University of California, Santa Barbara

Project Report

Visual Question Answering with Multi-Modal Hierarchical Co-Attention

Author:
Sai Nikhil Maram
Michael Zhang

Supervisor:
Prof. Xifeng Yan

June 13, 2018

Contents

1 Objective

The objective of our project is to have a deep learning model that answers open-end questions based on the given image. A typical application could be a system to aid vision-impaired people understanding contents on a picture. Furthermore, the system could digest real-time data and provide surrounding information.

2 Introduction

The advent of convolutional neural network has empowered computer to gain deep understanding of images, as well as natural language is processed by RNN with LSTM cells. Thus, we have witnessed a tendency to combine such two models as encoder to achieve the visual question and answering. There are many researchers that have explored this domain with various datasets and state-of-the-art models. These previous work of multi-modality serve as stepping stone towards the higher level of AI-complete system in the future.

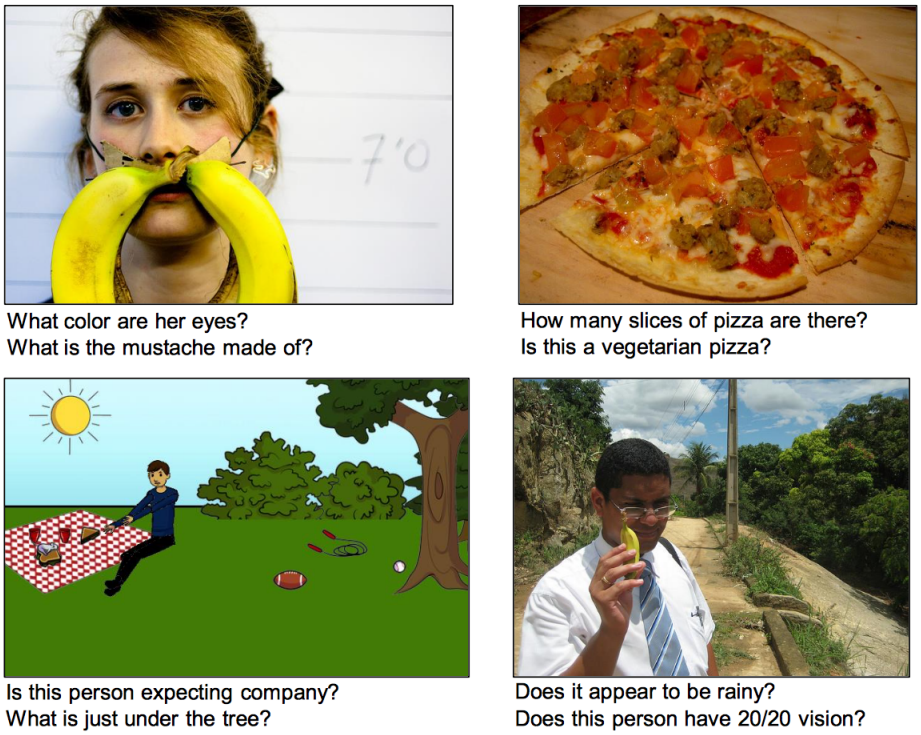

Figure 1: Examples of open-end visual question and answering

In this project, we experimented the parallel co-attention mechanism on both image and question modeling. This enables our model to focus on specific area in the image based on question. The result proves that the co-attention boosts the accuracy of answering both on training and testing datasets. This project is inspired by the paper[2] and [3] and implemented from scratch.

3 Model

We have implemented two Models for the task of Visual Question Answering. We refer them as Base Model and Hierarchical Co-attention Model. A typical system of VQA consists of image, question(represented by text) as inputs and answer to the question as output. Systems differ in how

the image and questions features are encoded into a common vector space, followed by decoding the vector space to get the answer. Typically, the image features are computed by Convolution Neural Network(CNN) whereas the text features are computed using Recurrent Neural Network(RNN) to preserve the temporal information in the text. Base Model considers the aggregate features from question and image to determine the answer. While the Hierarchical Co-attention model determines the answer by attended image and question features. Both of these models will be discussed in further sections. We used Base model as a baseline for our accuracy and results.You can find out implementation of Base Model at https://github.com/Heronalps/Visual_QA and Hierarchical Co-Attention Model at https://github.com/Heronalps/Visual_QA_Attn and

4 Base Model

The system consists of an Encoder which embeds the image and question into a common vector space and a decoder that decodes the vector space to obtain the answer.

