Supplementary Material: RMLVQA: A Margin Loss Approach For Visual Question Answering with Language Biases

Abhipsa Basu
abhipsabasu@iisc.ac.in

Sravanti Addepalli
sravantia@iisc.ac.in

R. Venkatesh Babu
venky@iisc.ac.in

Vision and AI Lab, Indian Institute of Science, Bangalore

This work introduces an angular adaptive margin loss (RMLVQA) to mitigate the language bias problem in VQA, ensuring stable performance in both in-domain and out-of-domain test data. The adaptive margins are determined in two ways: 1) From the frequency of answers in the training data, 2) From model confidence for each sample, obtained parallely during training. As demonstrated in the main paper, our method achieves the state-of-the-art scores on the benchmark VQA-CP v2 test data [1], while maintaining stable performance on the in-domain VQA v2 validation set [3] – making our proposed approach the *most* robust among all non-augmentation based methods. In this supplementary material, we present the details of the training procedure, followed by an analysis of the hyperparameters. Further, we combine our method with another existing margin loss based method AdaVQA [8] to show the flexibility of our method with respect to the underlying margin loss. We also show detailed analysis on the effects of changing the dataset as well as the base architecture. Finally, we conclude with more qualitative examples and an analysis on fair evaluation of our model, as suggested by Teney et al. [16].

A. Experimental Details

A.1. General Implementation Details

In this subsection we present some of the details of the base model that we used for our method RMLVQA. For all our experiments, we use UpDn [2] as the base network. Following previous works [4, 11, 13, 17, 19], we use a Faster RCNN model [14] pretrained by Anderson et al. [2] to extract the top 36 visual object feature vectors, each of dimension 2048. All the questions are tokenized into tokens of length 14. Each question word is encoded by Glove vectors [12] of dimension 300. These embeddings are passed on to a single layer GRU [5] to obtain the final question feature vector, which is of dimension 1024. We perform our experiments on a NVIDIA GeForce RTX 2080 Ti GPU. All our implementations are in PyTorch. The datasets VQA-CP v1 and v2 can be downloaded from https://computing.ece.vt.edu/~aish/vqacp/, VQA v2 [7] can be downloaded from https://visualqa.org/download.html, and GQA-OOD [10] can be downloaded from https://cs.stanford.edu/people/dorarad/gqa/index.html.

A.2. Implementation Details of RMLVQA (for VQA-CP v2)

We train *RMLVQA-Base* (i.e. the vanilla angular margin loss with only the frequency-based margins) and *RMLVQA* (our final method) for 30 epochs, with a learning rate of 0.001. The batch size is kept as 512, and Adamax is used for optimization. We list the hyperparameters and their values used by our models below:

- Entropy threshold: 4.5 (defined in Section G).
- Scaling parameter $\mathbf{s}$: 16.
- Standard deviation for the randomization of margins, σ: 0.1
- Temperature τ used for generating the instance-based margins and ensemling the logit heads during inference: 0.2
- Ensemble weight α: 0.5

During the first 15 epochs of training, we use the randomized frequency based margins only (Recall m_{ran} from main paper). Since the instance-level margins are learnt from model confidence, we emperically find it to be more useful to use them after the 15^{th} epoch. Therefore, from epoch $0-14$, we set $\beta = 1$ and from epochs $15-29$, it is set to be 0.9, where β is the weight used to combine the frequency based margins with the instance-based margins. We analyse the accuracy of VQA-CP v2 for various values of σ, τ and s, as shown in Fig. 1, and choose the final values with the help of these plots.

