask for more clarification by responding, for instance, *"which object behind the blue cube?"*. On the other hand, an unambiguous question can be easily answered by locating the target object and looking up the required attributes among its metadata from the scene knowledge.

As a result of the discussions so far, eliminating human effort boils down to posing unambiguous questions. For this purpose, we introduce **unique attribute combinations** to guarantee that answering the generated questions does not require any human effort by ensuring that they refer to a unique object in images. The following section explains unique attribute combinations in detail whereafter we elaborate on the templates used for generating questions.

4.3 Unique Attribute Combinations

A **unique attribute combination** is defined as a referring expression consisting of a set of attributes that can uniquely identify an object in an image. For instance, a unique attribute combination for the red sphere at the top right corner of Figure 3 can be *"red"* because it is the only red object in the image and it can be uniquely identified if one refers to it as *"the red object"*. Although it is the shortest unique attribute combination for the red sphere, there are other unique combinations for this object including *"large sphere"*, *"matt sphere"*, and *"large red sphere"*. As we are interested in short questions, we only produce short attribute combinations of the lengths 1 and 2.

Now, we explain how we create unique attribute combinations. Let uac_l be the unique attribute combination of length l. We begin by creating combinations of length 1 for all objects in the image, $uac_{l=1}$ by simply comparing the values of the similar attributes in all objects of the image, *e.g*. the colors or shapes of the objects; and then record the attribute values that appear in only one object such as *"red"* in Figure 3. This process is shown in Algorithm 1 line 4–9.

A general method to create unique combinations with

Query-attribute Type		
	Template	**Example**
Question	What [attribute$_q$] is the [uac] object?	What size is the red object?
Program	$scene \rightarrow \left\{filter\text{–}[attribute_{uac^i}][[uac^i]] \rightarrow\right\}_{i=1}^{l}$ $query\text{–}[attribute_q]$	$scene \rightarrow filter\text{–}\texttt{color}[red] \rightarrow query\text{–}size$

Existential Type		
	Template	**Example**
Question	Is there a [uac] object?	Is there a red object?
Program	$scene \rightarrow \left\{filter\text{–}[attribute_{uac^i}][[uac^i]] \rightarrow\right\}_{i=1}^{l} exist$	$scene \rightarrow filter\text{–}\texttt{color}[red] \rightarrow exist$

Table 1. The question and program templates for two types of questions are used in this work along with an example for each. The top table shows the the templates and example for *query-attribute* type where the question begins with *what* and asks about an attribute value of the target object. The lower table details the *existential* type where the questions asks if the target object presents in the image.

length $l > 1$ is generating all possible combinations of length l for all objects in an image. Then comparing the combinations and removing the ones that refer to more than one object by simply eliminating the repeated combinations. However, as illustrated in Algorithm 1 line 10–19, we modify this method to reduce the computational time. We join the values in $uac_{l=1}$ to other attribute values of the corresponding object, *e.g*. we attach *"red"* as a uac_1 to *"sphere"* to create *"red sphere"* or *"red large"* where $l = 2$. This strategy produces longer unique combinations at a fast rate but at the cost of missing a number of possibilities. Once the uacs are generated, the next step is using templates for question generation. Let us now introduce our templates as well as the generation details.

Algorithm 1 Creating Unique Attribute Combinations

```
1:  I: Image list
2:  A: List of attributes i.e. {color, shape, size, material}
3:  procedure CREATE-UAC(l)
4:      #Obtaining uac where l==1
5:      for $im \in I$ do
6:          for $obj \in im$ do
7:              for $attrib \in A$ do
8:                  if $obj.attrib.value$ is unique then
9:                      $uac_1 \leftarrow obj.attrib.value$
10:     #Obtaining uac where l≥1
11:     if $l \geq 1$ then
12:         for $im \in I$ do
13:             for $obj \in im$ do
14:                 for $u1 \in im.obj.uac_1$ do
15:                     #Choosing l-1 attribute
16:                     from all obj attributes
17:                     for attrib-comb $\in \binom{obj.attibutes}{l-1}$ do
18:                         if $u1 \notin$ attrib-comb then
19:                             $uac_l \leftarrow$ Concat($u1$, attrib-comb)
```

4.4 Generating Template-based Questions

We synthesize two types of questions from scene knowledge: *query-attribute* and *existential* questions. As their names suggest, *query-attribute* questions begin with *"what"* ask about the value of an attribute of the target object, *e.g*. *"the color of the cube"* or *"the shape of the big object"*, while *existential* questions start with *"is"* and ask whether the target object presents in the image. Table 1 shows the templates along with corresponding examples. [attribute$_q$] indicates a placeholder which must be replaced with the queried attribute name that can be any of the attributes from the attribute set, *i.e.* $\{size, color, material, shape\}$; while [attribute$_{uac}$] is the attribute name being presented in the uac. [uac] indicates a uac of the desired object.

