Visual Question Answering using Deep Learning

Aahitagni Mukherjee
ahitagni@iitk.ac.in

Shubham Agrawal
shubhag@iitk.ac.in

Abstract

In this project, we have employed a combination of recurrent neural network(RNN) and convolutional neural network (CNN) for a multimodal quesion answering task related to images. Our proposed end-to-end trainable neural network provides a framework for learning the image representation, the sentence representation for question, the inter-modal interaction between the image and question, and also takes into account the sequential nature of the question input for the generation of answer. The proposed model has four components: a Long-Short Term Memory(LSTM) component and a CNN to encode the question,an image CNN to extract the image representation, and one multimodal convolution layer to combine the two modalities of the image and question,which is further used to obtain the joint representation for generating the answer words. We employ our proposed model on DAQUAR, a dataset created for the image question answering (QA).

1. Introduction

Recently, multimodal learning between image and language using deep neural networks have received increasing interest as research topics. In order to further explore Researchers have proposed a new "AI-complete" task, which is named as visual question answering (VQA) [Antol et al.] or image question answering (QA) [Malinowski et al.]. The image QA task takes an image and a free-form, natural-language like question about the image content as the input and produces the answer as the output.

Joint understanding image and question together for image QA incorporates a number of challenging tasks from the fields of machine learning, natural language processing and computer vision, which have been regarded as the holy grail of automatic image understanding and AI in general [Antol et al.]. Therefore, the study on the image QA task can be useful to further explore the abilities of the AI research.

In this paper, we employ convolutional neural network (CNN) and Long-Short Term Memory(LSTM) to address the image QA problem. By training on a set of triplets consisting of (image, question, answer), our proposed CNN model learns to answer the free-form, natural-language like questions of the image content. This model is a modified form of the model proposed by [Ma et al.]. Our main contribution is the addition of a LSTM component in this model to capture the sequential nature of the sentence input for achieving better accuracy.

2. Related Work

There is a thread of recent work on Visual Question Answering based on neural networks:

- In neural-based approach[Malinowski et al.(2015)], image representation from a CNN is fed to each hidden layer of a single LSTM. The LSTM then models the concatenation of question and answer.

- The mQA[Gao et al.] approach contains four components: an LSTM to extract the question representation, a CNN to extract the visual representation, an LSTM for storing the linguistic context in an answer, and a fusing component to combine the information from the first three components and generate the answer.

- The VSE model(VIS+LSTM)[Kiros et al.] uses RNN's and Visual Semantic Embeddings. Here the image is treated as a single word, and the intermediate representation of the input thus obtained is used for classification into the correct class, which is the single word answer.

- The CNN approach[Ma et al.] uses 3 CNN's - one to extract sentence representation, one for image representation, and the third is a multimodal layer to fuse the two.

3. Approach

Our proposed CNN for image QA consists of 4 individual components: one image CNN encoding the image content, one LSTM which takes the sentence input sequentially, one sentence CNN composing the output of the LSTM into high semantic representation, one multimodal convolution layer fusing the image and question representation together

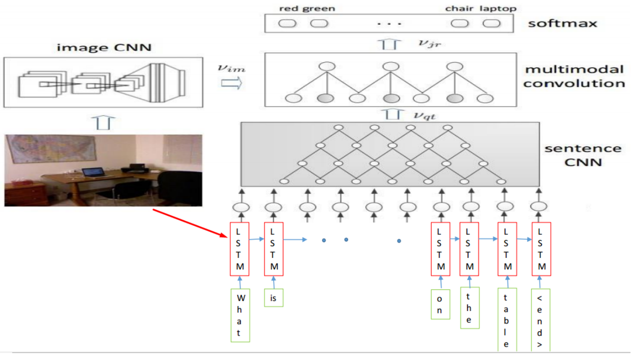


Figure 1. Our approach, described in Section 3.

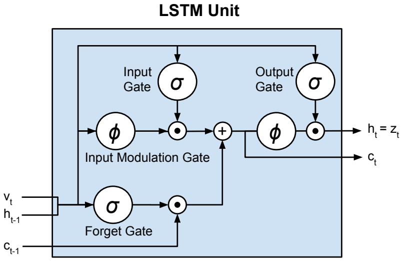


Figure 2. LSTM unit.[Malinowski et al.(2015)] described in Section 3.

and learning their joint representations. Finally, the joint representation is fed into a softmax layer to generate the answer to the multimodal input.

For image QA, the objective is to predict the answer a given the question q as well as the related image I:

$$a = \arg\max_{a \in \Omega} \; p(a|q, I; \theta), \tag{1}$$

where Ω is the set containing all answers. θ denotes all the parameters for performing image QA, which are learned during the training process.

3.1. LSTM for sentence

This single layer LSTM design is a modified form of the one proposed by [Malinowski et al.(2015)]. The difference is that, instead of getting final answer as the output of last LSTM cell, the generated sequential output of all the cells is passed on to the sentence CNN. Also, as a modification, the output of image CNN is fed both to the LSTM and multimodal convolution layer for a better modelling of the interrelationship between sentence and image.

