

Learning from Lexical Perturbations for Consistent Visual Question Answering

Spencer Whitehead¹, Hui Wu², Yi Ren Fung¹, Heng Ji¹, Rogerio Feris², Kate Saenko^{2,3}

¹UIUC ²MIT-IBM Watson AI Lab, IBM Research ³Boston University

{srw5, yifung2, hengji}@illinois.edu, {wuhu, rsferis}@us.ibm.com, saenko@bu.edu

Abstract

Existing Visual Question Answering (VQA) models are often fragile and sensitive to input variations. In this paper, we propose a novel approach to address this issue based on modular networks, which creates two questions related by linguistic perturbations and regularizes the visual reasoning process between them to be consistent during training. We show that our framework markedly improves consistency and generalization ability, demonstrating the value of controlled linguistic perturbations as a useful and currently underutilized training and regularization tool for VQA models. We also present VQA Perturbed Pairings (VQA P2), a new, low-cost benchmark and augmentation pipeline to create controllable linguistic variations of VQA questions. Our benchmark uniquely draws from large-scale linguistic resources, avoiding human annotation effort while maintaining data quality compared to generative approaches. We benchmark existing VQA models using VQA P2 and provide robustness analysis on each type of linguistic variation.¹

1. Introduction

Though great progress has been made towards Visual Question Answering (VQA), the approaches lack robustness and are sensitive to input variations [12, 1, 56, 42]. In particular, prior work [42] has shown that when presented with a rephrased version of a question, VQA models often produce inconsistent answers. We conjecture that this is likely because most VQA approaches ignore interconnections between related questions and handle each example independently during training, even though learning such relationships is paramount to generalization, robustness, and compositionality.

While a few attempts have been made to improve VQA robustness via augmentation and regularization techniques [53, 42, 40], they encourage consistency among related questions at the answer prediction level, without considering the stronger form of consistency between the in-

termediate computation steps. Furthermore, proposed augmentations are either costly (human-generated [42]) or suffer from quality control issues [49].

In this work, we propose a novel robust VQA approach that first augments the question and then enforces consistency not only of the answer, but also of the intermediate representations computed by the model. The intuition is that (like humans) VQA models should follow the same reasoning steps to solve two differently-phrased questions that have the same meaning. For example, to answer both questions “Are the buildings tall?” and “Are the buildings short?” a model should follow the same process, *i.e.* detect buildings in the image, then classify their height.

How to enforce such consistency of the intermediate reasoning steps appears to be a hard problem in general. We propose that an effective way to improve this stronger notion of *reasoning consistency* is by maintaining the associated computation steps between questions that differ by controlled variations. We leverage the family of interpretable, compositional VQA models called *Neural Module Networks (NMNs)* [4, 17, 16, 43] which explicitly represent sub-tasks like object detection and spatial reasoning as *modules* within the network and predict sequences of weights over these modules (akin to *programs*) to solve each question. However, unlike existing NMN work, we regularize the model to learn not only how to compose sub-tasks, but also follow the same sequences of sub-tasks to answer two variations of the same question, illustrated in Fig. 3.

In addition, we also propose a novel data augmentation algorithm based on auxiliary linguistic knowledge freely available in text-only corpora. Examples like in Fig. 1, seem to suggest that existing VQA models ignore the relatedness between questions with mild linguistic modifications. Questions like the antonymous pair in Fig. 1 are related in that they query the same property of the same object, but differ in that they ask the opposite of each other. Existing work ignores such modifications and focuses either on human-written paraphrased VQA questions [42] or auto-generated paraphrases using back-translation [49]. Human paraphrases tend to add filler phrases or even change the question meaning (see Sec. 3) and are very costly to an-

¹Data and resources are publicly available:
<https://github.com/SpencerWhitehead/vqap2>



Figure 1: We show that existing VQA models can be surprisingly sensitive to simple, low-level linguistic changes. Examples show three types of lexical changes, along with answers predicted by a recent model [22]. The goal of this work is twofold: how to benchmark such phenomenon, and how to improve a model’s robustness against linguistic perturbations.

note, while back-translations are also hard to control and have quality problems. Motivated by this, we propose to augment data with changes at a low level, such as simple lexical substitutions. We create variations by substituting parts of the questions using rules extracted from large-scale language resources [37, 38] which keeps the meaning and the realistic distribution of the original questions while avoiding the cost and/or semantic incoherence commonly found with prior work.

Finally, we contribute VQA Perturbed Pairings (VQA P2), a dataset of perturbed questions derived from VQA v2 that can be used to measure robustness to the specific linguistic phenomena not previously evaluated in VQA literature, covering the usage of different synonyms (*Synonymous*), different phrasings (*Paraphrastic*), and opposite attributes (*Antonymous*). VQA P2 is comprised of 36.2k VQA questions, where each question has a corresponding question in VQA v2 [12] that it differs from by controlled perturbations. VQA P2 can be easily expanded without the need for expensive human annotations [20, 40, 42], while maintaining control over the perturbations used.

To summarize, our main contributions are:

- A novel VQA consistency regularization method that augments questions and enforces similar answers and reasoning steps for the original and augmented questions.
- A new data augmentation method and VQA robustness benchmark, VQA P2, to diagnose the robustness of VQA models to controlled linguistic perturbations. To the best of our knowledge, this is the first approach to use large-scale linguistic resources automatically mined from human-written text to enhance VQA.

- Experiments applying our approach to regularize the training of modular, compositional, and interpretable VQA models, demonstrating that it leads to significant improvement in model consistency.

2. Related Work

VQA. Prodigious progress has been made towards VQA. Models that learn strong feature backbones with co-attention/bilinear attention have achieved top scores on the widely adopted VQA v2 benchmark [3, 25]. Recently, vision and language transformer models [47, 32, 45, 29], especially when pre-trained on large-scale vision and language datasets, have led to state-of-the-art performance on VQA accuracy. Despite this success, these models are generally not transparent and hard to diagnose. In contrast, compositional and modular approaches, like NMNs, offer a number of distinct advantages, such as being more interpretable [16, 35], requiring less training data [26, 34], and being able to handle complex, compositional questions [4, 17, 16, 43]. Although, there is often a trade-off between accuracy and interpretability.

VQA Benchmarks. VQA v2 [12] has become the de facto training and evaluation data resource. Although improvements on VQA v2 test accuracy has marked encouraging progress on VQA, previous work has found that models are prone to learning superficial correlations by taking advantage of dataset bias [12, 1], and the standard VQA accuracy metric does not help to quantify such bias. A number of efforts have since been proposed to serve as additional diagnostic and evaluation tool for VQA. CLEVR [23] and GQA [20] create compositional questions which are hard to answer if the model relies on language bias. However, the questions in these datasets are synthetic and do not match

the characteristics of natural questions about real images. VQA-CP [1] creates training and testing data splits which have different answer distributions. TDIUC [24] categorizes and measures performance by 12 different question types. While these benchmarks can measure a model’s generalization ability to new question types or new answer distributions, they do not serve to examine how much an answer might change if we change a specific input question. More related to our work is VQA-Rephrasings dataset [42], which has rephrased questions associated with a subset of original VQA questions. However, as we have discussed, the open-endedness nature of human-written paraphrases makes it hard to identify the source of model inconsistencies. Unlike our dataset (VQA P2), it does not control for the specific type and degrees of question variations, making it hard to codify how questions are related to one another and learn from such relationships.

