Supplementary: Sunny and Dark Outside?! - Improving Answer Consistency in VQA through Entailed Question Generation

Anonymous EMNLP-IJCNLP submission

Abstract

In this supplementary document, we list dataset construction details, training details, and qualitative examples from our datasets and consistency teacher module outputs.

1 Logic-ConVQA Dataset Creation

We use scene graph annotations from the Visual Genome Dataset and slot-filler NLP techniques to generate a dataset of consistent QA sets (**L-ConVQA**). Currently, we only focus on attribute, existential and relational consistency. We generate groups of questions phrased differently about a certain concept to make consistent QA sets. For example, for the attribute “white” of object “cup” in the Visual Genome scene graph, we generate “is the cup white? Yes”, “Is the cup black? No” and “What color is cup? White”. Here is a summary of our consistent sets:

Relational/Existential Consistency

- Is <object> <relation> <subject>? Yes. For example, is man standing near tree?, Yes
- Is there <object>? Yes, For example, is there man? Yes.
- Is there <subject>? Yes
- Who/What is <relation> <subject>? <object>. For example, Who is standing near tree? Man
- Is <other object> <relation> <subject>? No, Is <object> <relation> <other subject>? No. We cross verify with scene graph to make sure these are “no”. However, the scene graph isn’t exhaustively annotated for all images and hence, these maybe noisy sometimes.

Attribute Consistency

- What hypernym(<attribute>) is <object>? <attribute>. For example, “What color is cup? White”. We get hypernyms using WordNet.
- Is <object> <attribute>? Yes
- Is <object> opposite(<attribute>)? No.
  We get opposite attributes using WordNet.

Some WordNet hypernyms and opposites are noisy, so we manually generate a list of opposites for some adjectives or action words. We also observe that counting questions are often noisy because of annotations not being exhaustive and non-countable objects being annotated, hence, we skip it. We also randomly substitute “can you see” or “do you see” in place of “is there” to have diversity in questions and make them more natural sounding. We also filter by at least 15% area of bounding box to image to make sure the questions are about salient objects in the image.

2 Training Details

We implement all our Consistency Teacher Module (CTM) networks using PyTorch (Paszke et al., 2017). We use a learning rate of $1e-5$ for all our models and we use the Adam (Kingma and Ba, 2014) method for optimization.

As mentioned in the main paper, CTM consists of two submodules - Question Generator that generates similar-intent question from GT QA and Consistency Checker that evaluates whether answer to generated question in consistent to GT QA or not.

2.1 Question Generator

Question Generator first concatenates the deep features of the image and concatenated QA into an embedding. Image features are obtained us-

Table 1: Performance comparison of baseline VQA trained on VQA2.0, baseline VQA finetuned on ConVQA, and CMT. For commonsense-based ConVQA, CMT produces the best results in terms of accuracy and consistency.

	DATA	L-ConVQA			CS-ConVQA				
		Perf Con	**Avg Con**	**Top1**	**Perf Con**	**Avg Con**	**Top1**	**Yes/No**	**Num**
a) VQA	VQA2.0	36.25	71.36	70.34	26.13	59.61	60.03	65.49	31.39
b) FineTune	CS-ConVQA	34.54	70.39	69.48	26.39	59.65	60.07	65.80	35.92
c) FineTune	L-/CS-ConVQA	**54.68**	**83.42**	**83.16**	24.70	59.30	59.60	65.14	33.33
d) **+CTM**	L/CS-ConVQA	54.6	83.23	82.79	25.94	**60.39**	**60.78**	**66.63**	**36.89**
e) FineTune	L-/CS-ConVQA,VG	36.40	71.60	70.94	25.22	59.19	59.56	65.30	31.39
f) **+CTM**	L/CS-ConVQA,VG	47.91	80.26	79.95	26.52	**60.30**	**60.66**	**66.60**	35.92
g) **+CTMvg**	L/CS-ConVQA,VG	51.41	81.66	81.37	**27.49**	59.75	60.15	66.41	34.95

ing a ResNet152 (He et al., 2016). QA features are obtained using an embedding layer for each word in the question which is fed into a 1-layer question-encoder LSTM (Hochreiter and Schmidhuber, 1997). We take the last output of the question-encoder LSTM and concatenate that with the deep image features. These concatenated features are then fed into another 1-layer LSTM to generate a similar-intent question. The output LSTM is trained using teacher forcing and a cross entropy loss. Top-5 probability-weighted random-sampling is used during evaluation. The ResNet152 Image encoder is pre-trained on ImageNet and is kept frozen during training. The question generator is trained only on L-ConVQA for module refered to as **CTM**. For the module refered to as **CTMvg**, the question generator is trained on a mix of L-ConVQA and Visual Genome. When adding Visual Genome in the training for **CTMvg**, we just add the Visual Genome QA pairs corresponding to the same images as the L-ConVQA train set.

