Visual Question Answering

Paritosh Gupta   Rohit Dubey   Harish Mashetty

Abstract

This paper presents the implementation for visual question answering (VQA) by improving upon the existing solutions. VQA is an important research area and researchers have proposed many solutions which are mainly dependent on different deep learning architectures. To gain knowledge and further research in this area, we have proposed a deep learning solution which is based on 2017 VQA winner solution [1]. After exploring various existing architectures and going through numerous research papers, we came up with a state of art solution which uses joint embedding of images and questions, bottom-up attention to extract salient image regions, and dual top-down stacked attention for guided attention in context of the question asked, nonlinearity is implemented using gated tanh and sigmoid activations for final classification.

1. Introduction

Visual Question Answering (VQA) involves an image and a related text question, to which the machine must determine the correct answer (see Fig. 1). Visual Question Answering (VQA) is an Artificial Intelligence problem that lies at the intersection of NLP and Computer Vision. The model is based on a deep neural network that implements joint embedding of the input question and the given image, followed by the multi-label classifier over a set of candidate answers.

In our implementation, model takes two types of inputs- images and corresponding questions/answers. For image input we used faster R-CNN processed input for bottom-up attention available at [2] which is further normalized before feeding into the model. For questions we use 300-dimension Glove word embeddings [3] which is further encoded with hidden state of Gated recurrent unit (GRU) and feed it to the model.

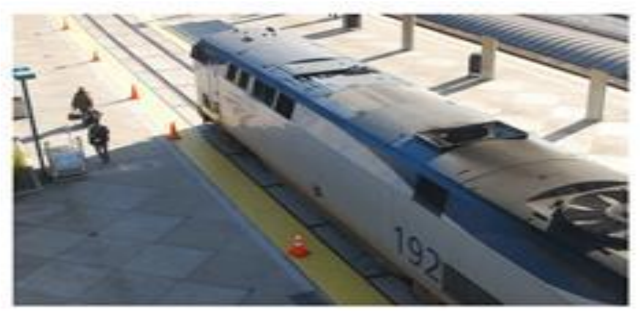
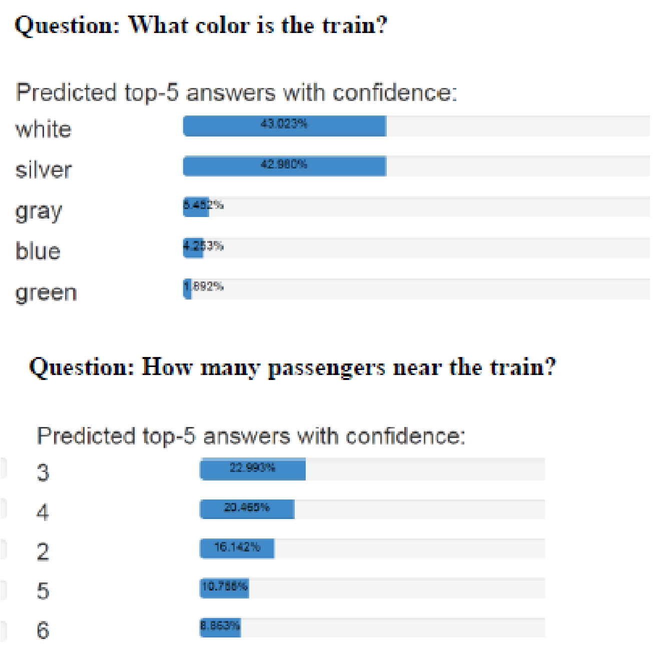


Figure 1. Example Visual questions and answers generated using CloudCV, http://vqa.cloudcv.org/



Question: What color is the train?

Predicted top-5 answers with confidence:

white	43.023%
silver	42.980%
gray	5.452%
blue	4.253%
green	1.892%

Question: How many passengers near the train?