Improving VQA Robustness. Robustness of deep learning models has been intensely studied in recent years, both in the context of image classification [55, 14] and in a number of natural language understanding tasks [11, 7]. Within VQA, robustness has been studied under the perspective of grounding model predictions to visual regions that are interpretable and consistent with human annotated attention maps [30, 53]. While visual grounding provides a better basis for learning representations of concept words in questions, many words and expressions in VQA questions do not have a direct correspondence with visual regions (i.e., not “groundable”). More related to our work, [53, 42, 40] examines model consistency under input question variations. However, these models that consider consistency among related questions only do so at the answer prediction level [40, 42, 53]. Additionally, another approach applies embedding constraints between questions to improve performance [50]. Our framework not only employs answer prediction loss, but also dissects the model and enforces similarities on the sub-tasks involved, which aligns with compositional and transferable modeling of VQA.

3. VQA P2 Benchmark

Our goal is to create an objective benchmark to measure the progress of robustness in VQA models, specifically the consistency of VQA predictions under different linguistic perturbations of the input questions. So the question is: what is the best method to create realistic, expressive, and semantically coherent, in an efficient fashion?

Visual question generation methods [30, 31] can generate a variety of questions, but the output can be incoherent for a specific image or may be limited to certain types of questions [31]. More recently, back-translation has emerged as a method of creating paraphrased questions for VQA [49]. However, in general, controlled generation of text is a difficult open challenge [18]. Alternatively,

template-based methods [20, 40] can offer more control and semantic coherence, but require extensive annotations (e.g., scene graphs [28]) and are limited in expressiveness by the templates used. Human-written paraphrases is the other option to generate robustness benchmark [42]. Other than the obvious drawback of being more costly, human-written question tend to be generic, not good for diagnosing the particular source that causes answer prediction inconsistencies, as well as being often overly verbose (as shown in Sec. 3.2).

Different from existing approaches, we build our dataset from the original VQA v2 data, and create variations by substituting parts of the questions using rules extracted from large-scale language resources. Doing so keeps the expressiveness and realistic distribution of the original questions and avoids the semantic incoherence commonly found with trained question generators. Utilizing large-scale linguistic resources offers an efficient means for creating questions that contain linguistic variations exhibited by humans [38].

3.1. Paraphrasing Creation Pipeline

Perturbation	Example rules (source → target)
Synonymous	$car \rightarrow automobile$ $refrigerator \rightarrow fridge$ $phone \rightarrow telephone$
Paraphrastic	$performed\ in\ [NP] \rightarrow carried\ out\ within\ [NP]$ $be\ considered\ [NP] \rightarrow be\ viewed\ as\ [NP]$ $participating\ in\ [VP] \rightarrow taking\ part\ in\ [VP]$
Antonymous	$open \rightarrow closed$ $wet \rightarrow dry$ $full \rightarrow empty$

Table 1: Example rules used to generate the data. Synonymous and antonymous rules are single word, while paraphrastic rules can contain multi-word expressions and grammatical constraints.

Substitution Extraction. We extract lexical substitution rules from the Paraphrase Database 2.0 (PPDB) [38], a lexical database containing over 100 million paraphrases automatically mined from human-written text, as well as WordNet [37], two large-scale linguistic resources, and apply these rules to existing VQA v2 questions. We create three types of perturbations: 1) *synonymous* perturbations that substitute a single word with its synonym; 2) *paraphrastic* perturbations that substitute multi-word phrases for single/multi-word phrases with the same meaning; and 3) *antonymous* perturbations where adjectives or verbs are substituted with their antonyms, which explores the ability to understand opposite states of an attribute or action. We extract synonymous and paraphrastic rules from the lexical and syntactic subsets of PPDB respectively, and accept the rule if the target is “equivalent” or “entailed” by the source,

according to PPDB’s lexical constraints. We gather antonymous rules from WordNet.

Rule Refinement. We automatically determine which rules can be applied by matching the source phrases and grammar requirements to the questions, discarding rules that are not applicable to any questions. We ensure that our rules are not simply adding unknown words by removing rules whose source or target contain words that don’t appear in the VQA vocabulary. We filter the synonymous and paraphrastic rules with a minimum confidence threshold to prevent low quality/frequency substitutions. Examples of these rules are shown in Tab. 1, where each rule is a mapping from a source to target phrase.

Applying Substitution Rules. We apply the filtered rules to VQA v2 questions and obtain our final benchmark. Since consistency metric requires the groundtruth annotation for the each question, and VQA v2 testing sets do not offer publicly available answer annotations, we use VQA v2 validation set as the basis for our benchmark. This practice is common among methods that require per-question answer annotations [42, 2].

During the rule application process, for each question, multiple substitutions for a specific type of perturbation can be made, which increases variations. For antonymous rules, we perform word sense disambiguation [48] and limit their application to yes/no questions of the form “*is/are the*” and “*is/are this/these*” where the WordNet synsets of the source word in both the question and rule match. Antonymous rules are limited to these kinds of questions to help ensure that we only apply the antonym substitutions to attributes/states that are directly queried (e.g., “*Is the window open?*”), and not simply mentioned in the question (e.g., “*What’s near the open window?*”). For answers, synonymous and paraphrastic questions use the same answers as their VQA v2 counterparts as they share the same meaning and antonymous questions take the opposite answers. This results in a set of 36,175 question-answer pairs that are perturbed counterparts of VQA v2 validation questions.

