Guiding Visual Question Answering with Attention Priors - Supplementary Material

Thao Minh Le, Vuong Le, Sunil Gupta, Svetha Venkatesh, Truyen Tran
Applied Artificial Intelligence Institute, Deakin University, Australia
{thao.le,vuong.le,sunil.gupta,svetha.venkatesh,truyen.tran}@deakin.edu.au

1. Introduction

In this supplementary material, we provide extra details on the proposed Grounding as Attention Priors (GAP) method and analysis. They include:

- Implementation details of the Language and visual embedding (Sec 3 main text)
- Mathematical proof on the meaning of the attention refinement formulations (Sec 4.3 main text).
- Implementation of the Neural gating functions (Eqs. 12 and 13 main text).
- Experiment details including datasets and baselines.
- Additional experimental results.
- Additional qualitative analysis.

2. Language and Visual Embedding

Textual embedding Given a length-T question, we first tokenize it into a sequence of words and further embed each word into the vector space of 300 dimensions. We initialize the word embeddings with the popular pre-trained vector representations in GloVe [14].

To model the sequential nature of the query, we use bidirectional LSTMs (BiLSTMs) taking as input the word embedding vectors. The BiLSTMs result in hidden states $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ at a time step i for the forward pass and backward pass, respectively. We further combine every pair $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ into a single vector $l_i = \left[\overrightarrow{h_i}; \overleftarrow{h_i}\right]$, where $[;]$ indicates the vector concatenation operation. The contextual words are then obtained by gathering these combined vectors $L = \left\{l_i \mid l_i \in \mathbb{R}^d\right\}_{i=1}^{T}$. The global representation q of the query is a combination of the ends of the LSTM passes $q = \left[\overrightarrow{h_1}; \overleftarrow{h_S}\right]$.

For our grounding framework with contrastive learning, we use a contextualized word representation extracted by a pre-trained BERT language model [6] for each word w_i in an extracted RE. These contextualized embeddings are found to be more effective for phrase grounding [7].

Visual embedding Visual regions are extracted by the popular object detection Faster R-CNN [15] pre-trained on Visual Genome [12]. We use public code[1] making use of the Facebook Detectron2 v2.0.1 framework[2] for this purpose. For each image, we extract a set of N RoI pooling features with bounding boxes $\{(a_j, b_j)\}_{j=1}^{N}$, where $a_j \in \mathbb{R}^{2048}, b_j \in \mathbb{R}^4$ are appearance features of object regions and bounding box's coordinators, respectively. We follow [18] to encode the bounding box's coordinators into a spatial vector of 7 dimensions. We further combine the appearance features with the encoded spatial features by using a sub-network of two linear transformations to obtain a set of visual objects $V = \left\{v_j \mid v_j \in \mathbb{R}^{d'}\right\}_{j=1}^{N}$, where d' is the vector length of the joint features of the appearance features and the spatial features. For ease of reading and implementation, we choose the linguistic feature size d and the visual feature size d' to be the same.

3. Meaning of Attention Refinement Mechanisms

In this section, we will give a proof that our choices for attention refinement in Eqs. (10, 11 and 14) in the main paper are the optimal solutions with respect to some criteria for probability estimate aggregation.

Let us consider the generic problem where a system has multiple estimates $\{P_i(x)\}_{i=1}^{n}$ of a true discrete distribution $P(x)$ from multiple mechanisms with corresponding degrees of certainty $\lambda = \{\lambda_i\}_{i=1}^{n}$. We first normalize these certainty measures so that they sum to one: $\lambda_i \geq 0$, $\sum_i \lambda_i = 1$. We aim at finding a common distribution $P'(x)$ aggregating the set of distributions $\{P_i(x)\}$ subject to a item-to-set distance D:

[1] https://github.com/MILVLG/bottom-up-attention.pytorch
[2] https://github.com/facebookresearch/detectron2

$$P'(x) = \underset{P(x)}{\operatorname{argmin}} D_\lambda\left(P(x); \{P_i(x)\}_{i=1}^n\right). \quad (1)$$

This problem can be solved for particular choices of the set-distance function D that measures the discrepancy between $P(x)$ and the set $\{P_i(x)\}_{i=1}^n$ under the confidence weights $\lambda = \{\lambda_i\}_{i=1}^n$. We consider several heuristic choices of the function D:

Additive form:

If we define the distance to be Euclidean distance from $P(x)$ to each member distribution $P_i(x)$ of the set, then the minimized term D becomes

$$D_\lambda\left(P(x); \{P_i(x)\}_{i=1}^n\right) = \frac{1}{2}\sum_i \lambda_i \sum_x \left(P(x) - P_i(x)\right)^2. \quad (2)$$

Here, we minimize D_λ w.r.t. $P(x)$:

$$\partial_P D = \sum_i \lambda_i \left(P(x) - P_i(x)\right).$$

By setting this gradient to zero, we have $P'(x) = \sum_i \lambda_i P_i(x)$. This explains for the additive form of our attention refinement mechanism in Eqs. (10 and 14 (upper part)) in the main paper where we seek a solution that best agrees with the grounding prior and the model-induced probability .

Multiplicative form:

If we define D as the weighted sum of the KL divergences between $P(x)$ to each member distribution $P_i(x)$ of the set:

$$D_\lambda\left(P(x); \{P_i(x)\}_{i=1}^n\right) = \sum_i \lambda_i \mathrm{KL}\left(P(x) \parallel P_i(x)\right) \quad (3)$$

$$= \sum_i \lambda_i \sum_x P(x) \log \frac{P(x)}{P_i(x)}. \quad (4)$$

Here, we minimize a Lagrangian with the multiplier η of D_λ w.r.t. $P(x)$, $L = D + \eta\left(\sum_x P(x) - 1\right)$:

$$\partial_P L = \sum_i \lambda_i \left(\log \frac{P(x)}{P_i(x)} + 1\right) + \eta \quad (5)$$

$$= \log P(x) - \sum_i \lambda_i \log P_i(x) + 1 + \eta \quad (6)$$

$$= \log P(x) - \log \prod_i P_i(x)^{\lambda_i} + 1 + \eta. \quad (7)$$

Setting this gradient to zero leads to:

$$P'(x) = C \prod_i P_i(x)^{\lambda_i}, \quad (8)$$

where C is a calculable constant to normalize $P'(x)$ such that its components sum to one. This explains for the multiplicative form of our attention refinement mechanism in Eqs. (11 and 14 (lower part)) in the main paper.

4. Neural Gating Functions

Here we provide the details of implementation choices for the neural gating functions in Eqs. (12 and 13). In particular, we use element-wise product between embedded representations of the input $\overline{v} \in \mathbb{R}^d$ and $q \in \mathbb{R}^d$ as following:

$$\lambda = h_\theta(\overline{v}, q) \quad (9)$$

$$= \sigma\left(w_\lambda^\top \left(\mathrm{ELU}\left(\left(W_v^\top \overline{v} + b_v\right) \odot \left(W_q^\top q + b_q\right)\right)\right)\right), \quad (10)$$

where $w_\lambda \in \mathbb{R}^d$, $W_v \in \mathbb{R}^{d\times d}$, $W_q \in \mathbb{R}^{d\times d}$ are learnable weights, b_v, b_q are biases, σ is the sigmoid function and $\odot$ denotes the Hadamard product. ELU [5] is a non-linear activation function.

For multi-step reasoning, we additionally takes as input the intermediate controlling signal $c_k \in \mathbb{R}^d$ at reasoning step k. Output of the modulating gate in Eq. 13 in the main paper is given by

$$\lambda_k = p_\theta\left(c_k, h_\theta(\overline{v}, q)\right), \quad (11)$$

where

$$h_\theta(\overline{v}, q) = W_h^\top \left(\mathrm{ELU}\left(\left(W_v^\top \overline{v} + b_v\right) \odot \left(W_q^\top q + b_q\right)\right)\right), \quad (12)$$

$$p_\theta = \sigma\left(w_\lambda^\top \left(\mathrm{ELU}\left(c_k \odot h_\theta(\overline{v}, q)\right)\right)\right). \quad (13)$$

Here $W_h \in \mathbb{R}^{d\times d}$, $W_v \in \mathbb{R}^{d\times d}$, $W_q \in \mathbb{R}^{d\times d}$, $w_\lambda \in \mathbb{R}^d$ are learnable weights, and b_v, b_q are biases.

