Tips and Tricks for Visual Question Answering: Learnings from the 2017 Challenge

Supplementary material

A. Additional Ablative Experiments

A.1. Mini-Batch Size

The size of mini-batches during the optimization proves to have a strong influence on the final performance. Mid-range values in $\{128, \underline{256}, \underline{384}, \underline{512}, 768\}$ proved superior to smaller mini-batches (including even smaller values), although they require significantly more memory and high-end GPUs. We observed the optimum mini-batch size to be stable across variations of the network architecture, through other experiments not reported in Table 2.

A.2. Training Set Size

We investigate the relation between performance and the quantity of training data. We create random subsets of our training data and train four different models on it.

(1) Our best reference model.
(2) The ablation that uses word embeddings learned from scratch, instead of GloVe vectors.
(3) The ablation with the output classifier learned from scratch, instead of pretrained $\boldsymbol{w}_o^{\text{text}}$ and $\boldsymbol{w}_o^{\text{img}}$.
(4) The conjunction of *(2)* and *(3)*.

We plot in Fig. 3 their performance against the amount of training data and make the following observations.

– Unsurprisingly, the performance improves monotonically with the amount of training data. It roughly follows a logarithmic trend. Remarkably, we already obtain **reasonable performance with only 10% of the data**. Consequently, the gain when training on the whole dataset appears small relative to the ten-fold increase in data. That observation is common among natural language tasks in which the data typically follows a Zipf law [2] and in other domains with long-tail distributions. In those cases, few training examples are sufficient to learn the most common cases, but an exponentially larger dataset is required for covering more and more of the rare concepts.

– The use of **extra data to pretrain** word embeddings and classifiers is **always beneficial**. The gap with the baselines models learned from scratch shrinks as more VQA-specific training data is used. It could suggest that a sufficiently large VQA training set would remove the benefit altogether. An alternative view however, is that those other sources of information are most useful for representing rare words and concepts [32] which would require an impractically large dataset to be learned from VQA-specific examples alone. That view then suggests that extra data is necessary in practice.

– Pretrained word embeddings and pretrained classifiers each provide a benefit of the same order of magnitude. Importantly, the two techniques are clearly **complementary** and the best performance is obtained by combining them.

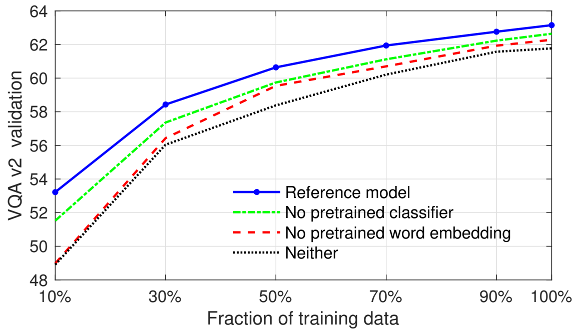

Figure 3. Performance of our reference model (Table 2, first row) trained on a subset of the training data. The use of additional non-VQA data for pretraining the word embeddings and the output classifiers is significant, especially when training on a reduced training set. Also not the tendency performance to plateau, and the small gain in performance relative to a 10-fold increase of training data. See discussion in Section A.2.

A.3. Ensembling

We use the common practice of ensembling several networks to obtain better performance. We use the most basic form of ensembling: multiple instances of the same model (same network architecture, same hyperparameters, same data) is trained with different initial random seeds. This affects the initialization of the learned parameters and the stochastic gradient descent optimization. At test time, the scores predicted for the candidates answers by all instances are summed, and the final answer is determined from the highest summed score.

As reported in Fig. 4, the performance increases monotonically with the size of the ensemble, *i.e.* the number of network instances. We obtained our final, best results, with an ensemble of 30 networks. The training of multiple instances is independent and obviously parallelizeable on multiples CPUs or GPUs. Interestingly, **even small ensembles** of 2–5 instances provide a significant increase in performance over a single network.

Note that those experiments include the validation split of *VQA v2* for training and use its *test-dev* split for evaluation, hence the higher overall performance compared to Tables 1 and 2.