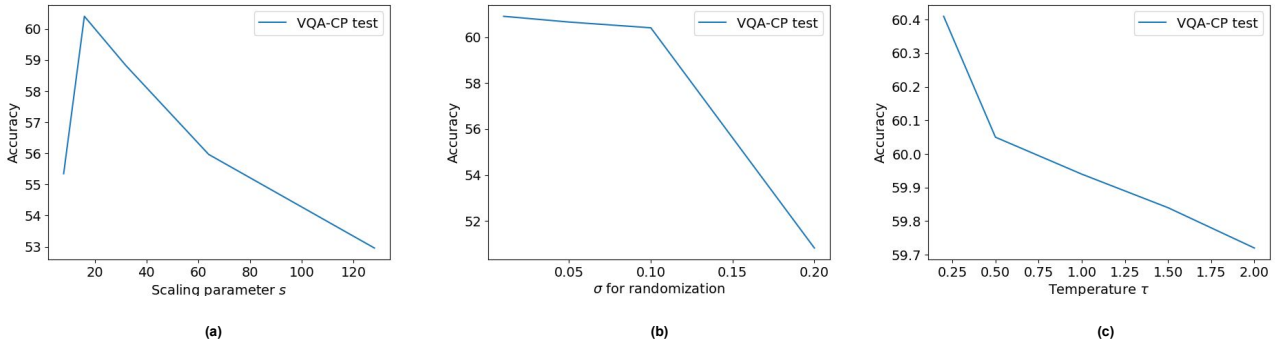

Figure 1. **Hyperparameter Analysis**. In this figure we present the variation of accuracies of VQA-CP v2 test set for given choices of hyperparameters: a) randomization standard deviation σ, b) temperature τ, c) scaling parameter s.

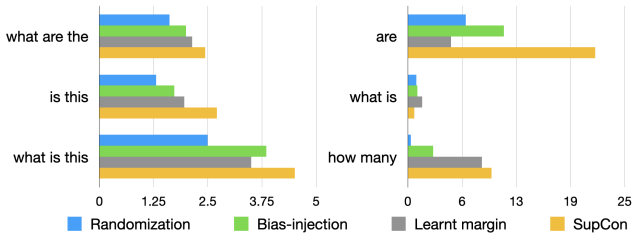

Figure 2. **Question-type wise Analysis**. In this figure we show model performance gap for each model component, for 6 randomly chosen question types with respect to RMLVQA-Base.

Table 1. **Ablations.** In this table we show another view of the ablations of ***RMLVQA***, evaluated on VQA-CP v2 test & VQA v2 val.

Model	VQA-CP v2 Test	VQA v2 Val
RMLVQA	60.41	59.99
-Randomization	60.77	58.08
-Bias Injection	59.16	59.61
-Learnable Margin	59.55	60.93
-SupCon Loss	59.87	59.60
-Backprop	58.80	59.67

A.3. Question-type wise Analysis

VQA-CP v2 has 65 question types. We use the question type information for the calculation of the frequency-based margins, as shown in Eq. 5 and 6 in the main paper. However, this information is not used in the bias-injecting component or the learnable margin calculation. The SupCon loss only uses the ground truth answers as supervision.

To demonstrate the effectiveness of our model, we show the performance *gap* of each model component for 6 randomly chosen question types in Fig. 2, with respect to the vanilla RMLVQA-Base.

A.4. Role of SupCon Loss in RMLVQA

In order to concretely understand the role of the SupCon loss, we add it to the baseline (i.e. the base UpDn architecture trained by CE loss). We find that it decreases baseline accuracy by 0.3%. This happens as it is calculated based on answers and the dataset is infact biased on the question types. Tables 3 (main paper), 1 and 2 show that this loss aids our model in both ID and OOD cases, irrespective of the underlying margin loss. We hypothesize that while the margin loss creates discriminative features for frequent and rare answers within a question type, the SupCon loss keeps similar answers together, creating subclusters inside a question type in the feature space. This aids the margin loss to further separate frequent and rare answers. This is validated by adding it with RMLVQA-base (the vanilla margin loss), which leads to an increase in accuracy by $\mathbf{2}\%$ on VQA-CP v2.