2.2 Consistency Checker

Consistency Checker evaluates the consistency of the original and the generated QA pairs and classifies them into three categories- consistent, contradictory, or unrelated. It uses a ResNet152 (He et al., 2016) and LSTM's (Hochreiter and Schmidhuber, 1997) to encode image and QA features similar to the Question Generator. The concatenated features are then passed to a 3-layer neural network with hidden neuron sizes of 1024, 512 and 256 for predicting the three classes. For both **CTM** and **CTMvg**, the consistency checker is trained using only the L-ConVQA training set augmented with selected inconsistent/unrelated pairs. Inconsistent/unrelated pairs are produced by simple techniques- changing the answer word, flipping yes/no answers, replacing entities in the scene graph triplets, and generating unrelated questions from different triplet for any one question in a pair of two consistent QA pairs.

2.3 Reinforcement-based training

We use a mix of CS-ConVQA, Logic-ConVQA and Visual Genome questions to seed our question generator. We answer the generated question using the VQA. We only positively reward examples where the consistency classifier prediction is above 90% for consistent class and the VQA confidence is above 70%. VQA Confidence is effective at weeding out some questions that are non-grammatical or irrelevant.

3 Quantitative Results

In the main paper, we report results for **CTMvg** on the L/CS-ConVQA,VG dataset. We also tried applying **CTM** (the module where question generator was trained only on L-ConVQA). We still see improvements in consistency and accuracy over the fine-tuned baseline (row f vs e).

Since the choice of seed QA pair is random, there are slight fluctuations in the numbers across multiple runs. However, we almost always see similar gains of CTM compared to the fine-tuned baselines when checkpoints are chosen by best validation accuracy around 11k to 12k batch iterations of batch size 8. The numbers reported were the first observed numbers when we ran the experiments. Checkpoints and code will be uploaded publicly.

4 Qualitative Results

In the pages below, we list qualitative results of our datasets - Logic-ConVQA (Figure 1) and CommonSense-ConVQA (Figure 2). We also list example outputs of our similar-intent question generator (Figure 3), consistency checker (Figure 4), Consistency Teacher Module (CTM) based training (Figure 5) and our improved VQA model compared to the baseline VQA (Figure 6).

References

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Figure 1: Qualitative examples from our automatically generated logic-based consistent VQA dataset (L-ConVQA). We show two sets per image- an attribute-based set and a relation based set.