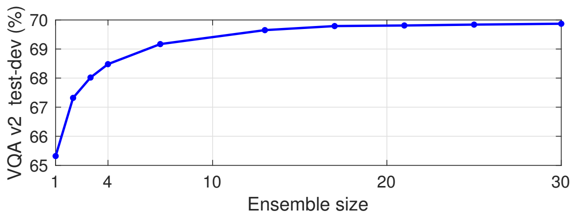

Figure 4. Performance of our best model (last row of Table 3) as a function of the ensemble size. The ensemble uses several instances of a same network trained with different random seeds. Their predicted scores are combined additively. Even small ensembles provide a significant increase in performance over a single network.

A.4. Output Classifiers

We take a closer look at the actual answers that improve with the proposed initialization (Section 3.5). We look, in Fig. 5, at the recall of each candidate answer j, defined as

$$\text{recall}_i \;=\; \frac{\sum_i^M (s_{ij} \overset{?}{=} 1.0 \;\;\wedge\;\; \hat{s}_{ij} \overset{?}{=} 1.0)}{\sum_i^M (s_{ij} \overset{?}{=} 1.0)} \qquad (11)$$

where M is the number of evaluated questions, s_{ij} is a ground truth score, and $\hat{s}_{ij}$ a predicted score. In Fig. 5, we plot the recall of a random selection of answers with and without pretraining the classifier. It is expected that pretraining $\boldsymbol{w}_o^{text}$ improves a variety of answers while $\boldsymbol{w}_o^{img}$ improves those with clearer visual representation. That intuition is hard to evaluate subjectively and is easy to confirm or disprove by cherry-picking examples. Note that, despite the overall benefit for the proposed approach, the **recall of many answers is affected negatively**. Other architectures may be necessary to obtain the full benefits of the approach. Another observation – not directly inferable from Fig. 5 – is that the recall the most influenced by pretraining – positively or negatively – is of answers with few training occurrences. This confirms the potential of the approach for **better handling rare answers** [32].

B. Cumulative Ablations

All ablative experiments presented in Section 4 consider one or two modifications of the reference model at a time. It is important to note that the cumulative effect of several modifications is not necessarily additive. In practice, this complicates the search for optimal architectures and hyperparameters. Some choices that appear promising at first may not pan out when combined with other optimizations. Conversely, some options discarded early on the search may prove effective once other hyperparameters have been tuned.

We report in Table 1 a series of cumulative ablations of our reference model. We consider a series of characteristics of our model in the inverse of the order in which they could be incorporated into other VQA models. The results follow the trends observed with the individual ablations. Removing each proposed contribution steadily decreases the performance of the model. This set of experiments reveals that the **most critical components** of our model are the **sigmoid outputs** instead of a softmax, the **soft scores** used as ground truth targets, the **image features from bottom-up attention** [3], the **gated tanh activations**, the **output layers** initialized using GloVE and Google Images, and the **smart shuffling** of training data.

C. Comparison with Existing Methods

We compare in Table 3 the performance of our best model with existing methods. Ours is an ensemble of 30 networks identical to the *reference* model (first row of Table 2) with the exception of the dimension of the hidden states, increased here to $1,500$. The issue of overfitting (Section 4.7) is mitigated by the large ensemble size. Compared to Table 2, this model also includes here the validation split of *VQA v2* for training. Our model obtained the first place at the 2017 VQA Challenge [1].

D. Implementation

Our model is implemented entirely in Matlab using custom deep learning libraries, with the exception of some Java code for multithreaded loading of input data. Training one network usually converges in 12–18 epochs, which takes in the order of 12–18 hours with $K = 36$ on a single Nvidia K40 GPU, or about twice as long on a CPU.

Acknowledgements

This research was partially supported by an Australian Government Research Training Program (RTP) Scholarship and by the Australian Research Council Centre of Excellence for Robotic Vision (project number CE140100016).

Method	VQA v2 test-dev All	Yes/no	Numb.	Other	VQA v2 test-std All	Yes/no	Numb.	Other
Prior (most common answer in training set) [12]	–	–	–	–	25.98	61.20	0.36	1.17
LSTM Language only (blind model) [12]	–	–	–	–	44.26	67.01	31.55	27.37
Deeper LSTM Q norm. I [23] as reported in [12]	–	–	–	–	54.22	73.46	35.18	41.83
MCB [11] as reported in [12]	–	–	–	–	62.27	78.82	38.28	53.36
MUTAN [7]	–	–	–	–	65.71	82.07	41.06	57.12
Athena	–	–	–	–	67.59	82.50	44.19	59.97
LV-NUS	–	–	–	–	66.77	81.89	46.29	58.30
HDU-USYD-UNCC	–	–	–	–	68.09	84.50	45.39	59.01
Proposed model								
ResNet features 7×7, single network	62.07	79.20	39.46	52.62	62.27	79.32	39.77	52.59
Image features from bottom-up attention, adaptive K, single network	65.32	81.82	44.21	56.05	65.67	82.20	43.90	56.26
ResNet features 7×7, ensemble	66.34	83.38	43.17	57.10	66.73	83.71	43.77	57.20
Image features from bottom-up attention, adaptive K, ensemble	**69.87**	**86.08**	**48.99**	**60.80**	**70.34**	**86.60**	**48.64**	**61.15**