The phrase inside the braces creates a sequence of filters with the same length as uac when being repeated for each attribute value in uac and combined together. In fact the examples provided in the table use a uac with $l = 1$, *i.e.* *"red"*; however in case the uac is longer, *e.g*. *"red sphere"* then identification process of the target object is performed in a multi-step filtering. For instance, first filtering the *red* `color` and second filtering the *sphere* `shape`. This process is formalized as the following phrase: $filter\text{–}\texttt{color}[red] \rightarrow filter\text{–}\texttt{shape}[sphere]$. It is also noteworthy to mention that we let it be possible that a question asks about one of the attributes that constitute the uac, *e.g.* *"what color is the red sphere?"*, although the answer is very obvious and already appears in the question itself. In other words, the attribute that is chosen to fill the [attribute] placeholder may be one of the attributes whose value is included in the [uac].

The *existential* template can be used to produce *yes/no* questions. Replacing a target object's uac with [uac] in the template can easily result in a *yes* answer while for a question with an answer *no* an invalid attribute should be joined to uac. Any attribute value which does not match the target object is considered as an invalid attribute for it. For instance, assume *"is there a red sphere in the image?"* as a positive-answered question, then *"is there a red* ***small*** *sphere?"* will make a negative-answered question because ***small*** is an invalid attribute for the target object and hence if attached to uac creates a referring expression with no visual equivalence in the image. The invalid attribute must be selected among the attributes that don't already contribute in the uac otherwise the referring expression becomes ambiguous.

As it may be clear, to select an answer to a generated question, we simply determine the target object and draw its scene information, then pick the value of the requested attribute. In the case of the *yes/no* answers, the answer is determined by the questions generation process based on whether an invalid attribute is used or not. Let us emphasize again that utilizing uac ensures that the question refers to a unique object and as the result, the answer will be distinct so that it can be automatically selected or generated.

5 Experiments

Dataset

Although our data augmentation method is generic and applicable to many VQA datasets for generating augmented questions, we make use of the scene information from the CLEVR dataset [13]. An example of it is provided in supplementary material. This information is available in the form of objects' attributes and their spatial relationship. Our proposed method aims to use the shallow attributes of objects, thus we only consider color, size, material and shape among all other information have been provided about the scene of an image. We also use the CLEVR dataset [13] for training. It provides a training set with $70k$ images, $\sim 700k$ *(image, question, answer)* tuples. The answers come from 28 classes including 8 colors, 2 sizes, 3 shapes, 2 materials, 11 numbers as well as yes and no. Our evaluation is conducted on the `val` split, which contains $\sim 150k$ questions and $15k$ unique images.

5.1 Setup

To simulate a low-data scenario, we select four small subsets from the training set with different sizes denoted as a percentage of the full dataset. We refer to these subsets as s-CLEVR$_x$ where $x \in \{5, 10, 20, 30\}$ indicates the size of the subset, *e.g.* s-CLEVR$_{20}$ is the subset that is chosen to be as small as 20% of the full training set with $\sim 140k$ *(image, question, answer)* tuples. We use the scene information of the images from s-CLEVR$_x$ denoted as I_x for generating augmented questions.

We conducted our experiments in two stages: ***data augmentation*** and ***training***. The first stage includes extracting uacs and generating question-answer pairs using the proposed data augmentation method. The uacs are extracted from the scene information of a selected set of images (See details in §5). To emphasise simplicity, we only use uac_1 for generating *query-attribute* questions and uac_2 for *existential* questions. uac_2, as described in the preceding section, are built upon uac_1s. For each uac_1, we generate *existential* questions with a probability of $p = 0.25$ where the chance of *yes* answers equals *no* ones. We refer to the generated question-answer set as Aug^{simple}. In the second stage, we add the augmented questions to the training set to create an augmented training set, *i.e.* s-CLEVR$_x$+Aug^{simple}. We use the *execution engine* $\mathcal{E}$ described in §3 as the model in the experiments. We train $\mathcal{E}$ on the augmented training set from scratch and evaluate on `val`.