A.5. Further Studies on Ablations

In Sec 4.2 in the main paper, we show how the individual model components help the final performance of **RMLVQA** in an additive manner. Here, we remove one component from the main model, and show the effect of this removal in Table 1 on both VQA-CP v2 and VQA v2. This gives us further insights about the use of each of them for *id* and *ood* performance. We note that the bias-injecting component and the SupCon loss aid the model for both the datasets. While randomization of the frequency-based margins helps the model generalize to *id* data, the learnable margins boost *ood* performance. Overall, our final model is robust, with equivalent performance on *id* and *ood* data.

Table 2. **Ablations on AdaVQA.** In this table we show the role that is played by each component of *RMLVQA* when the underlying margin loss is *AdaVQA*, evaluated on VQA-CP v2 test data.

Model	Y/N	Num	Others	Overall
AdaVQA	70.83	49.00	46.29	54.02
+ Randomization	73.17	53.94	46.04	55.78
+Bias injection	83.58	41.70	47.32	57.30
+ Learnable Margin	85.40	42.36	47.52	57.94
+ SupCon	86.92	40.98	47.77	58.45

B. Applying Our Method on AdaVQA

In this section, we demonstrate the flexibility of our method with respect to the underlying margin loss. While we use a margin loss with angular margin penalties in the main paper, AdaVQA [8] – the only margin loss based method addressing the language bias problem in VQA – uses a cosine margin penalty. We add each component of our method *RMLVQA* to AdaVQA and show the effectiveness of each in Table 2. We call the final model with AdaVQA as the underlying margin loss (after adding all components of *RMLVQA* to AdaVQA) AdaVQA$_{\text{RMLVQA}}$. It is specifically to be noted that for all ablations on AdaVQA$_{\text{RMLVQA}}$, we set the scaling parameter $s = 16$ even though Guo et al. [8] set it to be 32, as we observe that it generalizes better to the VQA-v2 validation set with the former value. We show that similar to the original *RMLVQA*, (as shown in Table 3 in the main paper), each component of AdaVQA$_{\text{RMLVQA}}$ aids the cosine margin loss. The final accuracy is 58.45% for VQA-CP v2 test data, which is 4.43% higher than AdaVQA, owing to the addition of the Gaussian randomization, bias-injecting module, learnable margins, the supervised contrastive loss and the inference stage ensembling (which is done for every ablation after the introduction of the bias-injecting component). We further show a comparison between the original *RMLVQA* and AdaVQA$_{\text{RMLVQA}}$ in Table 3, where show the accuracy values for both VQA-CP v2 test and VQA v2 validation sets – we observe the effectiveness of the angular margins, as the original *RMLVQA* dominates AdaVQA$_{\text{RMLVQA}}$ by 1.96% for the *ood* data and 0.89% for the *id* data. However, it is to be noted that both methods are highly robust, with almost similar performances in *id* and *ood accuracies*. Specifically, on combining our method with AdaVQA, the *id* accuracy increases by 12.12%, which is a considerable gain. This makes AdaVQA suitable for both *id* and *ood* generalization, thus solving the primary problem of the method. We still recommend using the angular margins in the underlying margin loss because of their dominance over the cosine margins, as evident from Table 3.

C. Performance of RMLVQA on Other Datasets

C.1. VQA-CP v1

This dataset is a reorganization of the VQA v1 [3] training and validation splits, created in a similar fashion as VQA-CP v2. We show the performance of our method in Table 4. We observe that our ensemble based model (i.e. RMLVQA) outperforms the vanilla model (i.e. RMLVQA-Base), similar to VQA-CP v2 and VQA v2, and *RMLVQA* achieves state-of-the-art performance with respect to all augmentation-free methods, with a final overall accuracy of 63.52%. It is to be noted that we keep the hyperparameters similar as those used for VQA-CP v2 due to lack of a validation set.