Table 3. Comparison of our best model with competing methods. Excerpt from the official *VQA v2 Leaderboard* [1].

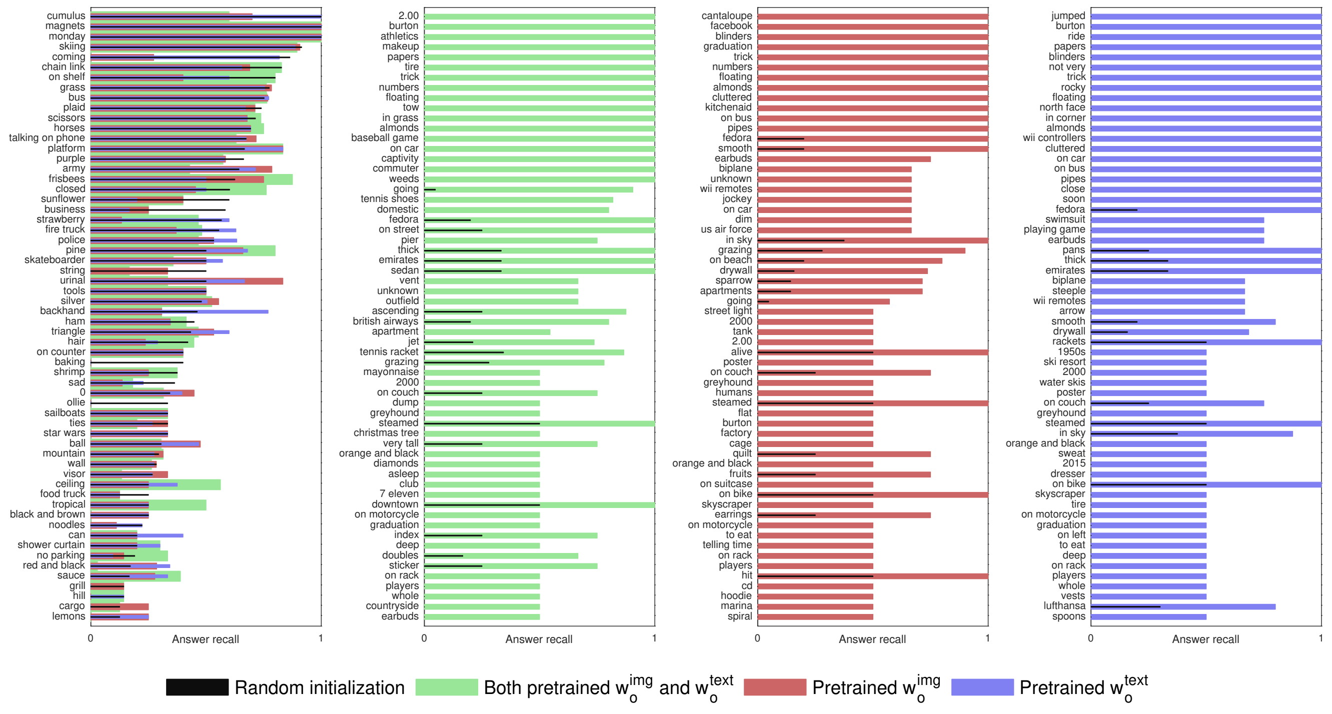

Figure 5. Effect of pretraining the output classifier on specific answers. We compare the per-candidate-answer recall of three models: a baseline using a classifier trained from scratch or pretrained (in **black**), and models using pretrained $\boldsymbol{w}_o^{text}$ and/or $\boldsymbol{w}_o^{img}$. *(Leftmost chart)* Random selection of answers sorted by their recall in the baseline model. *(Right three charts)* Top-60 answers with the largest improvement in recall by pretraining the classifier. See discussion in Section 4.6.