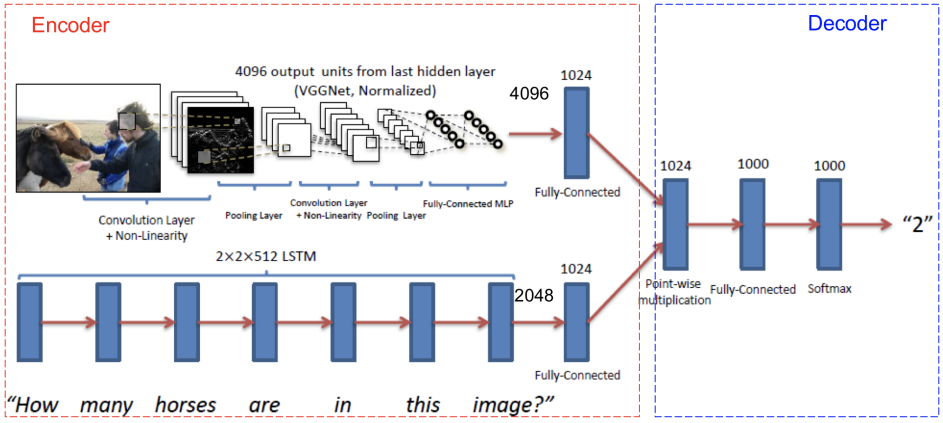

Figure 2: Base Model Architecture

4.1 Encoder

The encoder part consists of image and question encoding.

4.1.1 Image Encoding

A pre-trained vgg16 CNN model on Imagenet is used as an encoder. The Vgg16 model consists of 5 convolution layers, 2 fully connected layers & 1 softmax layer as in Fig 3. Outputs of fully connected layer are considered as image features which is of size 4096.

4.1.2 Question Encoding

As previously discussed, RNNs are used to encode the question into vector space by preserving temporal information. We have used LSTM as a RNN module to mitigate the problems of vanishing gradient descent. We have a fixed length of LSTM units as we will have a threshold on maximum number of words each question can have. A lot of pre-processing needs to be done on text as any computation model only deals with vectors. More about pre-processing is discussed in the Pre-processing section. We have used a 2 layer LSTM to encode the question. The state of the final LSTM unit is considered as question feature. A LSTM of 512 unit size is considered in each layer. Each LSTM unit gives hidden state of size 512 and cell state of size 512. Both the states are

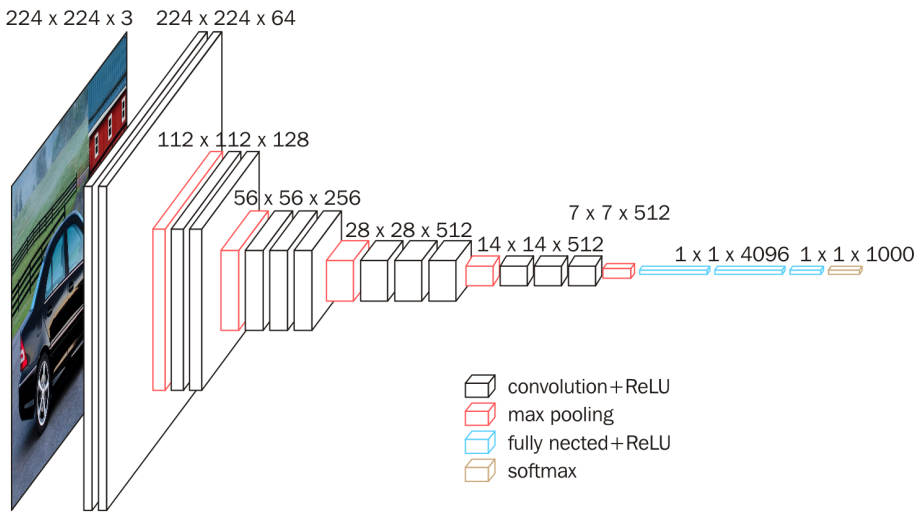

Figure 3: VGG16 model

concatenated to get a 1024 vector. Since two LSTM layers are considered we get a 2048 size vector as question feature.