The LSTM predicts a set of vectors for the sentence representation word-by-word $\{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{\mathcal{N}(q,I)}\}$, where $\boldsymbol{a}_t$ are output vectors of the LSTM, one for each word, and $\mathcal{N}(q, I)$ is the number of words in the concatenation of the given question and image. The problem is formulated as predicting a sequence of vectors, each of which is a representation of the part of the sentence from the start upto the corresponding word. Thus the prediction procedure can be formulated recursively:

$$\hat{\boldsymbol{a}}_t = \arg\max p(\boldsymbol{a}|\boldsymbol{I}, \boldsymbol{q}, \hat{A}_{t-1}; \boldsymbol{\theta}) \tag{2}$$

where $\hat{A}_{t-1} = \{\hat{\boldsymbol{a}}_1, \dots, \hat{\boldsymbol{a}}_{\boldsymbol{t-1}}\}$ is the set vectors for previous words, with $\hat{A}_0 = \{\}$ at the beginning, when no sentence representation has been generated yet. The process is terminated when $\hat{a}_t = \$$.

As shown in Figure 1, the LSTM is fed with a question sentence as a sequence of words, i.e. $\boldsymbol{q} = [\boldsymbol{q}_1, \dots, \boldsymbol{q}_{n-1}, [\![?]\!]]$, where each $\boldsymbol{q}_t$ is the t-th word in question and $[\![?]\!] := \boldsymbol{q}_n$ is the question mark - the end of the question. The maximisation is done at the final softmax layer after the multimodal layer. Both question and answer words are represented with one-hot vector encoding (a binary vector with exactly one non-zero entry at the position indicating the index of the word in the vocabulary) and embedded in a lower dimensional space, using a jointly learnt latent linear embedding, similar to as in [Malinowski et al.(2015)].

The design of LSTM unit is similar to the one used in [Malinowski et al.(2015)]. It has been described below for the sake of completeness. During training, we the question words sequence $\boldsymbol{q}$ is joined with the corresponding ground truth answer words $\boldsymbol{a}$, i.e. $\hat{\boldsymbol{q}} := [\boldsymbol{q}, \boldsymbol{a}]$. During testing and prediction, at time step t, $\boldsymbol{q}$ is joined with previously predicted answer words $\hat{\boldsymbol{a}}_{1..t} := [\hat{\boldsymbol{a}}_1, \dots, \hat{\boldsymbol{a}}_{t-1}]$, i.e. $\hat{\boldsymbol{q}}_t := [\boldsymbol{q}, \hat{\boldsymbol{a}}_{1..t}]$. The output from the image CNN ν_{im} is provided at every time step as input to the LSTM. We set the input $\boldsymbol{v}_t$ as a concatenation of $[\nu_{im}, \hat{\boldsymbol{q}}_t]$.

As shown in Figure 2, at each time step t, the LSTM unit takes an input vector $\boldsymbol{v}_t$ and predicts an output word $\boldsymbol{z_t}$. $\boldsymbol{z_t}$ is a linear embedding of the corresponding answer word $\boldsymbol{a_t}$. We use the LSTM unit as described in [Zaremba et al.] and the implementation in [caffe] by [Donahue et al.]. With the *sigmoid* nonlinearity $\sigma : \mathbb{R} \mapsto [0, 1]$, $\sigma(v) = (1 + e^{-v})^{-1}$ and the *hyperbolic tangent* nonlinearity $\phi : \mathbb{R} \mapsto [-1, 1]$, $\phi(v) = \frac{e^v - e^{-v}}{e^v + e^{-v}} = 2\sigma(2v) - 1$, the LSTM updates for time step t given inputs $\boldsymbol{v}_t$, $\boldsymbol{h}_{t-1}$, and the memory cell $\boldsymbol{c}_{t-1}$ as follows:

$$\boldsymbol{i}_t = \sigma(W_{vi}\boldsymbol{v}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{3}$$

$$\boldsymbol{f}_t = \sigma(W_{vf}\boldsymbol{v}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{4}$$

$$\boldsymbol{o}_t = \sigma(W_{vo}\boldsymbol{v}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{5}$$

$$\boldsymbol{g}_t = \phi(W_{vg}\boldsymbol{v}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \tag{6}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \tag{7}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t) \tag{8}$$

where $\odot$ denotes element-wise multiplication. The weights W and biases b of the network are learnt using the cross-entropy loss. Conceptually, as shown in Figure 2, Equation 3 corresponds to the input gate, Equation 6 the input

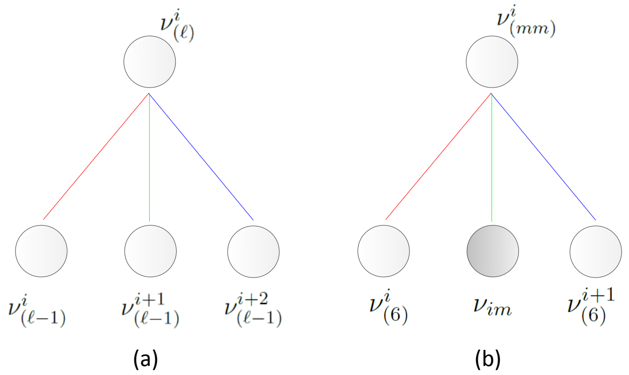


Figure 3. The convolution unit of sentence CNN (a) and multimodal convolution layer (b)[Ma et al.].

modulation gate, and Equation 4 the forget gate, which determines how much to keep from the previous memory c_{t-1} state. Intermediate sentence representation $\hat{A}_t$ is fed to the sentence CNN.