Predicted top-5 answers with confidence:

3	22.990%
4	20.465%
2	16.142%
5	10.755%
6	8.863%

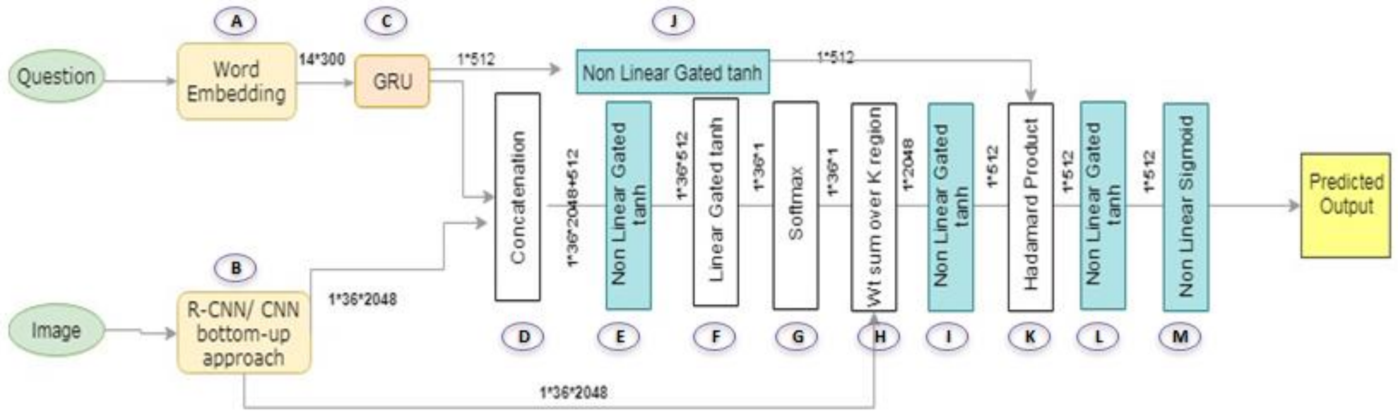


Figure2. Graphical representation of the model architecture. A deep learning network which uses a joint embedding of the input image and question, followed by a multi-label classifier over a fixed set of candidate answers.

2. Proposed Model

Our proposed solution takes VQA as a classification problem over a set of candidate answers. Questions are open-ended questions about images, with mostly one- or two-word answers. Our model is based on a deep neural network that implements a joint embedding of the image and of the question. The two inputs are mapped into fixed-size vector representations which derived from Faster RCNN and GRU, respectively. Further non-linear mappings of those representations are usually interpreted as projections into a joint "semantic" space. They are combined by concatenation of element-wise multiplication, before feeding to the classifier described above.

Let's understand the importance and working of each part of model architecture one by one

I. **Word Embedding:** The input for the model is text question and an image. First, a question is tokenized and trimmed to a maximum of 14 words. We find that not many questions' length exceeds 14 words (only 0.25% questions are greater than 14 words). After that, each word is converted into 300-Dimensional Glove embedding vector [300-dimension vectors are nothing but the Glove embedding], words missing in Glove embedding are initialized with a zero vector. We used Wikipedia/Gigaword pre-trained GloVe embedding which is available publicly.

   Questions shorter than 14 words are padded with zeros. The resulting sequence of word embedding is of size 14*300 and it is passed to GRU encoder (Fig 2- part C). The Recurrent gated hidden unit is of dimension 512 and we used its final state, after processing the 14-word embedding.

II. **Image Features:** The image features are a processed input coming from publicly available pre-trained data. The input feature method is based on ResNet CNN within a Faster R-CNN framework. It is trained to focus on specific elements in the given image, using annotations from the Visual Genome dataset. The resulting features can be interpreted as ResNet features centered on the top-K objects in the image. **This method provides image features using bottom-up attention.**

   We choose K=36 for fast processing considering the available resources. Each K region is thus represented by 2048-Dimensional vector that encodes the appearance of the image in that specific region.

III. **Image attention:** This model implements a standard question-guided attention mechanism used in modern VQA models. **This question-guided attention mechanism is termed as top-down attention**.

   For each input region i=1 to K in the image, the feature v_i is concatenated with question embedding q (Fig 2- part D). Then both passed through a non-linear layer f_a (Fig 2- part E) and a linear layer (Fig 2- part F) to obtain a scalar attention weight α associated with that location.

$$a_i = w_a f_a([v_i, q]) \quad (1)$$
$$\alpha = softmax(a) \quad (2)$$

$$\hat{v} = \sum_{i=1}^{k} \alpha_i v_i \quad (3)$$

Where w_a is a learned parameter vector. The attention weights are normalized over all locations with a SoftMax function (eq 2) (Fig 2- part G). Then image feature from all regions/locations are weighted by the normalized values and summed together (eq 3) (Fig 2- part H) to get a single 2048-sized vector $\hat{v}$ representing the attended image.