Since the outputs from both image and question encoding are different, we have a fully connected layer at image and question encoding to get them to a size of 1024. This can be seen in the Encoder part present in Fig 2. Thus the outputs of the encoder are two vectors of size 1024 which represents the image and question features.

4.2 Decoder

The Decoder performs a softmax-classification for the image and question features calculated by Encoder. Decoder predicts the best answer among the top 1000 chosen from dataset. The top 1000 answers accommodates around 85% answers of the dataset. Hence this is mostly a classification task rather than generating task for answers. The steps involved in classification are, First, a pointwise image and question features are multiplied to get a single vector of sizer 1024. This is fed to a fully connected layer of size 1000 and softmax layer as seen in decoder in Fig 2. The highest output from the softmax layer is the answer to the give question.

5 Hierarchical Co-Attention Model

In the Base Model, we have seen the encoder takes the output of the final fully connected layer of CNN and final LSTM unit state as the outputs. While these features represent the whole image and question,no specific priority is given to certain words of the question or certain portions of the image. In Hierarchical Co-Attention Model, we consider multiple features w.r.t image and question and give priority to certain features. The priority given to certain features is called attention. In our model, we consider attention to image features based on question features and attention to question features based on image features. This is so called co-attention part in our model which will be discussed further in the next section. Before explaining the model, we would first present you about attention mechanism, features considered in Hierarchy Model and co-attention mechanism.

5.1 Attention

Attention corresponds to importance given to a particular feature of the input among its multiple features. Here we have question and image Attention as they constitute inputs of our system.

5.1.1 Question Attention

- Importance(weights) given to each word in the question.
- Question features is obtained by weighted sum of word embeddings with attention weights

$$\begin{gathered}\text{Word Embedding} : \{w_1, w_2, \cdots w_T\} \\ \text{Attention Weights} : \{a_1, a_2, \cdots a_T\} \\ \text{Attended Question Features} : \sum_{i=1}^{T} a_i * w_i\end{gathered}$$

5.1.2 Image Attention

- Importance(weights) given to each feature map(outputs of a Convolution layer in CNN).
- Image features is obtained by weighted sum of feature maps with attention weights.

$$\begin{gathered}\text{Image features} : \{f_1, f_2, \cdots f_N\} \\ \text{Attention Weights} : \{b_1, b_2, \cdots b_N\} \\ \text{Attended Image Features} : \sum_{i=1}^{N} b_i * f_i\end{gathered}$$

Here we can see instead of taking a single feature, we take multiple features and do a weighted average of all the features present depending on its importance to get the final feature. Now that we know what attention is given features and weights. We need to determine the multiple features considered for image and question and how to calculate the attention weights. We will be talking about these in further sections.

5.2 Features

In this section, we talk about the features that are considered for image and question that are used in calculating the attended image and question features respectively. We present you the Hierarchy present in the Question and specify the features in each level.

5.2.1 Hierarchical Question Features

In this section we present you 3 levels of hierarchy present in the question and the features at each level. In the later sections, we will demonstrate how these features are used in determining the co-attention.