C.2. GQA-OOD

This is another popularly used benchmark [10] to evaluate a VQA model's performance on out-of-distribution (*ood*) data. Unline VQA-CP v1 & v2, GQA-OOD does not explicitly carry the question type information, and hence we cannot estimate the frequency based margins m_{freq} directly in this dataset. Instead, we calculate them without any question type information in the following way:

$$\bar{m}_{freq}[i] = \frac{n_i + \epsilon}{\sum_{j=1}^{|\mathcal{A}|} n_j + \epsilon} \tag{1}$$

$$m_{freq}[i] = 1 - \bar{m}_{freq}[i] \tag{2}$$

where $\bar{m}_{freq}[i]$ measures the probability of occurrence of answer a_i in the *entire* training data. n_i is the frequency of answer a_i in the training set and ϵ is a hyperparameter for avoiding computational overflow. $m_{freq}[i]$ is the adaptive margin for answer $a_i \in \mathcal{A}$ corresponding to the entire dataset. Thus, our algorithm no longer has any assumption on the bias present in the dataset. We demonstrate the results in Table 5, where we show that the vanilla RMLVQA-Base outperforms the baseline BUTD [2] by 1.8%, and our proposed model surpasses both BUTD and the vanilla model by 2.67% and 0.86% respectively. Thus, our method successfully generalizes to this dataset without accessing knowledge of the question types. We also note that our model performs well on the tail classes alongside the head classes as compared to the other methods listed in Table 5. With the help of the provided validation set, we choose the hyperparameter values to be the following: $\sigma = 0.0001$, $\tau = 2.0$, $s = 16$, epochs=10. The weighting parameter β used to combine the frequency based margins with the instance-based margins is set to 1.0 till the 6^{th} epoch, and 0.9 from epoch 7. [R1]

Table 3. **Accuracy comparisons between the angular and the cosine margin losses** on the VQA-CP v2 test and VQA v2 validation sets with respect to different answer types. We observe that the angular margins outperform the cosine ones on both *id* and *ood* data, in their vanilla form and in the final form.

Model	Y/N-CP	Num-CP	Others-CP	Overall-CP	Y/N	Numbers	Others	Overall
AdaVQA	70.83	49.00	46.29	54.02	47.78	34.13	51.14	46.98
$\text{AdaVQA}_{\text{RMLVQA}}$	86.92	40.98	47.77	58.45	75.10	**37.72**	53.04	59.10
RMLVQA-Base	79.78	**49.62**	48.49	57.24	51.99	31.15	50.94	49.72
RMLVQA	**89.98**	45.96	**48.74**	**60.41**	**76.68**	37.54	**53.26**	**59.99**

Table 4. **Accuracy comparisons with the state-of-the-art methods on the VQA-CP v1 dataset** with respect to different answer categories. The best performance in each cateogory is highlighted in bold. * denotes numbers shown by Guo et al. [8]

Model	Y/N	Numbers	Others	Overall
SAN [18]	35.34	11.34	24.7	26.88
GVQA [1]	64.72	11.87	24.86	39.23
UpDn [2]*	43.76	12.49	42.57	38.02
S-MRL [4]	41.96	12.54	41.35	37.13
AdvReg [13]	74.16	12.44	25.32	43.43
RUBi [4]	75.00	13.3	30.49	46.08
LMH* [6]	76.61	29.05	43.38	54.76
RMLVQA-Base	90.31	**40.07**	**46.32**	62.71
RMLVQA	**91.24**	38.55	45.52	**63.52**

Table 5. **Accuracy comparisons with the state-of-the-art methods on GQA-OOD dataset** with respect to head, tail and overall answers. The best performance in each cateogory is highlighted in bold.

Model	Acc-All	Acc-Tail	Acc-Head
BUTD [2]	46.4	42.1	49.1
RUBi+QB	46.7	42.1	49.4
RUBi [4]	38.8	35.7	40.8
LM [6]	34.5	32.2	35.9
BP [6]	33.1	30.8	34.5
RMLVQA-Base	48.21	42.43	51.76
RMLVQA	**49.07**	**44.5**	**51.88**

D. Effect of Using Different Baselines

All our experiments are generally done with UpDn [2] as the base network. In this section, we demonstrate the performance of our model on other baselines like SAN [18], S-MRL [4] and LXMERT [15]. We show in Table 6 that our proposed method outperforms the respective baseline for each case. For LXMERT, we refer Guo et al. [9] for the overall accuracy. The results show that our method is independent of the underlying base network. It is to be noted that LXMERT is a popular vision-language *pretrained* model, frequently used for several multimodal downstream tasks. We demonstrate that our method is able to improve the performance of LXMERT by 9.17%.