The weighted sum of all the image regions/locations is a basic attention mechanism known as simple one-glimpse, one-way attention.

IV. **Improvement on existing paper [1]: We also implemented dual top down stacked attention (eq 4), which queries an image two times to focus attention to the specific region relevant to the context of asked question.**

$$\hat{\boldsymbol{v}} = \boldsymbol{q} + \hat{\boldsymbol{v}} \qquad (4)$$

V. **Multimodal Fusion:** The image representation $\hat{v}$ we got in step 3 and question (q) are passed through non-linear layers (gated tanh) and then combined with simple Hadamard product (element-wise multiplication):

$$\boldsymbol{h} = \boldsymbol{f_q}(\boldsymbol{q}) \circ \boldsymbol{f_v}(\hat{\boldsymbol{v}}) \qquad (5)$$

The resulting vector h is called as the joint embedding of the image and question. This h is then fed to the output classifier.

VI. **Output Classifier:** We treat VQA as a multi- label classification task. A set of candidate answers (output vocabulary), is a predetermined set of all correct answers that appeared more than 8 times in training dataset. This result in N=3129 candidate answers and acted as classes in classification task. In VQA2 dataset, each question has been asked to multiple people, and their response recorded, thus in many cases there is no one correct answer, for example some people can identify color of an object as silver, while others might say white. Thus, instead of taking one correct answer we create soft score using formula in equation (6), thus any answer which was given by three or more persons is taken as correct. This approach of soft scoring takes care of any ambiguity/disagreement between human annotators.

$$\boldsymbol{Acc}(\boldsymbol{ans}) = \mathbf{min}\left\{\frac{\text{\# humans that said ans}}{\mathbf{3}}, \mathbf{1}\right\} (\mathbf{6})$$

The multi-label classifier passes the joint embedding h through a non-linear layer f and then through a linear mapping w to predict as score s' for each N candidates:

$$\hat{s} = \sigma\big(w_o f_o(h)\big) \qquad (7)$$

where sigma is a sigmoid (logistic) activation function and w is RN*512 is learned weight matrix.

The sigmoid normalize the final scores to (0,1), which are followed by a loss similar to binary cross entropy, although we used soft target scores. This final stage is acting as logistic regression that predicts the correctness of each candidate answer. The objective function is

$$L = -\sum_{i}^{M}\sum_{j}^{N} s_{ij}\log\big(\widehat{s_{ij}}\big) - \big(1 - s_{ij}\big)\log\big(1 - \widehat{s_{ij}}\big) \; (8)$$

where i and j are index for M training questions and N candidate answers. The ground-truth scores s are the soft accuracies of ground truth answers.

The advantage of this approach is that sigmoid outputs allow optimization for multiple correct answers per questions and use of soft scores as targets provides a somewhat better training signal than binary targets, as they capture the occasional uncertainty in ground truth observations.

The classification accuracy is calculated by comparing predicted and actual set of answers per question for given sample.

VII. **Classifier Training:** The output formula is

$$\hat{\boldsymbol{s}} = \sigma\big(\boldsymbol{w_o} f_o(\boldsymbol{h})\big) \qquad (7)$$

Score of a candidate answer j is determined by dot product of joint image-question representation $f_o(\boldsymbol{h})$ and the jth row of $\boldsymbol{w_o}$. The weight $\boldsymbol{w_o}$ is learned during the training for each candidate answers. We use the linguistic information in the form of GloVe word embedding of the answer word (similar to how we described question embedding).

We also used the visual information gathered from images representing the candidate answers. We used publicly available pre-trained fast R-CNN input which provide 2048-sized vector of features of each candidate answers. These vectors are placed in the corresponding row of matrix $\boldsymbol{w_o}$.