In addition to the questions, the model requires the image features to produce the answer. The image features are the output of $conv4$ of ResNet-101 [9] pre-trained on ImageNet [7]. We then test the model on the validation set and compare the results with two baselines.

Avoiding Reproducing Questions

A concern in our setting when working with a synthetic dataset like CLEVR is that by automatically generating questions we may end up replicating the questions in the dataset. Since CLEVR questions have also been produced using templates, we should ensure that we did not reproduce the questions of the other $(100-x)\%$ of the dataset when augmenting s-CLEVR$_x$ with automatically-generated questions. Otherwise comparing the outcomes when using augmented data to that of the original dataset is not fair.

To address this concern, we first randomly select a subset of $x\%$ of the CLEVR images, I_x. then collect all the questions referring to those images, and use them as the $x\%$ subset of the dataset, s-CLEVR$_x$. In CLEVR the rate of the question referring to a specific image is almost fixed to 10. So the set of questions about I_x roughly equals $x\%$ of the full dataset. We only make use of the scene information of I_x for data augmentation.

Baselines

We use the *execution engine*, $\mathcal{E}$, to assess the proposed data augmentation impact on the VQA performance in comparison to two baselines. One baseline is when the model is trained on s-CLEVR$_x$ which includes only the subset of CLEVR complex questions about the images in I_x while the other baseline trains the model on only generated questions denoted as Aug^{simple} which includes simple and short questions. We do not compare with the state of the art, because the goal of our paper is to study VQA in a low-data regime,

	# images	# unique colors	# unique shapes	# unique materials	# unique size
5%	3,480	10,198	2,604	862	835
10%	6,961	20,288	5,204	1,727	1,673
20%	13,917	40,520	10,331	3,490	3,365
30%	20,859	60,927	15,575	5,179	5,145

	Exist Questions	**Attribute Questions**	**All Questions**
Subcategories	yes, no	color, shape, size, material	-
Questions length	6	6	6
Program Length	4	4	4
Count in 5%	10,817 (16%)	57,996 (84%)	68,813
Count in 10%	21,715 (16%)	115,568 (84%)	137,283
Count in 20%	43,375 (16%)	230,824 (84%)	274,199
Count in 30%	65399 (16%)	347,304 (84%)	412,703

Table 2. The statistics of generated questions for each training subset. **Top:** the number of images are chosen to generate augmented questions and the number of unique attributes, *i.e.* uac_1, that are extracted from those images. **Bottom:** The length and number of generated questions per question type and in total.

and to the best of our knowledge, there is no other work that conducts similar research. Thus, we focus on improving the performance of our baseline models.

5.2 Results and Discussion

Data Augmentation Statistics

Table 2 shows some statistics of the data augmentation experiment. As seen in the upper part of the table, from $\sim 70k$ images in the CLEVR dataset of which we randomly select $x\%$ of images for each subset. In other words, the size of Ix for each subset in our experiments is as shown in the table, *e.g.* in the case of $I20$ it is 13,917. Then, all uac_1 were extracted from the image scenes which can be further divided into colors, shapes, materials, and sizes. The statistics neatly correlate with the number of values of each attribute. For instance in CLEVR dataset, the attribute `color` includes eighth values of *blue, cyan, green, gray, yellow, brown, red,* and *purple*. Thus it is more likely for a certain object in an image to have a color different from other objects in the image. As a result, the number of unique colors are presented in the images $\in Ix$ are largely higher than other attributes. On the other hand, `size` has only two values of *large* and *small*. So an object is less probable to be the only `small` or `large` object in the image particularly in the current VQA datasets in which the images are relatively rich in terms of the number of objects.

The lower part of Table 2 reports the statistics of the generated questions from the selected subsets. For instance, from the extracted uac_1 of I_{20}, we generated 274,199 questions in total of which 43,375 are of *exist* type while the remaining are from *what-[attribute]* type. The number of *existential* questions with *yes* and *no* answers is almost equal containing 21,665 and 21,710 questions respectively. Considering that the size of training sets are $35k$ for s-CLEVR$_5$, $70k$ for s-CLEVR$_{10}$, $140k$ for s-CLEVR$_{20}$, and $210k$ for s-CLEVR$_{30}$, we can say that the numbers of generated questions are roughly twice the original sets size. The rest of 203,824 questions ask about four attributes of `color, shape, size`, and `material` each 57,706 questions. As for every uac_1, four questions corresponding to the four attributes are generated, hence the count of the questions for all attributes is the same. We use only uac_1 for *what-[attribute]* questions and uac_2 for *existential* questions, according to templates in Table 1, the question's length will be 6 and the length of their corresponding programs will be 4.