Word Level

In the word level, the words are embedded into vector space which is also called word embedding in NLP domain. The embeddings of the words are randomly initialized which are learned during training. Each word is embedded to a dimension of 512.

$$Q^w = \{q_1^w, q_2^w, \cdots q_T^w\}$$

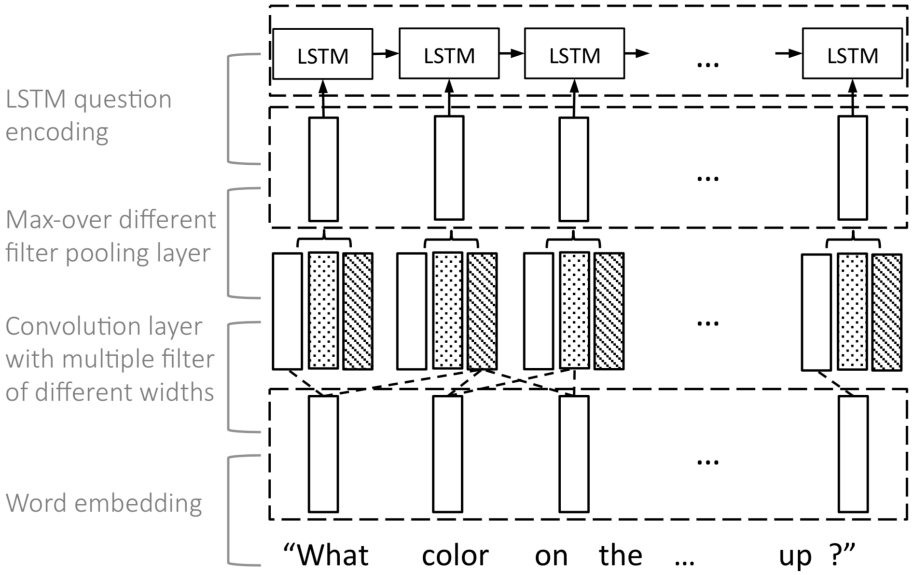

Figure 4: Question Hierarchy

Phrase level

At phrase level, initially unigram, bigram and trigram features are considered for each word. The bigram and trigram features provide more information about the context, the word is present in. For example, What color is the table ? and What food is on the table ?. Both the questions first word is same but if we know more about the context then the system can know better about the question. The bigram/trigram features provide information about the context. The gram features are computed by 1D convolution with different filter sizes at each word.

$$\hat{q_{s,t}}^p = tanh(W_c^s q_{t:t+s-1}^w), \quad s\epsilon\{1,2,3\}$$

where s determines the filter size. The phrase level feature of the word is the maximum of these gram features

$$q_t^p = max(\hat{q_1}^p, \hat{q_2}^p, \hat{q_3}^p)$$
$$Q^p = \{q_1^p, q_2^p, \cdots q_T^p\}$$

Sentence Level

The phrase level feature at each word is fed as an input to the LSTM at the sentence level. The sentence level features at each word is the hidden state of LSTM at the current time stamp(at current word). A LSTM of 512 unit size is considered for this purpose.

$$q_t^s = LSTM(q_t^p, q_{t-1}^s)$$
$$Q^s = \{q_1^s, q_2^s, \cdots q_T^s\}$$

$\{Q^w, Q^p, Q^s\}$ represents the features of question at word, phrase and sentence level respectively. Each Q is a vector of size $T = 25$, which represents the maximum question length. Each element of Q, q has a dimension of $d = 512$. Hence each Q is of size 25*512. In the report, we refer question feature as features in any one of the three levels.

5.2.2 Image Features

In the Base Model, we have taken the output of fully connected layer as the image features. Output of the fully connected layer specifies the prominent object present in the image, as the

output of the fully connected layer is used for image classification. Since for attention, we need multiple features corresponding to image, considering output of convoluted layer in CNN would be right thing. Each feature map of the convolution layer can be considered as a feature. In VGG16 (3) , we consider output of final convolution layer for this purpose. The output of final convolution layer is 14*14*512. We do a reduce mean along the row(axis = 1) to get image features of dimension 14*512. This represents that each image has 14($N = 14$) features with dimension of each feature being 512.

$$\text{Image features} = \{f_1, f_2, \cdots, f_N\} \ , \quad N = 14$$

Note: Image features are same at each level.