Table 6. **Performance of our model on three different network architectures**, evaluated on VQA-CP v2 test data.

Base	Y/N	Num	Other	All
SAN	35.34	11.34	24.7	26.88
SAN+RMLVQA	88.56	42.53	46.10	58.02
S-MRL	42.85	12.81	43.20	38.46
S-MRL+RMLVQA	90.02	47.69	47.15	58.45
LXMERT	-	-	-	58.07
LXMERT+RMLVQA	95.40	70.68	51.68	67.24

E. More Qualitative Results

We present some qualitative results in Fig. 3 with respect to *RMLVQA* on VQA-CP v2, to show the effectiveness of the bias-injecting component in our model. For example, in Fig. 3(a), we see that the vanilla model predicts "bus" for a "yes/no" type question, whereas the randomized margin trained model predicts a wrong answer, but the answer is relevant to the question type of the given question. Fig. 3(a),(b),(c) show the effectiveness of the addition of the bias-injecting component, wherein it is able to output relevant and correct answers for each of the samples. In Fig. 3(d) however, we see that all the model components answer "giraffe" even though the question is of "yes/no" type. Further qualitative examples are shown to demonstrate the effectiveness of the instance-based margins in Fig. 4. For each question shown in this figure, the correct answer is a frequently available class in the training set. However, the baseline still outputs wrong answers for each of them, whereas our method effectively corrects the errors made by the baseline. We believe that this happens as the learnable margin component ensures more margin penalties for such

Algorithm 1 Train

Input: Training set divided into B mini-batches along with the frequency based margins, model M to train
Output: Trained model

1: M.train().
2: **for** b in B **do**
3: **for** (v, q, a, m^b_{freq}) in b **do**
4: f, f_b, $\mathbf{x}$ = M(v, q)
5: $m^b_{ran} = \mathcal{N}(m^b_{freq}, \sigma)$
6: $m^b_{ins} = softmax(f_b/\tau)$
7: m^b_{comb} = combine(m^b_{freq}, m^b_{ins})
8: $L = L_{Angular}(m_{comb}) + L_s(f_b, a) + L_{sup-con}(\mathbf{x}, a)$
9: L.backward()
10: **end for**
11: **end for**
12: **return** M

frequent but hard samples.

Fig. 5 shows further examples which demonstrate the incremental changes in performance on the VQA-CP test data, after addition of the individual components of the model, i.e. 1) RMLVQA-Base, 2) Rand (RMLVQA-Base + Randomization of margins), 3) BI (Rand + the Bias-injecting component), 4) LM (BI + instance-level margins).

F. Psuedo Code of Our Model

In this section, we present the pseudo codes of our entire algorithm. In Algorithm 1 we show how our model is trained. For each minibatch b consisting of some image v, question q, answer a and the frequency-based margins m^b_{freq} (computed for the question type of q), we pass the image and the question to the model M, which returns the primary logits f, the logits from the bias-injecting component f_b, and the multimodal feature $\mathbf{x}$ (lines 2-4). We compute the randomized and instance-based margins in lines 3 and 4, and combine the two in line 7 using the procedure mentioned in Subsection 3.2 in the main paper. Finally, we compute the loss function using the three losses as mentioned in Section 3 of the main paper, and backpropagate through it to update the model (lines 8-9).

Algorithm 2 shows how the trained model is used for prediction on a trained/validation data, following the inference strategy mentioned in Section 3 of the main paper. Recall that α refers to the ensemble weight, $\hat{p}$ refers to the prediction probabilities from the primary classifier c, and $\hat{p_b}$ refers to the prediction probabilities from the bias-injecting component c_b.