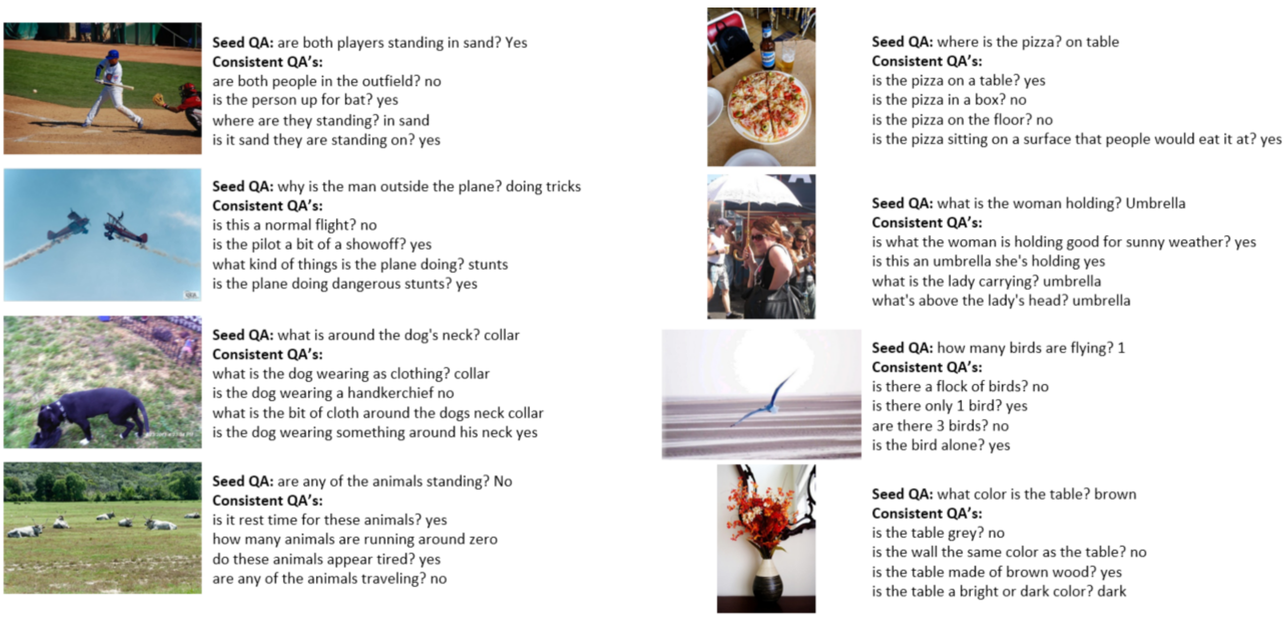

Figure 2: Qualitative examples from our human-annotated Common-Sense-based consistent dataset (CS-ConVQA).

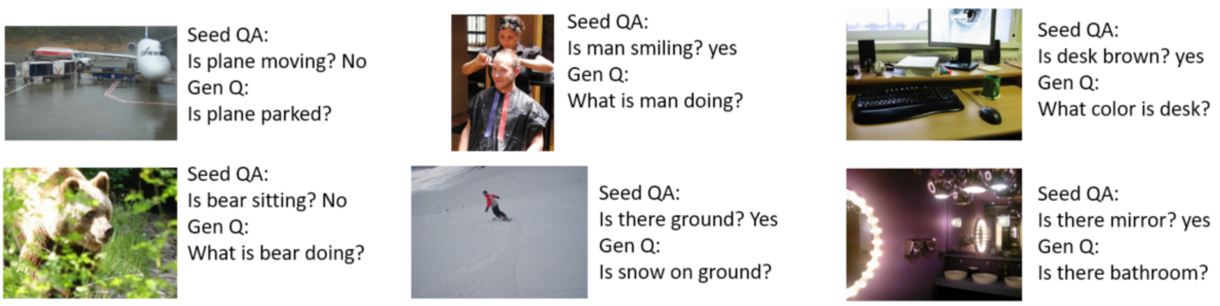

Figure 3: Qualitative examples of our similar-intent question generator outputs. Seed QA is the seed question-answer pair input to the generator along with the image and the Gen Q is the generated question.

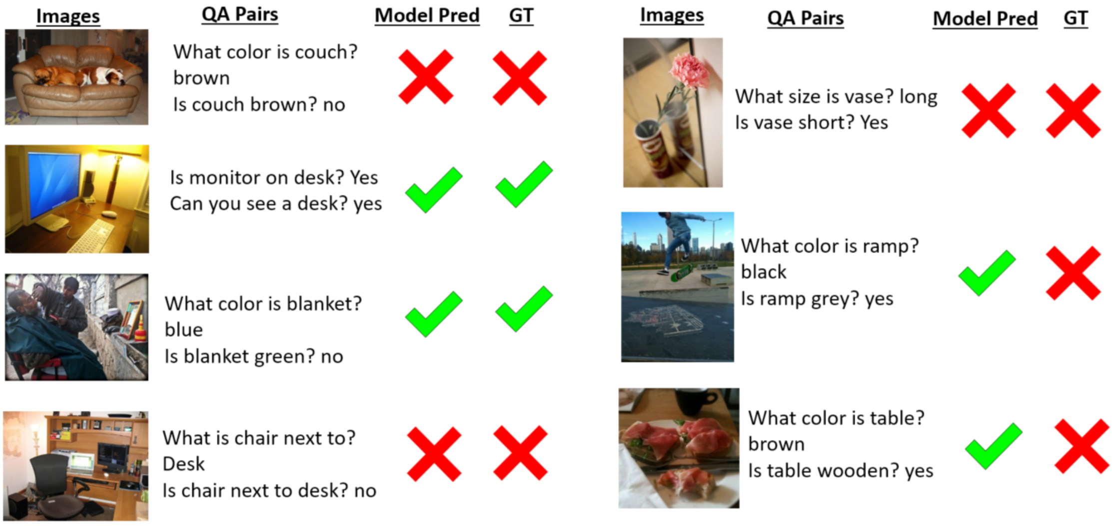

Figure 4: Qualitative examples of our consistency checker performance. GT is ground truth.

Figure 5: Examples of training using CTM. Gen QA is question generated by our CTM question generator and answered by VQA. Con Checker is whether our consistency checker deemed it as consistent. Incorrect reject was when the Con Checker deemed the question as unrelated or the VQA had low confidence. Note that in the bottom right image, the con checker understandably fails because it mistakenly thinks the sport is baseball.

Figure 6: Examples of our improved VQA consistency.