How does the augmentation change the training set distributions?

Figure 4 compares the distribution of questions length in s-CLEVR$_{20}$ and Aug^{simple} as well as the `valid` set. From the close-up shot of the curves, we can see that the distribution of s-CLEVR$_{20}$ and `valid` set is very similar. This is due to the fact that the distribution of the `train` and `valid` set are similar in the CLEVR dataset and the s-CLEVR$_{20}$ is a random subset of `train` set which means they share the same distribution. As seen, augmenting s-CLEVR$_{20}$ with Aug$_{20}^{simple}$ dramatically changes the distribution of questions in terms of length and type. One may hypothesise that such distribution dissimilarity will adversely affect the performance. However, the results demonstrate that in our setting it significantly enhances the model performance in predicting answers.

Training

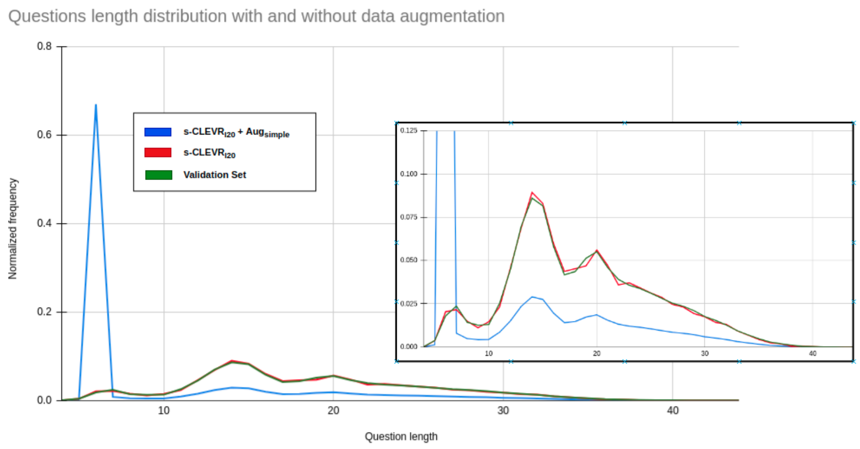

Figure 4. Comparison of the distributions of questions length with and without data augmentation. s-CLEVR$_{20}$ and `val` set are sampled from the dataset original distribution while s-CLEVR$_{20}$+Aug^{simple} shows the distribution of the augmented training set.

Training Set	5%	10%	20%	30%
Aug^{simple}	31.69	30.18	30.14	30.17
s-CLEVR	46.91	49.90	54.24	87.70
s-CLEVR$_+$Aug^{simple}	69.23	83.06	87.81	91.26

Table 3. The accuracy on CLEVR `val` set when training on different sized sets.

Table 3 depicts the results of model validation when training on (1) generated simple examples, Aug$_x^{simple}$, (2) a subset of complex questions, s-CLEVR$_x$, and (3) the augmented sets, s-CLEVR$_x$+Aug$_x^{simple}$. As seen our data augmentation method, *i.e*. training on s-CLEVR+Aug^{simple} outperforms the baselines. The best performance of our approach is achieved by training on and s-CLEVR$_{10}$+Aug$_{10}^{simple}$ and s-CLEVR$_{20}$+Aug$_{20}^{simple}$ with 34 scores accuracy increase. This large improvement confirms our initial hypothesis about lacking the basic concepts in a small training set with only complex questions. Adding simple questions to the training set helps the model to learn the basic concepts as the basis of complex ones. Then the model learns how to combine those simple concepts by training on a small set of complex questions. These two skills together make the model effectively generalize on unseen examples and it is the reason for producing 34% more accurate answers on `val` set in the cases of 10% and 20% subsets.

This analysis also explains the results of other subsets. In the case of s-CLEVR$_5$ the set of complex questions is not large enough for the model to learn all variations of combining the basic concepts. So the improvement is less than the next two larger subsets.

s-CLEVR$_{30}$+Aug$_{30}^{simple}$ increase only 5% improvement in the accuracy. It is because s-CLEVR$_{30}$ contains almost 200k of complex examples. The numbers of complex examples are quite large so that the model can implicitly infer many basic concepts by repeatedly visiting close examples. In this case, explicitly introducing basic concepts cannot largely enhance the performance.