5. Experiment details

5.1. Datasets

VQA v2 is a large scale VQA dataset entirely based on human annotation and is the most popular benchmark for VQA models. It contains 1.1M questions with more than 11M answers annotated from over 200K MSCOCO images [13], of which 443,757 questions, 214,354 questions and 447,793 questions in train, val and test split, respectively.

We choose correct answers in the training split appearing more than 8 times, similar to prior works [17, 2]. We report performance as accuracy calculated by standard VQA accuracy metric: $\min(\frac{\#\text{humans that provided that answer}}{3}, 1)$ [3].

GQA is currently the largest VQA dataset. The dataset contains over 22M question-answer pairs and over 113K images covering various reasoning skills and requiring multi-step inference, hence significantly reducing biases as in previous VQA datasets. Each question is generated based on an

Model	VQA-CP2 test				VQA v2 val			
	Overall	Yes/No	Number	Other	Overall	Yes/No	Number	Other
UpDn baseline	40.6	41.2	13.0	48.1	63.3	79.7	42.8	56.4
UpDn+GAP	40.8	41.2	13.2	48.3	64.3	81.2	44.1	56.9
UpDn+RUBi	48.6	72.1	12.6	46.1	62.7	79.2	42.8	55.5
UpDn+RUBi+GAP	48.9	72.2	12.8	46.4	64.2	81.4	44.3	56.3

Table 1. Performance on VQA v2 val split and VQA-CP2 test split with UpDn baseline.

associated scene graph and pre-defined structural patterns. GQA has served as a standard benchmark for most advanced compositional visual reasoning models [9, 8, 10, 16]. We use the balanced splits of the dataset in our experiments.

5.2. Baseline Models

Bottom-Up Top-Down Attention (UpDn) UpDn is the first to introduce the use of bottom-up attention mechanism to VQA by utilizing image region features extracted by Faster R-CNN [15] pre-trained on Visual Genome dataset [12]. A top-down attention network driven by the question representation is used to summarize the image region features to retrieve relevant information that can be decoded into an answer. The UpDn model won the VQA Challenge in 2017 and became a standard baseline VQA model since then.

MACNet MACNet is a multi-step co-attention based model to perform sequential reasoning where they use VQA as a testbed. Given a set of contextual word embeddings and a set of visual region features, at each time step, an MAC cell learns the interactions between the two sets with the consideration of their past interaction at previous time steps through a memory. In particular, an MAC cell uses a controller to first compute a controlling signal by summarizing the contextual embeddings of the query words using an attention mechanism. The controlling signal is then coupled with the memory state of the previous reasoning step to drive the computation of the intermediate visual attention scores. At the end of a reasoning step, the retrieved visual feature is finally used to update the memory state of the reasoning process. The process is repeated over multiple steps, resembling the way humans reason over a compositional query. In our experiments, we use a Pytorch equivalent implementation[3] of MACNet instead of using the original Tensorflow-based implementation. We choose the number of reasoning steps to be 6 in all experiments. For experiments with GAP, we only refine the attention weights inside the controller (linguistic attention) and the read module (visual attention) at the first reasoning step where the grounding prior shows its best effect in accelerating the learning of attention weights, hence leads to the best performance overall.

Bilinear Attention Networks (BAN) BAN is one of the most advanced VQA models based on low-rank bilinear pooling. Given two input channels (language and vision in the VQA setting), BAN uses low-rank bilinear pooling to extract the pair-wise cross interactions between the elements of the inputs. It then produces an attention map to selectively attend to the pairs that are most relevant to the answer. BAN also takes advantage of a multimodal residual networks to improve its performance by repeatedly refining the retrieved information over multiple attention maps. We use its official implementation[4] in our experiments. In order to make the best judgments of the model's performance with our attention refinement with grounding priors, we remove the plug-and-play counting module [19] in the original implementation.

Regarding the choice of hyper-parameters, all the experiments regardless of the baselines are with d=512. The number of visual objects N for each image is 100 and the maximum number of words T in a query is set to be the length of the longest query in the respective dataset. We train all the models using Adam optimizer with a batch size of 64. The learning rate is initialized at 10^{-4} and scheduled with the warm up strategy, similar to prior words in VQA [11]. Reported results are at the epoch that gives the best accuracy on the validation sets after 25 training epochs.