3.2. Comparison to Other Paraphrasing Methods

Two main alternative methods for generating paraphrases of questions are human annotations and generative approaches. Fig. 2 shows a few examples from human-written paraphrased VQA questions [42] and auto-generated paraphrases using back-translation [49]. When primed with the original question, human annotators tend to give paraphrases that are longer than the original question, with a large percentage of these simply adding filler phrases that do not change the original sentence by much. For example, in the first human-written example in Fig. 2, the original question is essentially still preserved within the paraphrased version. Additionally, human paraphrases can introduce multiple sources of variations, such as introduc-

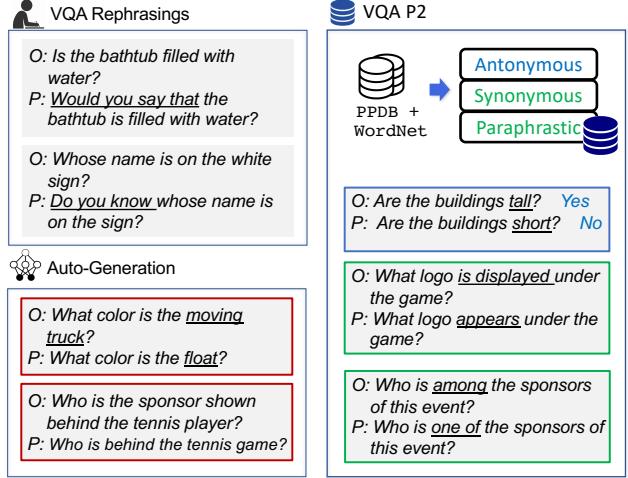


Figure 2: Examples of VQA P2, human generated [42], and back-translated (Auto-Generation) variants (P) of input questions (O). VQA P2 uniquely offers three types of controlled linguistic variations to benchmark robustness.

ing commonsense related items, sentence structural changes as well as lexical alterations. Consequently, using human-written paraphrases as a diagnostic benchmark lacks a level of precision needed to diagnose and understand model performance. On the other hand, automated methods like back-translation suffer from many quality control issues. Fig. 2 shows how this method frequently generates mismatched phrasal replacement or semantic drift, where the generated questions no longer hold the same meaning.

In contrast, our benchmark has built-in quality control as the replacement rules extracted from linguistic resources correlate well with human judgements [38] and are generally reliable. We target at linguistic perturbations to factor out other sources of variations when diagnosing models. Our dataset offers control over the kinds of perturbations used and allows for more precise evaluation of model consistency, meaning we can evaluate a model’s capacity for addressing specific types of perturbations offering more diagnostic insight. Additionally, due to the fast, automated nature of this process, our dataset is easily extensible to more linguistic variations and larger sets of perturbations.

4. Approach

Given an image-question pair, (I, Q) , a VQA model maps the pair to a distribution over an answer set, $f(I, Q) \rightarrow \mathbf{a}$. Existing approaches most often treat this as a classification task, minimizing the prediction loss between the predicted answer and the ground truth answer. This standard approach, however, does not take into account possible relations among questions. The goal of our approach is to train a model to be aware of question relationships, and

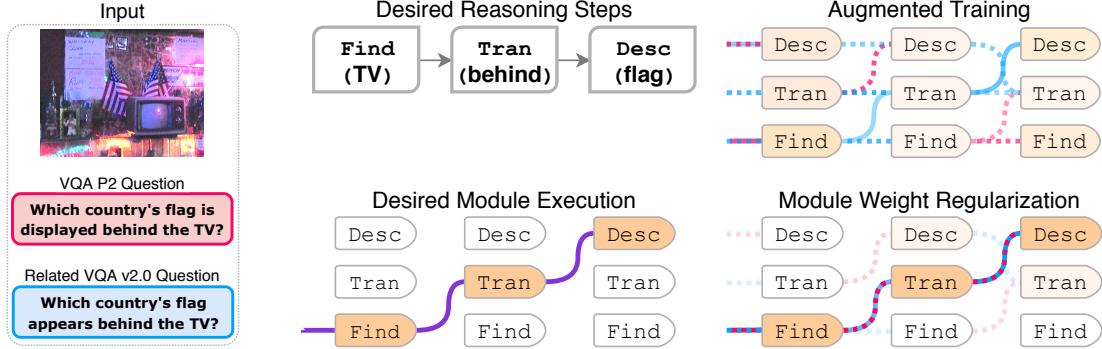


Figure 3: We propose a novel Q3R framework that improves the robustness of VQA models against linguistic variations by augmenting questions and encouraging similar module weights between related questions.

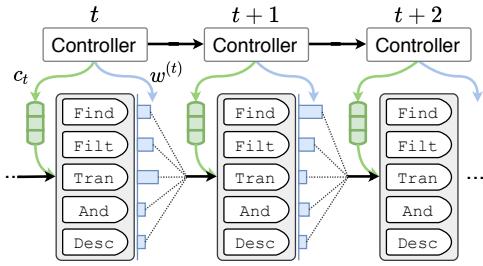


Figure 4: Illustration of the backbone model.

thereby learn to be more consistent when answering.

Modeling Question Relatedness. Typically, given an input question, a predictable set of reasoning steps are expected to answer the question. For example, in Fig. 3, the sub-networks which can answer “*Which country’s flag is displayed behind the TV?*” should be able to decompose this task into components such as “*Find(TV)*”, “*Transform(Behind)*”, and “*Describe(which country’s flag)*” and learn to transfer all sub-networks to the question: “*Which country’s flag appears behind the TV?*” Essentially, there should be a set of elementary operations where each one solves a less complex task than the original question and related questions can share these operations, following a similar order of execution. Based on this intuition, we propose our **Question-Relatedness Regularized Reasoning (Q3R)** framework.

4.1. Q3R Framework

Our framework is comprised of three components: 1) a method to create linguistic variations of input questions; 2) a compositional backbone model, guided by question-based module selection; and 3) a mechanism to enforce similarities of related questions at the module level.

Creating Related Questions. The input to our pipeline is an image-question pair, (I, Q) , and, during training, a cor-

responding answer A . We create a module, $g(Q, A) \rightarrow (\tilde{Q}, \tilde{A})$, that takes Q and A as input and outputs a related question-answer pair (\tilde{Q}, \tilde{A}) . In practice, any approach that can create linguistic variants of a given question-answer pair can be applied here. For example, one can use generative models [49] to augment questions. Although this data is typically too noisy for benchmarking purpose, it can provide additional data for training. In this work, we simply use the rules provided by VQA P2 to create linguistic substitutions, making g a function of the perturbation type, t , as well. In future work, we will explore combinations of rule-based paraphrases and generative methods.

Backbone Model. Our backbone model is comprised of input encoders, a controller network, and a set of re-usable modules, \mathcal{M} . The input encoders compute a set of visual and textual features for I and Q , respectively. As shown in Fig. 4, the controller is responsible for decomposing the reasoning process into a sequence of steps that are executed by the network. At each step, t , the controller reads the question and, based on this question, produces module weights, $w^{(t)} \in \mathbb{R}^{|\mathcal{M}|}$, which are used for module selection, as well as a textual parameter, c_t , which is an input to the modules. In this formulation, module selection is essentially an attention over the module outputs, which allows for end-to-end training. The sequence of weights over all reasoning steps represents a soft layout that specifies what modules are utilized at each step. The modules implement different sub-tasks that the model has at its disposal during reasoning.

Our framework is agnostic to the specific designs of the components of the backbone network and can work within the general controller-module framework. We therefore select two state-of-the-art compositional models to instantiate our backbone, specifically, **StackMN** [16], and **XNM** [43]. To verify that our training scheme is not tied to either one of the network, we also present a new model by adopting modules from both networks, called **HybridNet**. More architectural details are in Appendix D.