Algorithm 2 Evaluate

Input: Test/Validation set divided into B mini-batches along with the frequency based margins, trained model M

1: M.eval().
2: **for** b in B **do**
3: **for** (v, q, a, m^b_{freq}) in b **do**
4: f, f_b, $\mathbf{x}$ = M(v, q)
5: pred = $\alpha.\hat{p} + (1-\alpha).\hat{p_b}$
6: **end for**
7: **end for**
8: **return** M

G. Entropy Threshold for Setting Margins

In VQA-CP, not all question types suffer from the language bias equally. For such question types, adaptive margins are not required. It suffices to use a constant margin for these question types, whereas for others, adaptive margins need to be used. Inspired by AdaVQA, [8], entropy is used to distinguish which question types in the training set exhibit the language bias more than others. For example, let qt_k be a certain question type. Its entropy is calculated as following:

$$e_{qt_k} = -\sum_{i=1}^{|\mathcal{A}|} \bar{m}^k_{freq}[i]\ log_2\ \bar{m}^k_{freq}[i]$$

A higher value of entropy suggests a more uniform distribution whereas a lower value points at non-uniformity. This means that the question types with higher entropy do not exhibit language biases. Therefore, for all questions in the training data that belong to a certain question type whose entropy is higher than a threshold, we set the margin values as constant (more specifically, 1). For others, we calculate the margins in the same way as shown in Subsection 3.2 of the main paper.

H. Ensuring a Fair Evaluation.

We follow the recommendations by Teney et al. [16] for fair handling of the VQA-CP dataset:

- We randomly extract 8000 samples from VQA-CP v2 training data to evaluate how the same trained model performs on an *id* validation data and an *ood* test set. In Table 8, we compare the performances of our method with respect to RandImg [16], AdaVQA and *RMLVQA-Base*. We observe that while both *RMLVQA* and AdaVQA$_{\text{RMLVQA}}$ achieve competitive performance on the *id* validation set as compared to RandImg, they surpass the latter on VQA-CP test (5.04% and 2.5% respectively) and VQA v2 validation sets (2.75% and 1.86% respectively). More im-

Figure 3. **Effectiveness of the bias-injecting component**. In this figure we present examples from the VQA-CP v2 test data, where *RMLVQA-Base* and *RMLVQA-Base + Gaussian Randomization* both predict irrelevant answers and the addition of the bias-injecting component (BI) corrects them for most of the cases. BI: *RMLVQA-Base* + Gaussian Randomization + Bias-injecting component.

Figure 4. **Effectiveness of the learnt instance-based margins**. In this figure we present examples from the VQA-CP v2 test data, where the baseline *UpDn* fails to predict answers which are frequent. This is corrected by our proposed model by flexibly changing margins for easy and difficult samples in the training set.

portantly, *RMLVQA* outperforms its vanilla version by 12.96%, while AdaVQA$_{\text{RMLVQA}}$ outperforms *AdaVQA* by 16.73% on the *id* validation set. This shows how our final model overcomes the problem of overfitting to the VQA-CP test distribution, and becomes more robust. We believe that it is unsuitable to select the ensemble weight α by looking at the *id* validation and particularly test accuracies. Hence, we set $\alpha = 0.5$ to obtain our final results for both the margin losses as mentioned in the main paper. However, if some knowledge of the test distribution is available, one can vary α such that the performance is better on one of the datasets as shown in Table 2 in the main paper. We report the accuracies obtained by training the model on the *entire* training data in Table 1 in the main paper for a fair comparison with other methods.

- We also focus on the *other* type questions as they are much less likely to be answered correctly with "a naive or malfunctioning method". For both *RMLVQA* and AdaVQA$_{\text{RMLVQA}}$, we observe that the *other* type accuracies are comparable with all state-of-the-art methods, surpassing that of RandImg (45.99%).

- All experiments in Table 1 (main paper) are averaged over 5 seeds, ensuring that the accuracy of our model is not spurious.