It is noteworthy that the poor results on Aug^{simple} sets are as expected since the training sets are not challenging enough for a model to learn how to deal with complex questions. Therefore many predicted answers on `val` set are incorrect.

6 Conclusion

This paper explores VQA in low data settings motivated by the low performance of VQA models in the absence of sufficient data. To improve the performance, we propose a data augmentation method aiming to explicitly inject certain inductive biases. The inductive biases are based on the inherent compositionality of the questions which allows a complex question to be decomposed into smaller parts. We utilize this feature to augment the training set with basic questions where the learning can occur easier. The proposed data augmentation approach relies solely on the existing training set without seeking help from any other data resources. The results show that our method outperforms the baseline in all cases by a large margin.

References

[1] Vedika Agarwal, Rakshith Shetty, and Mario Fritz. Towards Causal VQA: Revealing and Reducing Spurious

Correlations by Invariant and Covariant Semantic Editing. *arXiv:1912.07538 [cs]*, May 2020. arXiv: 1912.07538.

[2] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't Just Assume; Look and Answer: Overcoming Priors for Visual Question Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4971–4980, June 2018.

[3] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48, 2016.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *IEEE international conference on computer vision*, pages 2425–2433, 2015.

[5] Yonatan Bitton, Gabriel Stanovsky, Roy Schwartz, and Michael Elhadad. Automatic Generation of Contrast Sets from Scene Graphs: Probing the Compositional Consistency of GQA. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 94–105, Online, 2021. Association for Computational Linguistics.

[6] Vinay Damodaran, Sharanya Chakravarthy, Akshay Kumar, Anjana Umapathy, Teruko Mitamura, Yuta Nakashima, Noa Garcia, and Chenhui Chu. Understanding the Role of Scene Graphs in Visual Question Answering. *arXiv:2101.05479 [cs]*, Jan. 2021. arXiv: 2101.05479.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, June 2009. ISSN: 1063-6919.

[8] Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. Evaluating Models' Local Decision Boundaries via Contrast Sets. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, volume EMNLP 2020 of *Findings of ACL*, pages 1307–1323. Association for Computational Linguistics, 2020.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. ISSN: 1063-6919.

[10] Drew A Hudson and Christopher D Manning. GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6700–6709, 2019.

[11] Drew A. Hudson and Christopher D. Manning. Learning by Abstraction: The Neural State Machine. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5901–5914, 2019.

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456, Lille, France, July 2015. JMLR.org.

[13] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910, 2017.

[14] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2989–2998, 2017.

[15] Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. Data Augmentation for Visual Question Answering. In *International Conference on Natural Language Generation*, pages 198–202, Santiago de Compostela, Spain, 2017. Association for Computational Linguistics.

[16] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, and others. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017. Publisher: Springer.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, volume 1 of *NIPS'12*, pages 1097–1105, Red Hook, NY, USA, Dec. 2012. Curran Associates Inc.

[18] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[19] Arijit Ray, Karan Sikka, Ajay Divakaran, Stefan Lee, and Giedrius Burachas. Sunny and Dark Outside?! Improving Answer Consistency in VQA through Entailed Question Generation. In *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing,* {*EMNLP-IJCNLP*}, Sept. 2019. arXiv: 1909.04696.

[20] D. Ruprecht and H. Muller. Image warping with scattered data interpolation. In *IEEE Computer Graphics and Applications*, volume 15, pages 37–43, Mar. 1995.

[21] Ramprasaath R. Selvaraju, Purva Tendulkar, Devi Parikh, Eric Horvitz, Marco Tulio Ribeiro, Besmira Nushi, and Ece Kamar. SQuINTing at VQA Models: Introspecting VQA Models With Sub-Questions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10003–10011, 2020.

[22] Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. Cycle-Consistency for Robust Visual Question Answering. In *IEEE Conference on Computer Vision and Pattern Recognition.*, 2019. arXiv: 1902.05660.

[23] Connor Shorten and Taghi M. Khoshgoftaar. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1):60, July 2019.

[24] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple

Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[25] Hao Tan and Mohit Bansal. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[26] Damien Teney, Lingqiao Liu, and Anton Van Den Hengel. Graph-Structured Representations for Visual Question Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3233–3241, Honolulu, HI, July 2017. IEEE.

[27] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4995–5004, 2016.