5.3 Co-Attention

In the previous section, we have seen the features that are considered for image and question at different levels and how attended features are calculated given attention weights. In this section, we will tell you how the attention weights are generated using co-attention mechanism. Image and Question features considered in calculations are transpose of original features. In co-attention mechanism, both image($V \epsilon R^{d*N}$) and question features($Q \epsilon R^{d*T}$) are used to calculate the attention weights. We consider which question feature attends to a particular image feature and vice versa. Following steps are involved in calculating the attention weights.

1. Affinity Matrix : Calculates the similarity between all pair image and question features.

$$C = tanh(Q^T W_b V), \qquad C \epsilon R^{T*N}$$

2. Use the Affinity Matrix as a feature to predict the image and question attention maps. First, create a hidden state using Multi-Layer Perceptron(MLP) with inputs as C, V, Q

$$H^v = tanh(W_v V + (W_q Q)C) \qquad H^q = tanh(W_q Q + (W_v V)C^T)$$

3. softmax squashing is done on Hidden state to get the attention weights.

$$a^v = softmax(w_{hv}^T H^v) \qquad a^q = softmax(w_{hq}^T H^q)$$

4. Attended features are calculated using attention weights and features.

$$\hat{v} = \sum_{n=1}^{N} a_n^v v_n \qquad \hat{q} = \sum_{t=1}^{T} a_t^v q_t$$

   The attended features are calculated at each level. For each level the input image features are same, only question features change. The above Co-Attention mechanism is applied at each level to calculate word ($\{\hat{v}^w, \hat{q}^w\}$), phrase($\{\hat{v}^p, \hat{q}^p\}$) and sentence level($\{\hat{v}^s, \hat{q}^s\}$) attended features.

5.4 Hierarchical Co-Attention Model

In the previous sections, we have seen how to calculate the attended image and question features given input and question features. In this section we will show how these attended features are used to predict the answer. Similar to Base Model, Hierarchy model also consists of Encoder and Decoder.