Regularization Method. We propose to regularize the training of the backbone compositional model to improve its consistency and robustness against linguistic variations at the module level. Controlling and regularizing question relatedness at the module level offers several benefits. First, it provides finer control of the model’s active sub-networks than only using supervision at the output layer. Second, the related question pairs share intermediate activation similarity but do not necessarily need to match one another at a lower level (e.g., attention maps within modules), making the model less sensitive to surface-level sentence variation.

Given an image, I_j , as well as a pairing of an original question, Q_j , and its perturbed version, \tilde{Q}_j , the controller maps each question to a set of module weights at each reasoning step, $w_j^{(t)}$ and $\tilde{w}_j^{(t)}$, respectively. Across all reasoning steps, these module weights can be interpreted as selecting “paths” over the grid of all modules across all time steps, especially when the weights are computed with Gumbel softmax [21] as done with XNM [43]. Ideally, if two questions agree on the basic sub-tasks, they should also agree on the activated module paths (as in Fig. 3). Thus, we define the regularization loss term as:

$$\mathcal{L}_m(Q_j, \tilde{Q}_j, I_j) = \lambda \sum_{t=1}^T d(w_j^{(t)}, \tilde{w}_j^{(t)}), \quad (1)$$

where λ is a hyperparameter to scale the loss term and d is a distance metric between two distributions. We find that KL divergence or common vector norm losses, such as L1-norm, work well as d in the proposed loss term.

Algorithm 1: Q3R Training Procedure

```

input: steps  $N$ ; input data  $\mathcal{D}$ ; module  $g$ ; network  $f$ 
for  $i \in \{1, \dots, N\}$  do
    sample  $(I_i, Q_i, A_i)$  from  $\mathcal{D}$ 
    compute  $\mathcal{L}_{CE}$  w.r.t.  $(I_i, Q_i, A_i)$ 
    if  $Bernoulli(r) = 1$  then
        sample  $(I_j, Q_j, A_j)$  from  $\mathcal{D}$ 
         $t \sim \text{cat}(\mathcal{T}|\mathbf{p})$ 
         $(\tilde{Q}_j, \tilde{A}_j) \leftarrow g(Q_j, A_j, t)$ 
        compute  $\mathcal{L}_m$  w.r.t.  $(I_j, Q_j, \tilde{Q}_j)$ 
         $\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_m$ 
    else
         $\mathcal{L} = \mathcal{L}_{CE}$ 
    end
    update  $f$  to minimize  $\mathcal{L}$ 
end

```

Training Procedure. Given the regularization term, \mathcal{L}_m , and the cross-entropy loss between the predicted answer distribution and the ground truth answer prediction, $\mathcal{L}_{CE}(\mathbf{a}_i, \mathbf{a}_i^* | I_i, Q_i)$, we employ a multi-task training procedure [10, 33], where we treat each perturbation type as a different task, optimizing for each perturbation type individually. As in Algorithm 1, for each input image-question

pair sampled from VQA v2, we compute \mathcal{L}_{CE} . Then, with probability r , we sample a perturbed pairing to compute our module weight loss. Otherwise, we simply update the model using \mathcal{L}_{CE} . When utilizing our loss, we sample a perturbation type, t , from a categorical distribution, denoted $\text{cat}(\mathcal{T}|\mathbf{p})$, over the set of perturbation types, \mathcal{T} , with probabilities, \mathbf{p} . We input the question Q_j and the perturbation type into g to obtain \tilde{Q}_j that differs from the input question according to the sampled type. We use the perturbed question and original image-question pair, (I_j, Q_j, \tilde{Q}_j) , to compute \mathcal{L}_m . The network is then updated to minimize the sum of these loss terms. In practice, when operating on batches, we sample a batch of a particular perturbation type. We noticed that this procedure offers a more stable learning process and better performance than mixing different perturbation types in a single batch.

5. Experiments

Data and Metrics. We use VQA v2 (Appendix C) for training and VQA P2 for our main evaluations. Since our goal is to benchmark and advance existing models’ consistency, it requires groundtruth answer labels for each test question. Since answers for the VQA v2 test sets are not publicly available, we use questions from the validation set for testing and never use it during training, as is common for evaluating on annotated subsets of VQA v2 [2, 52]. Our metrics are standard VQA accuracy as well as the consensus score (CS) [42] between pairs of related questions.²

Models. Existing VQA literature is vast [54, 51, 6] and the goal of our work is not to exhaustively study VQA models. We focus on the consistency and robustness of models in response to linguistically perturbed data. Therefore, we pick representative models for our evaluations. **BAN** [25] is a bilinear model that achieves the single-model top performance on VQA v2 without external data. BAN uses bilinear co-attention with residual connections to model the interactions between image regions and words. **Transformer** [47] is a top performing architecture that acts as a multi-modal encoder. For fair comparison, we do not use large-scale pre-training. As described in Sec. 4.1, **StackNMN** [16] and **XNM** [43] are examples of expert-free, end-to-end trainable NMNs. We experiment with both models as well as a hybrid, called **HybridNet** (see Sec. 4.1). Experiments with BAN and Transformer examine the consistency of non-NMN state-of-the-art architectures. The NMN experiments explore the performance of different module implementations when employing our framework.

Implementation. For fair comparison, all models use the same visual features from [3], input textual features from [39], and random seed of 0. Whenever possible, we

²A non-zero CS for a pair of questions requires a model to answer both questions correctly [42].

Model	Training Data	VQA v2		Synonymous			Paraphrastic			Antonymous			
		Pert	Ori	Δ	Pert	Ori	Δ	Pert	Ori	Δ	Pert	Ori	
BAN [25]	VQA v2	66.1	64.5	66.3	-1.8	56.3	56.7	-0.4	73.9	86.0	-12.1		
Transformer [47]	VQA v2	63.5	61.0	64.2	-3.2	53.0	54.7	-1.7	73.0	84.2	-11.2		
StackNMN [16]	VQA v2	62.7	61.2	63.5	-2.3	53.2	53.6	-0.4	74.8	84.9	-10.1		
XNM [43]	VQA v2	64.5	62.8	65.2	-2.4	55.6	56.8	-1.2	74.3	85.1	-10.8		

Table 2: Accuracy on the VQA v2 validation set as well as VQA P2 subsets. *Ori* and *Pert* refer to the set of questions before and after applying a particular perturbation, respectively, and Δ is the difference in performance between these sets.

use publicly available implementations. For Q3R, $\lambda = 1.0$ and $r = 0.2$ for all experiments and we use a L1 loss for our distance metric. Please see Appendix A for details.