In Table 7 (where we show the variation in accuracies for varying α for the in-domain VQA-CP v2 validation set, VQA-CP v2 test set, and the VQA v2 validation set), we

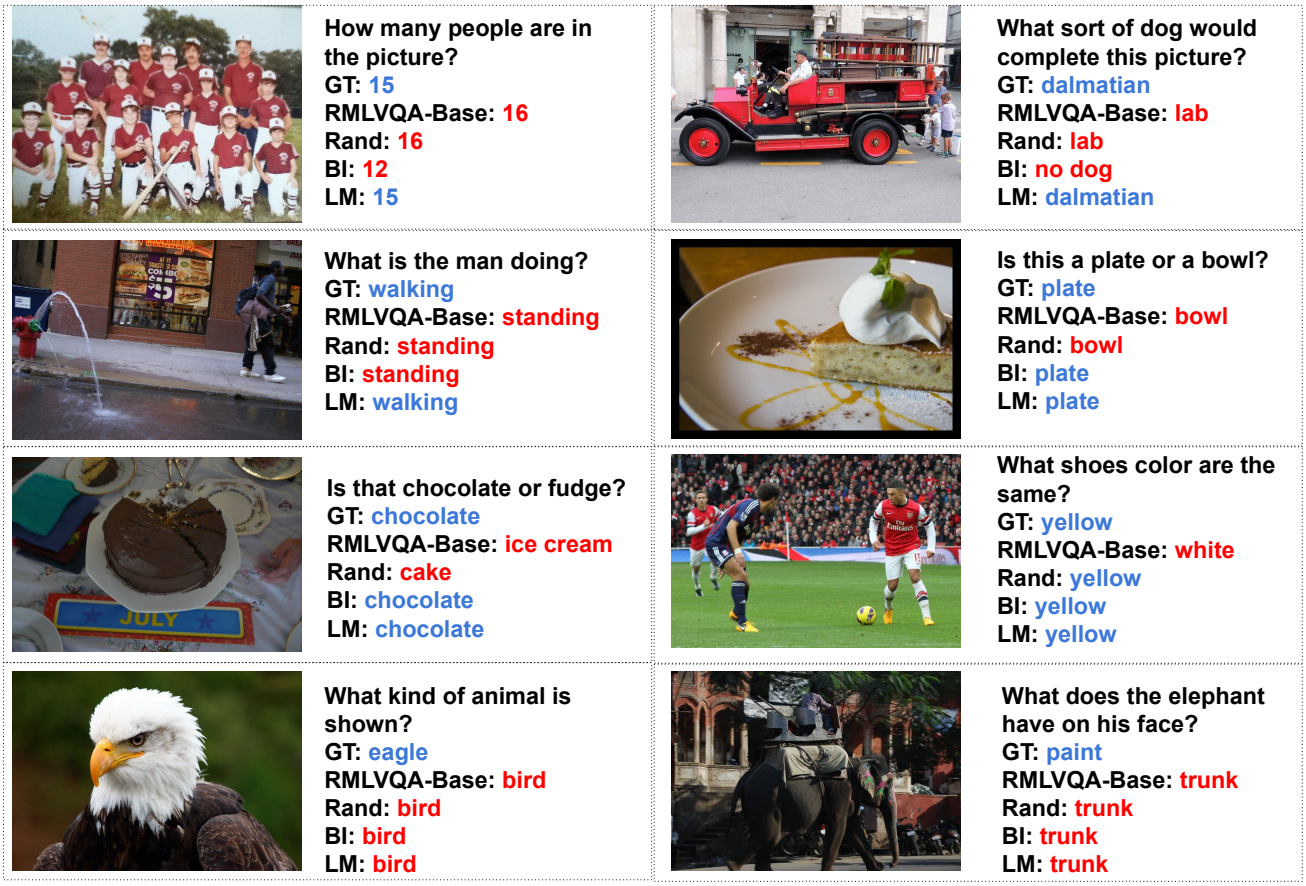

Figure 5. **Examples of samples from VQA-CP v2 test data**. In this figure we present examples from the VQA-CP test data, showing the performance of each model component of *RMLVQA* on these examples. Correct answers are marked in blue, wrong answers are marked in red. Rand: RMLVQA-Base + Gaussian Randomization, BI: Rand + Bias-injecting component, LM: BI + instance-level margins.