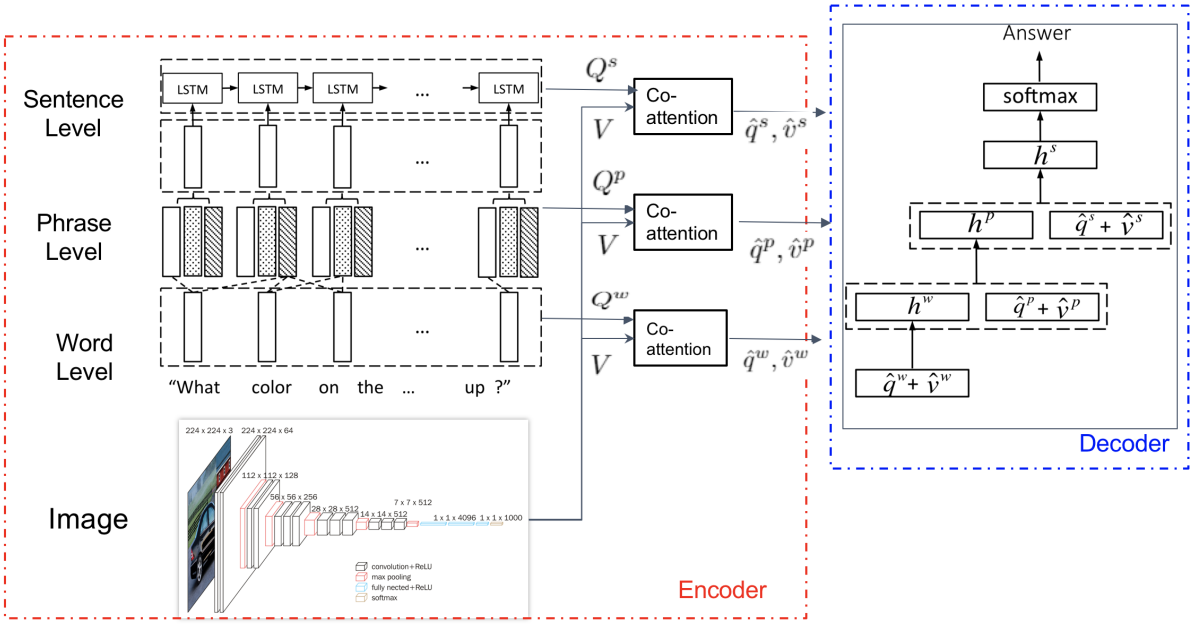

Figure 5: Hierarchical Model Architecture

5.4.1 Encoder

Similar to the Base Model, the encoder in Hierarchy model encodes the question and image into a vector space. The output of the encoder in the Base Model is the aggregated features of image and question. Here the output of the encoder is attended features in both image and question. In Features section 5.2, we have seen how we calculate the image and question features at different level. In the Co-Attention section 5.3, we have seen how we calculate the attended features($\{\hat{v}, \hat{q}\}$) given the image(V) and Question features(Q). The encoder part of the model consists of co-attention mechanism at each level in finally producing $\{\hat{v}^w, \hat{q}^w\}, \{\hat{v}^p, \hat{q}^p\}, \{\hat{v}^s, \hat{q}^s\}$ as seen in encoder in Fig 5.

5.5 Decoder

As in Base Model, we consider the top 1000 answers among the dataset and is a classification task to the decoder. The input to the decoder are the attended features at different levels to predict the answer. A Multi-Layer Perceptron (MLP) is used in decoder to predict the answer from these features. First word level features $\{\hat{v}^w, \hat{q}^w\}$ are fed to MLP to obtain a hidden state $\hat{h}^w$. The hidden state $\hat{h}^w$ is then concatenated with attended features in the phrase level $\{\hat{v}^p, \hat{q}^p\}$. This concatenated vector is used as an input to the next MLP to obtain hidden state $\hat{h}^p$. Again $\hat{h}^p$ is concatenated with sentence level attended features $\{\hat{v}^s, \hat{q}^s\}$ and fed to MLP to obtain hidden state $\hat{h}^s$. The hidden state at sentence level $\hat{h}^s$ is then fed to a softmax layer and maximum ouput is considered as the answer to the question

6 Datasets

Based on our literature review, most of previous work before VQA used relative small and monotonic image dataset, such like SUN that solely contains scenes, places and objects within. In the project, we used Microsoft Common Objects in Context (COCO) datasets. The training and testing dataset contains 82,783 and 40,504 images respectively. In the question side, the VQA project provides 443,757 questions each with 10 answers for training images. Averagely, each image

has 5.4 corresponding questions. It's worth pointing out that these answer are generated by workers on Amazon Mechanical Turk and have different confidence levels as "yes", "no" and "maybe". Apparently, the answers with "yes" confidence level are chosen in the training and testing.

We can roughly categorize questions in the dataset into five classes. They are fine-grained recognition ("What kind of cheese is on the pizza?"), object detection("How many bikes are there?"), activity recognition("Is this man laughing?"), knowledge base reasoning("Is this a vegetarian pizza?"), and common sense reasoning("Does this person have 20/20 vision?"). The top five question types are "What is"(13.84%), "What color"(8.98%), "What kind"(2.49%), "What are"(2.32%) and "What type"(1.78%). In terms of answer, the top 5 ones are yes(22.82%), no(15.35%), 2(3.22%), 1(1.87%), white(1.68%).

7 Preprocessing

Preprocessing is a major step in our project. The input data consists of a question and answer json files respectively. The first step would be to create a <question,answer,image> pair from the json files. Following are the steps done for each question,answer and image.

7.1 Image

1. The CNN is the encoder is initialized as a pre-trained VGG16 model.
2. Every image is rescaled to [244 * 244 * 3] dimension before being fed into CNN.

7.2 Question

1. In order to construct vocabulary, we used nltk library to tokenize all words present in training questions.
2. Thus, only questions whose words are in vocabulary are considered in the validation dataset.
3. To feed them into LSTM, all questions are left padded to a maximum length of 25 in the encoder.