5.1. Benchmarking Robustness with VQA-P2

We benchmark existing models on their robustness against controlled lexical perturbations, as shown in Tab. 2. Interestingly, we see that the different classes of models have less trouble with paraphrastic changes than they do with synonymous, with the average drop in performance being -0.9 compared to -2.4 . This is likely due to the fact that paraphrastic changes tend to effect transitional phrases (e.g., “*be considered*”), which models may ignore [46], whereas synonymous changes effect these as well as concept mentions (e.g., “*car*”) that are needed to answer the question. We see that models struggle the most with antonymous changes, dropping at least -10.1 . Despite having lower VQA v2 accuracy, the NMN architectures perform better on the perturbed antonymous questions compared to BAN and Transformer. The results suggest that the Transformer architecture [47] trained from scratch is one of the less robust architectures across the different types of perturbations. Overall, we see that existing models struggle with these controlled variations across the board, where the largest difficulties appear on the logical consistency measured with antonymous perturbations and concept mention consistency measured with synonymous perturbations. To our knowledge, this is the first study of VQA robustness analysis on different types of linguistic variations,

5.2. Improving Model Robustness using Q3R

We evaluate the effect of adding Q3R to different NMN architectures. Tab. 3 shows that, on VQA P2, adding our framework (+Q3R) results in significant improvements over the base models for both accuracy and CS. We also measure validation accuracy on VQA v2 in Tab. 3, which shows modest yet consistent gains, although our focus is on consistency, not overall VQA v2 accuracy. This shows that models regularized by Q3R generally see improvements in consistency and overall performance, including on VQA v2.

Model	VQA P2			VQA v2
	Pert	Ori	CS	
StackNMN	63.3	66.9	66.2	62.6
	66.9	67.4	72.2	62.7
HybridNet	63.3	67.0	66.6	63.0
	67.0	67.4	72.5	63.1
XNM	64.7	68.3	68.8	64.5
	68.1	68.9	74.4	64.7

Table 3: Effect of adding Q3R to different NMNs, measured by accuracy and CS. *Pert* denotes perturbed questions; *Ori* are the corresponding questions in VQA v2.

Analysis by Perturbation Types. A benefit of having information on the specific types of linguistic variations is that we can profile a model’s performance by each type to assist understanding and diagnosis. Tab. 4 shows that models are generally more confident with the antonymous perturbations, likely because “yes/no” questions have higher answer prediction scores in general. We note that performance gains are significant on single-word changes and less obvious on multi-word changes.

Model	Syn.		Par.		Ant.	
	μ	CS	μ	CS	μ	CS
StackNMN	62.6	64.7	51.6	53.8	79.8	76.1
	64.0	70.3	52.9	56.9	84.9	84.7
HybridNet	62.5	65.0	53.3	55.7	79.9	76.4
	64.1	70.8	55.0	59.4	84.0	83.7
XNM	64.1	67.6	56.7	60.7	79.7	76.0
	65.7	72.9	56.4	61.8	84.4	84.7

Table 4: Average accuracy (μ) between *Pert+Ori* and CS on each type of perturbation: Synonymous (Syn.), Paraphrastic (Par.) and Antonymous (Ant.).

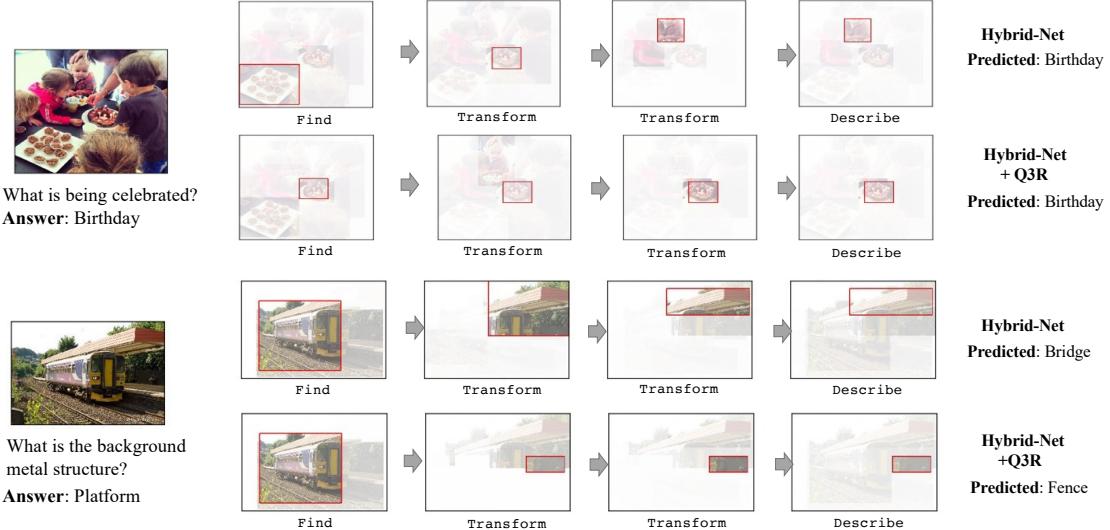


Figure 5: Visual attentions and selected modules across reasoning steps. Attention weights are indicated by the transparency of the bounding box; red boxes are the highest weighted regions. Please see Appendix E for more examples.

5.3. Further Analysis

Effect of Regularization. To further understand the effect of applying our loss on the reasoning steps involved in executing NMNs, we compute different statistics using HybridNet. For each question in VQA P2, we compute the average normed difference in output module weights between the perturbed question, \tilde{Q}_j , and the original question, Q_j for all steps. Applying our loss leads to a 93% reduction (7.4×10^{-3} to 0.5×10^{-3}) on the average difference of the module weight distribution at each step. In other words, our framework encourages more consistent reasoning steps between related questions. Also, it appears that our framework can lead to more intuitive visual attentions, as shown in Fig. 5. These results suggest that our framework may give rise to more interpretable and consistent visual reasoning.

Additional Evaluation. As we discussed in Sec. 3.2, human-written rephrasings typically contain various sources of change: such as those that involve common sense knowledge, or structural level change of sentences. Nonetheless, we are interested in observing the outcome of the additional study on VQA-Rephrasings. When trained with our Q3R framework, XNM receives a +0.7 accuracy improvement on paraphrased questions as well as a +0.4 improvement on CS score. This moderate gain may be explained by the fact that linguistic perturbations are present in human-written questions, so the model’s robustness on this dataset benefits from our training framework.

5.4. Discussion

Comparison to Expert Layout Supervision. Many NMNs adopt expert layouts to guide the search for optimal module

paths [4, 17]. While this approach is suitable on synthetic datasets with simple scenes and spatial reasoning [23], it has limited success on realistic images [16]. Our loss can be viewed as providing weak supervision to module layouts, avoiding the need for ground-truth module layout annotation, which is costly and not clearly defined for natural questions about real images.