6. Additional Experimental Results

Apart from the experimental results in Sec. 5 in the main paper, we provide additional results on VQA-CP2 dataset [1] to support our claim that GAP complements related works with regularization schemes. We choose RUBi [4] as a representative bias reduction method for VQA with a general linguistic debiasing technique and yet effective on VQA-CP2 dataset. Table 1 presents our experimental results with UpDn baseline. As clearly seen, even though linguistic biases are not the main target, GAP still shows consistent improvements on top of both UpDn baseline and UpDn+RUBi baseline. It is to emphasize that applying the regularization by RUBi for linguistic bias considerably hurts the performance on VQA v2 even though RUBi largely improves performance on VQA-CP2 test split. GAP brings the benefits of pre-computed attention priors and rejects the dam-

[3] https://github.com/tohinz/pytorch-mac-network

[4] https://github.com/jnhwkim/ban-vqa

age caused by the regularization effects by RUBi to maintain its excellent performance on VQA v2 while slightly improving the baseline's performance on VQA-CP2. Looking more closely at the results per question type on VQA-CP2 (Row 1 vs. Row 2, and Row 3 vs. Row 4), GAP shows its universal effect on all question types with the strongest effect on "Other" question type which contains open-ended arbitrary questions. On the other hand, RUBi (Row 3 vs. Row 1) shows its significant impact only on binary questions "Yes/No" but considerably hurts "Number" and especially "Other" question types. This reveals that the regularization scheme in RUBi is overfitted to "Yes/No" questions specifically due to the limitation of the data generation process behind this dataset.

The analysis in this Section is consistent with our results in Figure 4 in the main paper and is clearly evident to GAP's universal effects in improving VQA performance. The additional results with RUBi shown in this Section also state GAP's complementary benefits upon the use of the learning regularization methods targeting only a specific type of data as such in VQA-CP2.

7. Additional Qualitative Analysis

Fig.6 in the main paper provides one case of visualization on the internal operation of our proposed method GAP as well as its effect on VQA models. We provide more examples here for UpDn baseline (Fig. 1) and MACNet baseline (Fig. 2) with the same convention and legends.

In each figure, left subfigures present the linguistic-visual alignments learned by our unsupervised grounding framework. Right subfigures compare the visual attentions before and after applying GAP. In all cases across two different baselines (UpDn and MACNet), GAP clearly helps direct the models to pay attention to more appropriate visual regions, partly explaining their answer predictions.

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 *CVPR*, pages 4971–4980, 2018. 6

[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 *CVPR*, pages 6077–6086, 2018. 5.1

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *CVPR*, pages 2425–2433, 2015. 5.1

[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. 6

[5] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *International Conference on Learning Representations*, 2016. 4

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, 2018. 2

[7] Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. Contrastive learning for weakly supervised phrase grounding. In *Computer Vision–ECCV 2020*, pages 752–768. Springer, 2020. 2

[8] Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. Language-conditioned graph networks for relational reasoning. *ICCV*, 2019. 5.1

[9] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, pages 6700–6709, 2019. 5.1

[10] Drew A Hudson and Christopher D Manning. Learning by abstraction: The neural state machine. *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 5.1

[11] Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0. 1: the winning entry to the vqa challenge 2018. *VQA 2018 Challenge*, 2018. 5.2

[12] 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. *IJCV*, 123(1):32–73, 2017. 2, 5.2

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 5.1

[14] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014. 2

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 2, 5.2

[16] Violetta Shevchenko, Damien Teney, Anthony Dick, and Anton van den Hengel. Visual question answering with prior class semantics. *arXiv preprint arXiv:2005.01239*, 2020. 5.1

[17] Damien Teney, Peter Anderson, Xiaodong He, and Anton Van Den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *CVPR*, pages 4223–4232, 2018. 5.1

[18] Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L Berg. A joint speaker-listener-reinforcer model for referring expressions. In *CVPR*, pages 7282–7290, 2017. 2

[19] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning to count objects in natural images for visual question answering. *International Conference on Learning Representations*, 2018. 5.2