Table 7. **Role of α in the inference stage of RMLVQA.**

α	VQA-CP v2 Test	VQA-CP v2 in-dom Val	VQA-v2 Val
1.0	60.54	51.95	58.16
0.8	60.50	52.66	58.83
0.6	60.33	53.58	59.57
0.5	60.16	54.18	59.99
0.4	59.87	54.90	60.46
0.2	57.46	57.48	61.26
0.0	39.48	66.37	61.54

observe that the *id* validation accuracy of VQA-CP is proportional to that of VQA v2, suggesting a correspondence in the performances of both. We also show the performance of our model on the GQA-OOD dataset [10] in subsection C.2, where the margins are calculated without knowledge of the question type.

Table 8. **Comparison of different models on the VQA-CP v2 in-domain validation set.**

Model	VQA-CP val	VQA-CP test	VQA-v2 val
UpDn	65.36	41.74	63.48
RandImg	54.24	55.37	57.24
AdaVQA	36.66	53.59	47.64
RMLVQA-Base	41.22	57.11	49.72
AdaVQA$_{\text{RMLVQA}}$	53.39	57.87	59.10
RMLVQA	54.18	60.16	59.99

References

[1] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha 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, 4

[2] 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. 1, 3, 4

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015. 1, 3

[4] Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. *Advances in neural information processing systems*, 32, 2019. 1, 4

[5] 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. *arXiv preprint arXiv:1406.1078*, 2014. 1

[6] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out: ensemble based methods for avoiding known dataset biases. *arXiv preprint arXiv:1909.03683*, 2019. 4

[7] 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

[8] Yangyang Guo, Liqiang Nie, Zhiyong Cheng, Feng Ji, Ji Zhang, and Alberto Del Bimbo. Adavqa: Overcoming language priors with adapted margin cosine loss. *arXiv preprint arXiv:2105.01993*, 2021. 1, 3, 4, 5

[9] Yangyang Guo, Liqiang Nie, Zhiyong Cheng, Qi Tian, and Min Zhang. Loss re-scaling vqa: revisiting the language prior problem from a class-imbalance view. *IEEE Transactions on Image Processing*, 31:227–238, 2021. 4

[10] Corentin Kervadec, Grigory Antipov, Moez Baccouche, and Christian Wolf. Roses are red, violets are blue... but should vqa expect them to? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2776–2785, 2021. 1, 3, 7

[11] Gouthaman KV and Anurag Mittal. Reducing language biases in visual question answering with visually-grounded question encoder. In *European Conference on Computer Vision*, pages 18–34. Springer, 2020. 1

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 1

[13] Sainandan Ramakrishnan, Aishwarya Agrawal, and Stefan Lee. Overcoming language priors in visual question answering with adversarial regularization. *Advances in Neural Information Processing Systems*, 31, 2018. 1, 4

[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1

[15] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019. 4

[16] Damien Teney, Ehsan Abbasnejad, Kushal Kafle, Robik Shrestha, Christopher Kanan, and Anton Van Den Hengel. On the value of out-of-distribution testing: An example of goodhart's law. *Advances in Neural Information Processing Systems*, 33:407–417, 2020. 1, 5

[17] Zhiquan Wen, Guanghui Xu, Mingkui Tan, Qingyao Wu, and Qi Wu. Debiased visual question answering from feature and sample perspectives. *Advances in Neural Information Processing Systems*, 34, 2021. 1

[18] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29, 2016. 4

[19] Xi Zhu, Zhendong Mao, Chunxiao Liu, Peng Zhang, Bin Wang, and Yongdong Zhang. Overcoming language priors with self-supervised learning for visual question answering. *arXiv preprint arXiv:2012.11528*, 2020. 1