Beyond NMNs. Our results seem to suggest the advantage of representing and incorporating inductive bias at the modular level, rather than just using answer-level supervision. While we have demonstrated our method using Neural Module Networks, we note that it could be generalized to improve the robustness of any other interpretable VQA architecture that involves the computation of sub-tasks, such as those based on executable symbolic programs [57, 34].

6. Conclusion

We show that a promising direction to improve the robustness and consistency of VQA models is by modeling and learning from lexical perturbations. We propose a novel approach based on modular networks, which creates two questions related by linguistic perturbation and regularizes the visual reasoning process between them to be consistent during training. We introduce a new benchmark, VQA P2, that features categorized, controllable linguistic variations that allows us to investigate and diagnose sources of inconsistencies in model predictions, will be made publicly available. Empirical results show that existing models have difficulties with different types of linguistic variations and that our approach is effective towards improving robustness and generalization ability.

References

- [1] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Anirudha Kembhavi. Don’t just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980, 2018. 1, 2, 3
- [2] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Anirudha Kembhavi. Don’t just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980, 2018. 4, 6
- [3] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018. 2, 6, 1
- [4] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48, 2016. 1, 2, 8
- [5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015. 1
- [6] Andrea Burns, Reuben Tan, Kate Saenko, Stan Sclaroff, and Bryan A Plummer. Language features matter: Effective language representations for vision-language tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7474–7483, 2019. 6
- [7] Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. Towards robust neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018. 3
- [8] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734. Association for Computational Linguistics, Oct. 2014. 1
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 1
- [10] Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, 2015. 6
- [11] Wee Chung Gan and Hwee Tou Ng. Improving the robustness of question answering systems to question paraphras- ing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. 3
- [12] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 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*, pages 6904–6913, 2017. 1, 2
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2
- [14] Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *International Conference on Learning Representations*, 2019. 3
- [15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1
- [16] Ronghang Hu, Jacob Andreas, Trevor Darrell, and Kate Saenko. Explainable neural computation via stack neural module networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Proceedings of the European Conference on Computer Vision*, pages 55–71, 2018. 1, 2, 5, 6, 7, 8
- [17] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813, 2017. 1, 2, 8
- [18] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596, 2017. 3
- [19] Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations*, 2018. 1
- [20] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6700–6709, 2019. 2, 3
- [21] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017. 6
- [22] Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0.1: the winning entry to the vqa challenge 2018. *arXiv preprint arXiv:1807.09956*, 2018. 2
- [23] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017. 2, 8, 1
- [24] Kushal Kafle and Christopher Kanan. An analysis of visual question answering algorithms. In *ICCV*, 2017. 3

- [25] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems*, pages 1564–1574, 2018. [2](#), [6](#), [7](#)
- [26] Seung Wook Kim, Makarand Tapaswi, and Sanja Fidler. Visual reasoning by progressive module networks. In *International Conference on Learning Representations*, 2019. [2](#)
- [27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. [1](#)
- [28] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017. [3](#), [1](#)
- [29] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. In *Arxiv*, 2019. [2](#)
- [30] Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6116–6124, 2018. [3](#)
- [31] Feng Liu, Tao Xiang, Timothy M Hospedales, Wankou Yang, and Changyin Sun. ivqa: Inverse visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8611–8619, 2018. [3](#)
- [32] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*, 2019. [2](#)
- [33] Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. In *International Conference on Learning Representations*, 2016. [6](#)
- [34] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, 2019. [2](#), [8](#)
- [35] David Mascharka, Philip Tran, Ryan Soklaski, and Arjun Majumdar. Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4942–4950, 2018. [2](#)
- [36] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Workshops at International Conference on Learning Representations*, 2013. [1](#)
- [37] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. [2](#), [3](#)
- [38] Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, 2015. [2](#), [3](#), [4](#)
- [39] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014. [6](#), [1](#)
- [40] 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 *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5860–5865, 2019. [1](#), [2](#), [3](#)
- [41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. [1](#)
- [42] Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. Cycle-consistency for robust visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6649–6658, 2019. [1](#), [2](#), [3](#), [4](#), [6](#)
- [43] Jiaxin Shi, Hanwang Zhang, and Juanzi Li. Explainable and explicit visual reasoning over scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8376–8384, 2019. [1](#), [2](#), [5](#), [6](#), [7](#)
- [44] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2014. [1](#)
- [45] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. ViL-bert: Pre-training of generic visual-linguistic representations. In *ICLR*, 2019. [2](#)
- [46] Sanjay Subramanian, Sameer Singh, and Matt Gardner. Analyzing compositionality in visual question answering. In *VIGIL@ NeurIPS*, 2019. [7](#)
- [47] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5100–5111, 2019. [2](#), [6](#), [7](#), [1](#)
- [48] Liling Tan. Pywsd: Python implementations of word sense disambiguation (wsd) technologies [software]. <https://github.com/alvations/pywsd>, 2014. [4](#)
- [49] Ruixue Tang, Chao Ma, Wei Emma Zhang, Qi Wu, and Xiaokang Yang. Semantic equivalent adversarial data augmentation for visual question answering. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 437–453, Cham, 2020. Springer International Publishing. [1](#), [3](#), [4](#), [5](#)
- [50] Damien Teney, Ehsan Abbasnejad, and Anton van den Hengel. On incorporating semantic prior knowledge in deep learning through embedding-space constraints. *arXiv preprint arXiv:1909.13471*, 2019. [3](#)

- [51] Damien Teney, Peter Anderson, Xiaodong He, and Anton Van Den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4223–4232, 2018. 6
- [52] Kento Terao, Toru Tamaki, Bisser Raytchev, Kazufumi Kaneda, and Shun’ichi Satoh. Rephrasing visual questions by specifying the entropy of the answer distribution. *arXiv preprint arXiv:2004.04963*, 2020. 6
- [53] Jialin Wu and Raymond Mooney. Self-critical reasoning for robust visual question answering. In *Advances in Neural Information Processing Systems*, pages 8601–8611, 2019. 1, 3
- [54] Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40, 2017. 6
- [55] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020. 3
- [56] Xiaojun Xu, Xinyun Chen, Chang Liu, Anna Rohrbach, Trevor Darrell, and Dawn Song. Fooling vision and language models despite localization and attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4951–4961, 2018. 1
- [57] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *Advances in neural information processing systems*, pages 1031–1042, 2018. 8
- [58] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning to count objects in natural images for visual question answering. In *International Conference on Learning Representations*, 2018. 1, 2

Appendix

A. Implementation Details

As noted in Sec. 5, all models use the same visual features from [3]³ and same pre-trained GloVe embeddings from [39]⁴. XNM, StackNMN, and HybridNet use the implementation provided by [43] to ensure consistency amongst different NMN models.⁵ We implement the modules and controller of StackNMN to match the paper description and official implementation. All NMN models are trained with the Adam optimizer [27] and have the same learning rate of 0.0008, training initialization seed of 0, and batch size of 256. Following their implementations, we use hidden dimension sizes of 512 for StackNMN and 1024 for XNM, while we use 1024 for HybridNet to match XNM. We use the recommended number of reasoning steps, $T = 3$, for XNM and use the same for StackNMN. HybridNet uses $T = 4$ to test longer reasoning sequences as well as for visualization purposes. With our Q3R framework, $\lambda = 1.0$ and $r = 0.2$ for all experiments. For BAN, we use the original source code with the 8-glimpse model, provided by the authors and adopt their training settings.⁶ We do not use the counting module [58] nor additional training data from Visual Genome [28]. For the Transformer model, we use LXMERT [47] as the architecture and utilize the publicly available code with the recommended settings from the authors.⁷ For fair comparison, we do not use large-scale pre-training.