7.3 Answers

1. Based on statistics, top 1000 answers are considered from dataset which accommodates to about 85% of answers.
2. Each answer has different confidence level. Picking the answer with confidence level 'yes'.
3. Indexing the answers from 1 to 1000 for softmax classification.

8 Training

We realized the settings of training directly affect the final result of entire model, thus we tried various setup and hyper-parameters to accommodate our model to full extent.

With respect to word embeddings, we tried GloVe and word2vec embeddings at the beginning, but found out most words need to be distinguished are highly correlated to each other. The most noteworthy example is "yes" and "no". Moreover, pre-trained embeddings decelerates the training

process due to enormous lookups for vectors. These are the reasons why we choose to randomly initialize words with embedding dimension of 512.

In the co-attention model of LSTM, an one-dimension convolution layer is used to calculate the gram features in the phrase level. This layer serves as max-pooling layer to extract gram feature among uni-gram, bi-gram and tri-gram phrases based on norm. Since tensorflow doesn't have built-in library to calculate such maximum among different batch tensors, we wrote our own function to perform such operation. In the next level of sentence, a single layer of LSTM with hidden unit size of 512 is used. Every hidden state of such LSTM is collected along the way of training to calculate weighted attention question features.

Additionally, we picked "tanh" for activation function and cross-entropy for loss calculation. The adaptive learning rate optimizer, Adam, is used with an initial learning rate of 0.0001. The batch size of 128 is used to run for 20 epochs.

One primary difficulty we encountered is that each epoch initially takes around six hours to run on a 8 GPU virtual machine on Google Cloud. We investigated the reason and found the bottleneck is extracting CNN feature maps for the same image repeatedly, because each image has multiple corresponding questions. Thus, we extracted and saved the feature maps of 82,783 training images to a static dictionary file, which greatly accelerates the training process to around 2 hours.

9 Results

As result, the base model achieves the accuracy of 50% on the training dataset and 37% on the validation dataset. On the other hand, the hierarchical co-attention model reaches the accuracy of 60% on the training dataset and 45% on the validation dataset. Obviously, the co-attention model outperforms the base model on two datasets respectively. We further investigated the image and question pairs of right, diverse and wrong prediction from base and co-attention models.

9.1 Both Models predicting correct

Figure 6: Image, question and answers of Right Prediction

For those image and question pairs of right prediction, the image tends to have fewer objects that are separated clearly against each other. The question is also straightforward that any individual would be able to answer intuitively as seen in Fig 6. For example, the 2nd image in top row inf

Fig 6 asking for which animal is this ? Both models predict accurately as Bear is prominent in the image.