VIII. **Non-Linear Layers:** Our model used non-linear layers in multiple occasions. We tried ReLU and tanh and selected tanh after considering the trade-off between time complexity vs performance. In our model, each non-linear layer uses gated tanh activation. The tanh function is defined as-

$$\boldsymbol{\hat{y} = \tanh(Wx + b)} \quad (9)$$
$$\boldsymbol{g = \sigma(W'x + b')} \quad (10)$$
$$\boldsymbol{y = \hat{y} \circ g} \quad (11)$$

This formulation is inspired from similar gating operations within recurrent units such as GRUs

IX. **Training:** We use stochastic gradient descent to train the network. Different model hyper parameters are selected based on multiple iterations and optimizing the model.

3. Preliminary results:

The model is run with following settings:

- Batch Size- 512 (training and validation)
- Hidden dimension- 512
- Epoch- 160

The size of training dataset is 138,178 and of evaluation dataset is 159.

The train classification accuracy we got was 62.24%

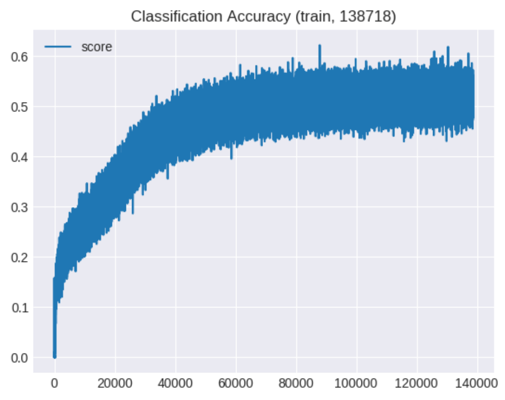


Fig 3: Train Classification Accuracy

The eval classification accuracy we got was 57.35%

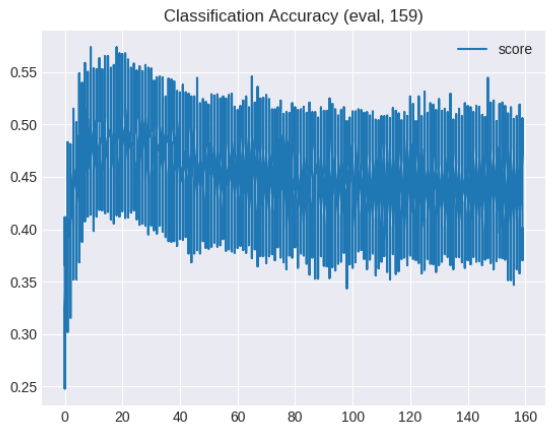


Fig 4: Evaluation Classification Accuracy

The train cross entropy loss we got was 1084

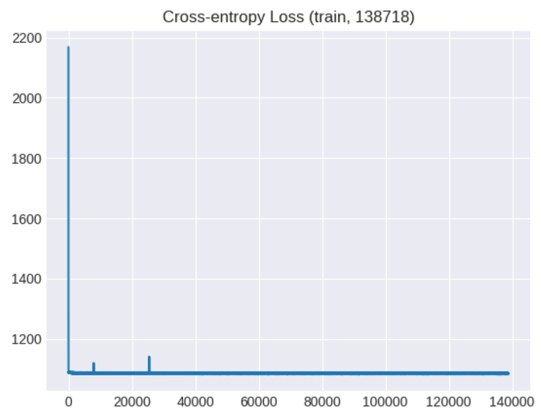


Fig 5: Train Cross-entropy Loss

The eval Cross Entropy Loss we got was 12

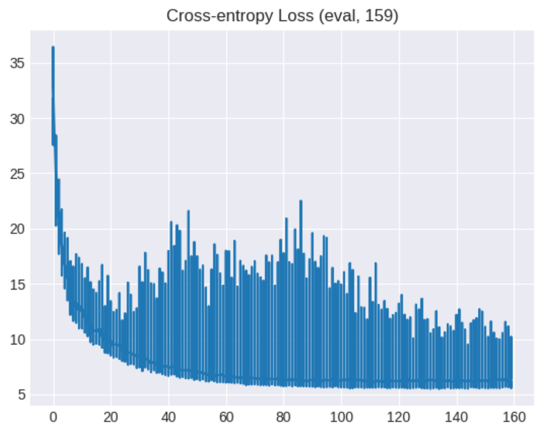


Fig 6: Evaluation Cross-entropy Loss

4. References:

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

[2]https://imagecaption.blob.core.windows.net/imagecaption/trainval_36.zip

[3] http://nlp.stanford.edu/data/glove.6B.zip