B. Effect of Hyper-parameters

The main hyperparameters involved with using Q3R are the weight parameter, λ , and the distance function, $D(\cdot)$. Changing λ from 1 to 0.2 results in a value change of < 0.1 on VQA v2 accuracy and < 0.1 on CS score on VQA P2, for StackNMN. Also, the difference of applying KL-divergence based loss function and L1-norm based loss function results is also < 0.1 , for both VQA v2 accuracy and CS score on VQA P2.

C. Additional Dataset Information

To measure the model’s robustness against question variations, it requires the availability of answer annotations during evaluation. Since the testing data of VQA v2 [12] does not have public groundtruth information, we use the validation split of VQA v2 as the testing set for all models.

³<https://github.com/peteanderson80/bottom-up-attention>

⁴Common Crawl 840B: <https://nlp.stanford.edu/projects/glove/>

⁵<https://github.com/shijx12/XNM-Net>

⁶<https://github.com/jnhwkim/ban-vqa>

⁷<https://github.com/airsplay/lxmert>

For training, we use the training data of VQA v2, which contains 443,757 questions for 82,783 images; for testing, we use the validation split of VQA v2 (214,354 questions for 40,504 images) and VQA P2 (36,175 question-answer pairs)

In Sec. 5.3, we evaluate our framework on VQA-Rephrasings [42]. VQA-Rephrasings is a human-written paraphrased dataset that spans the 40,504 validation images of VQA v2, where each image has a corresponding question group that contains 1 original question from the VQA v2 validation set and 3 rephrasings of that questions, for a total of 162,016 questions.

D. Model Details

D.1. Controller-Module Architecture

Our proposed training framework is applicable to the any controller-module framework [4, 17, 16, 43]. Next, we provide details of the components of our backbone architecture, which is comprised of input encoders, a controller, and a set of functional modules.

Input encoders. The visual features, $x \in \mathbb{R}^{d_v \times K}$, represent K different regions of the image, which may be grid features from a convolutional layer of a CNN [44, 13], object features from an object detector [41], or a scene graph [28, 23]. Similarly, representations of question tokens, (h_1, \dots, h_N) , or the entire question, q , may be obtained via embeddings [36, 39], an encoder [15, 8], or a pre-trained language model [9].

Controller. At each reasoning step, t , the controller reads the question and computes c_t and $w^{(t)}$. $w^{(t)} = \{w_m^{(t)}\}_{m=1}^{|\mathcal{M}|}$ is a set of module weights that are utilized by the network to perform a soft selection of modules at step t . These module weights are computed via a soft attention [5], forming a distribution over the modules, which is used to compute the output at step t by taking the weighted average over each module’s output. The vector $c_t \in \mathbb{R}^d$ is a textual parameter, typically computed using an attention over the question tokens [16], which is given as input to the modules and used, for example, as a query for visual attentions. Fig. 6 shows the architecture of the controller, which has a similar design as the controller in [16, 19]. At each step, the controller first concatenates the history vector, which is the textual parameter from the previous step, with the question representation, q , which is the final hidden state of a bi-directional GRU [8]. Next, the concatenated vector is given as input to a MLP. The output of the MLP (with an optional normalization layer) is then used to predict module weights as well as compute an attention over the question tokens, which yields a new textual parameter, c_t .

Modules. Each module is responsible for an elementary operation, with inputs that may be any combination of attention maps generated at previous steps, the textual parameter

Module	Input	Output	StackNMN [16]	XNM [43]
Find	x, c_t	\hat{a}	$\text{conv}_2(\text{conv}_1(x) \odot W_{ct})$	$\text{conv}(g(x, W_{ct}))$
Transform	x, c_t, a	\hat{a}	$\text{conv}_2(\text{conv}_1(x) \odot W_1 \sum(a \odot x) \odot W_2 c_t)$	$\text{norm}((\sigma(RW_{ct}) \odot M)a)$
Filter	x, c_t, a	\hat{a}	$\text{And}(\text{Find}(x, c_t), a)$	(same as StackNMN)
And	a_1, a_2	\hat{a}	$\min(a_1, a_2)$	(same as StackNMN)
Describe	x, c_t, a	$z^{(t)}$	$W_1^\top (W_2 \sum(a \odot x) \odot W_3 c_t)$	$\sum a \odot x$
NoOp	none	-	-	-

Table 5: Neural modules of StackNMN and XNM. Here x are the visual features, c_t is the textual parameter, each a (and \hat{a}) is an attention map over the image regions, $z^{(t)}$ is used to compute the final answer prediction, and all W are learned parameters. For XNM, R is the set of edge features and M is the adjacency matrix of the scene graph.

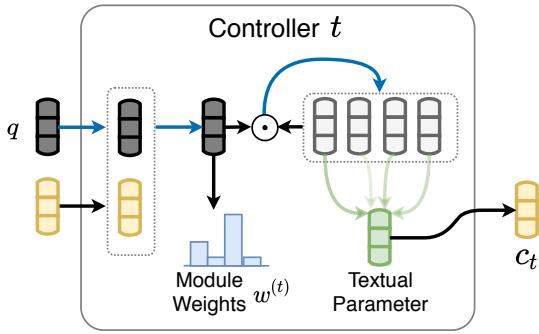


Figure 6: Illustration of the controller network.

from the controller, the visual features, or any other relevant features. A module outputs either an attention map over the image regions, $a \in \mathbb{R}^K$, or a vector, $z^{(t)} \in \mathbb{R}^d$, which may be used for answer prediction. The output of the modules at the last time step is combined with the question representation and used to predict the final answer via an output layer, such as a MLP.

D.2. Models

The backbone architectures can be instantiated differently to realize a variety of models with distinct module functionalities, reasoning steps, feature backbones, etc. In this work, we adopt three designs of the controller-module models to test the effects of our Q3R training framework.