9.2 Attention Model Predicts correct

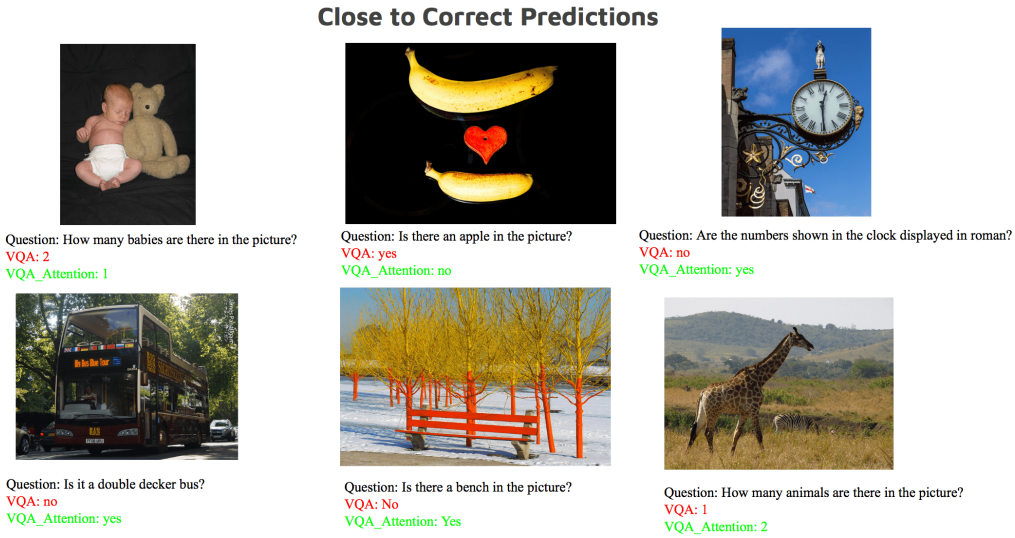

Figure 7: Image, question and answers of Close Prediction

With respect to the image and question pair that base and co-attention models give diverse answers, we figured out this is either because of the similarity of different objects on the image or the mixture of background and object of interest. In such scenario, the co-attention mechanism is able to focus on the specific area of the picture guided by question. Therefore, the model provides right answer other than base model as seen in Fig 7. For example, the top left image in Fig 7, the base model thinks the doll as also a baby while the attention model just concentrates on the baby.

9.3 Both Models predicting wrong

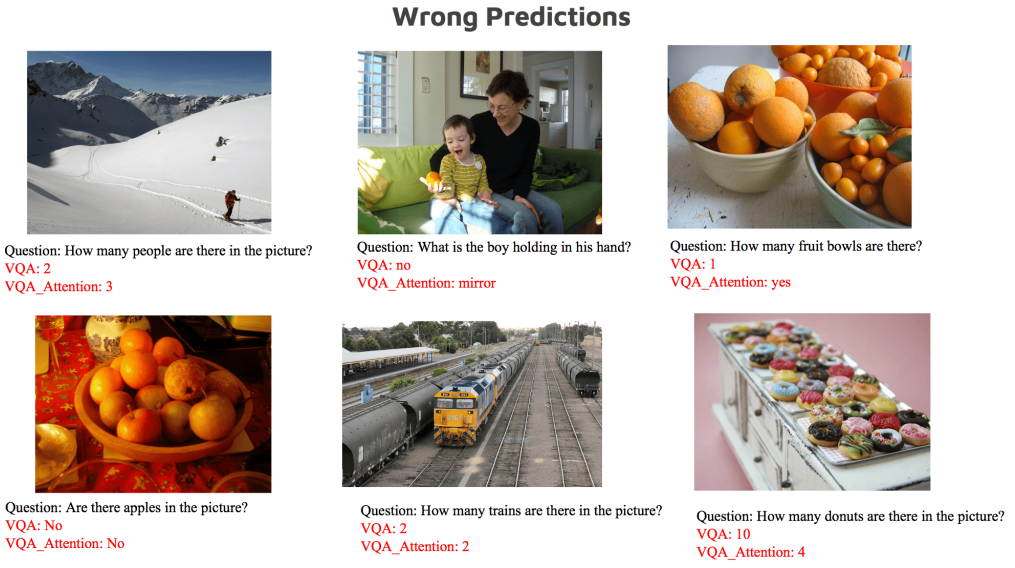

Figure 8: Image, question and answers of Wrong Prediction

For those images having numerous and vague objects, both models are unable to identify the right objects to extract and answer the question upon. Further, the polychrome nature of these images also play a role in making wrong predictions as seen in Fig 8. In our future work, we expect to improve our encoder with more sophisticated neural network to gain fine-grained features that leverage our entire model accuracy.

References

[1] Xu H., Saenko K. (2016) Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9911. Springer, Cham.

[2] Lu, J., Yang, J., Batra, D., & Parikh, D. (2016). Hierarchical question-image co-attention for visual question answering. In Advances In Neural Information Processing Systems (pp. 289-297).

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In ICCV, 2015.

[4] Zhang, P., Goyal, Y., Summers-Stay, D., Batra, D., & Parikh, D. (2016, June). Yin and yang: Balancing and answering binary visual questions. In Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on (pp. 5014-5022). IEEE

[5] Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., & Parikh, D. (2017, July). Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In CVPR (Vol. 1, No. 6, p. 9).