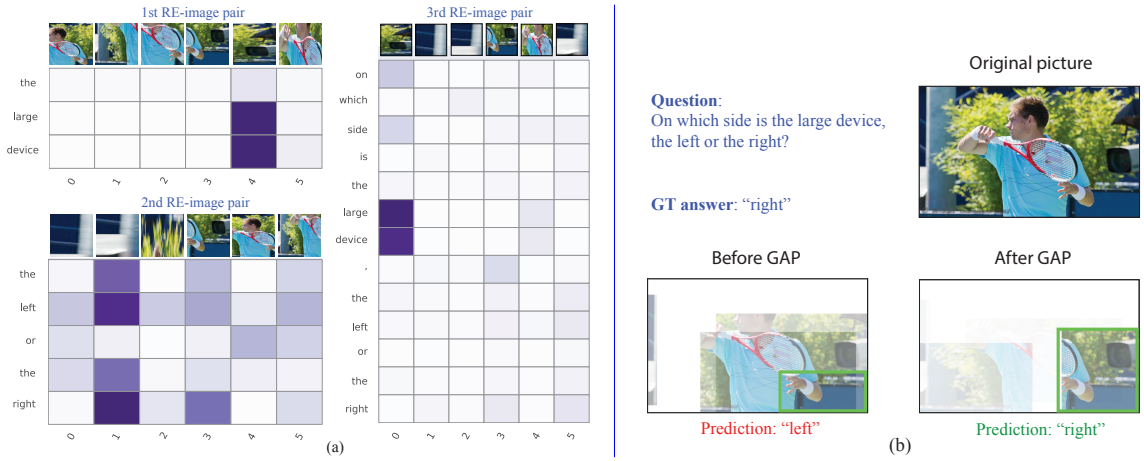

Figure 1. Qualitative analysis of GAP with UpDn baseline. (a) Region-word alignments of different RE-image pairs learned by our unsupervised grounding framework. (b) Visual attentions and prediction of UpDn model before (left) vs. after applying GAP (right). GAP shifts the model's highest visual attention (green rectangle) to more appropriate regions while the original puts attention on irrelevant parts.

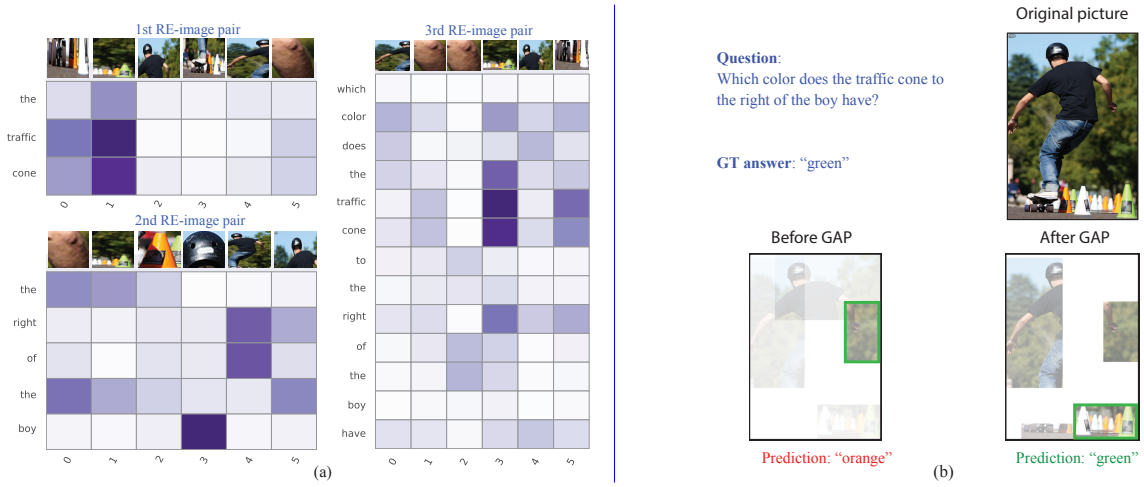

Figure 2. Qualitative analysis of GAP with MACNet baseline. (a) Region-word alignments of different RE-image pairs learned by our unsupervised grounding framework. (b) Visual attentions and prediction of MACNet model before (left) vs. after applying GAP (right). Visualized attention weights are obtained at the last reasoning step of MACNet.