StackNMN [16]. For StackNMN, the original method uses grid features from a CNN [13] as visual features. For fair comparison with other models, we use object features [3] for StackNMN, which is a stronger feature backbone for VQA. The modules in this architecture largely use element-wise multiplications to fuse visual and linguistic features, compute different attention maps, or obtain answer vectors. Additionally, in Find and Transform, 1D convolutions are also used to compute weighted visual and multi-modal features. The specific module designs for StackNMN are shown in Tab. 5.

XNM [43]. For XNM, the visual features, $x \in \mathbb{R}^{d_v \times K}$, are object features that represent nodes in the input scene graph.

The edge features, $R \in \mathbb{R}^{K \times K \times 2d_v}$, for the scene graph are the concatenations of neighboring edges and $M \in \mathbb{R}^{K \times K}$ is the adjacency matrix of the scene graph. The Transform module is the only module that considers the scene graph and uses the graph information to shift the visual attention according to the graph connectivity. These modules adopt the naming conventions of StackNMN, but XNM, in particular, has a different Transform implementation, which learns attention transforms on image scene graph representations, and an alternative multi-modal fusion method [58]. The detailed module information is shown in Tab. 5.

HybridNet. To further investigate whether our framework can be effective regardless of the module implementations, we present a hybrid of StackNMN and XNM. Specifically, HybridNet utilizes the Transform module of StackNMN, while maintaining the rest of the design from XNM.

E. Output Examples

Here are some example outputs from HybridNet with and without Q3R. In Fig. 7, the model trained without Q3R maintains its answers despite the perturbation, whereas the model trained with our framework predicts the appropriate answer and also matches visual attentions between them. Then, in Fig. 8, we see an example where the questions share the same meaning and the same modules are selected for both questions and both models, but the model without our framework yields inconsistent answers.

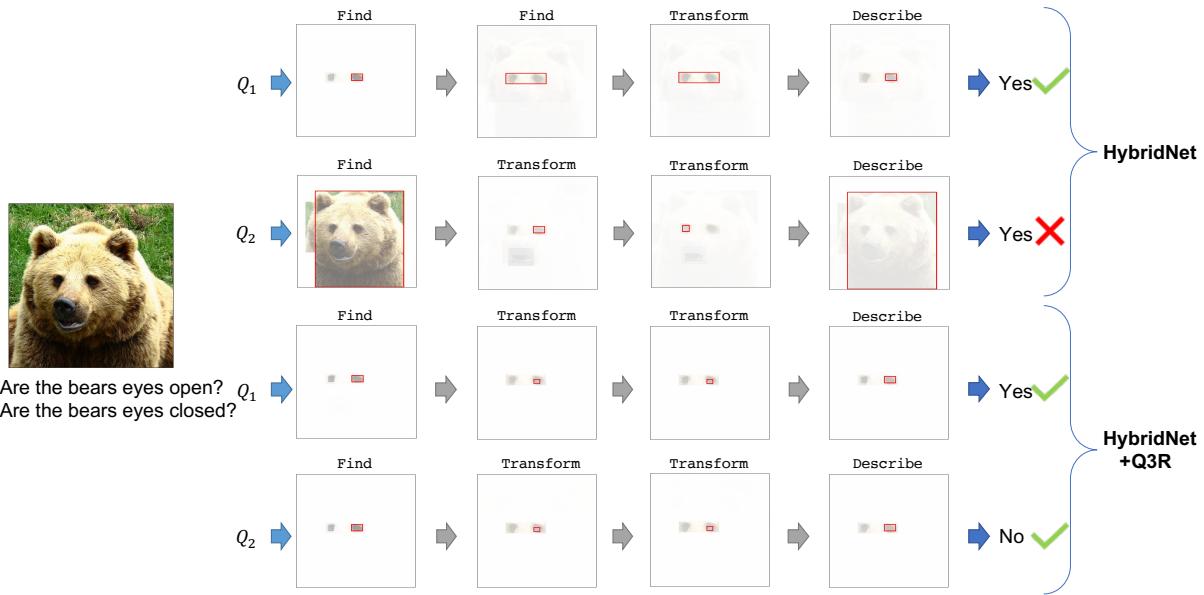


Figure 7: Example visual attentions, module selections, and outputs for an antonymous pair of questions. The top two rows are from HybridNet trained only on VQA v2, while the bottom two rows are from HybridNet trained with Q3R.

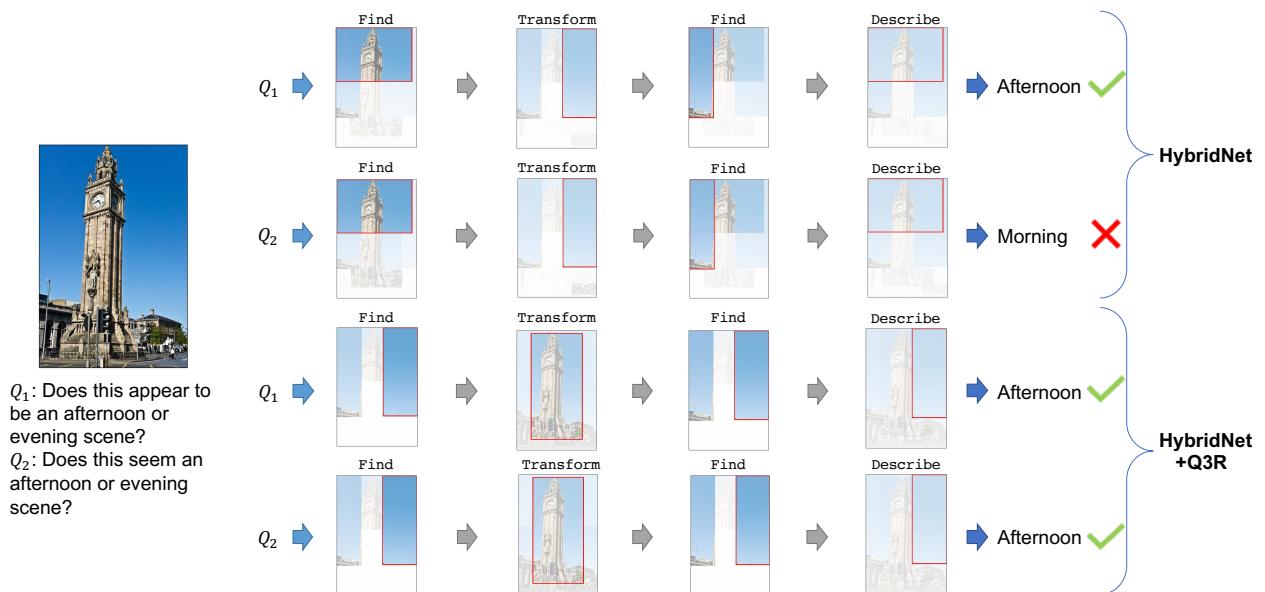


Figure 8: Example reasoning outputs for a paraphrastic pair of questions. Again, the top two rows are from HybridNet trained on VQA v2 and the bottom two are from HybridNet+Q3R.