3.2. Sentence CNN

The sentence CNN described here is a modified version of the one described in [Ma et al.]. The difference is: the input to the sentence CNN is not the word embeddings, but the intermediate sentence representations from the LSTM.

For a sequential input ν, the convolution unit for feature map of type-f on the ℓ^{th} layer is

$$\nu^i_{(\ell,f)} \overset{\text{def}}{=} \sigma(\mathbf{w}_{(\ell,f)}\vec{\nu}^i_{(\ell-1)} + b_{(\ell,f)}), \tag{9}$$

where $\mathbf{w}_{(\ell,f)}$ is the parameters for the f feature map on ℓ^{th} layer, σ is the nonlinear activation function, and $\vec{\nu}^i_{(\ell-1)}$ denotes the segment of $(\ell-1)^{th}$ layer for the convolution at location i , which is defined as follows.

$$\vec{\nu}^i_{(\ell-1)} \overset{\text{def}}{=} \nu^i_{(\ell-1)} \parallel \nu^{i+1}_{(\ell-1)} \parallel \cdots \parallel \nu^{i+s_{rp}-1}_{(\ell-1)}, \tag{10}$$

where s_{rp} defines the size of local “receptive field” for convolution.Choosing it as 3 for the convolution process results in the convolution unit shown in Figure 3 (a). The parameters within the convolution unit are shared for the whole question with a window covering 3 semantic components sliding from the beginning to the end. The input of the sentence CNN for the first convolution layer output vectors from the LSTM denoted as:

$$\vec{\nu}^i_{(0)} \overset{\text{def}}{=} \nu^i_{wd} \parallel \nu^{i+1}_{wd} \parallel \cdots \parallel \nu^{i+s_{rp}-1}_{wd}, \tag{11}$$

where $\nu^i_{(wd)} = \hat{a}_i$ is the LSTM output vector corresponding to the i^{th} word in the question.

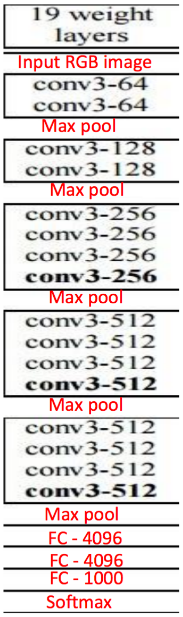


Figure 4. 19-layer VGGNet[Simonyan et al.]: the image CNN

After the convolution process, the sequential s_{rp} semantic components are composed to higher semantic representations. However, these composition may not be the meaningful representations. The max-pooling process following each convolution process is performed:

$$\nu^i_{(\ell+1,f)} = \max(\nu^{2i}_{(\ell,f)}, \nu^{2i+1}_{(\ell,f)}). \tag{12}$$

3.3. Image CNN

The image content is encoded into an image representation vector as follows:

$$\nu_{im} = \sigma(\mathbf{w}_{im}(CNN_{im}(I)) + b_{im}), \tag{13}$$

where σ is nonlinear activation function, which in our model is ReLU [Dahl et al.].

A number of choices are available for CNN_{im}. [Malinowski et al.(2015)] used [Googlenet] for this purpose. In this model, we employ the 19 layer VGGNet [Simonyan et al.] to encode the image content. This CNN model has produced more accurate results than Googlenet on image captioning tasks.

CNN_{im} takes the image as the input and output a fixed length vector as the image representation. By removing the top last ReLU layer and the softmax layer of the CNN [Simonyan et al.], the output of the last FC layer is passed on as the image representation. The dimension of this output which is 1000. $\mathbf{w}_{im}$ is the mapping matrix of the dimension $d \times 1000$, which provides twofold benefits:

- The dimension of the image representation is significantly reduced from 1000 to d. As such, the total number of parameters for the multimodal convolution layer can be significantly reduced. Consequently, and the size of the training set can be reduced.

- The dimension of the image should match the dimension of the composed output from the sentence CNN.

$\mathbf{w}_{im}$ reduce the dimension of image representation to further help the multimodal convolution process.

3.4. Multimodal Convolutional Layer

The image representation ν_{im} and question representation ν_{qt} are obtained by the image and sentence CNN, respectively. The multimodal convolution layer from [Ma et al.] is used for joining the multimodal inputs together to generate their joint representation for further answer prediction. The convolution unit for the multimodal CNN is shown in Figure 3 (b). Based on the image representation and the two consecutive semantic components from the question side, the mulitmodal convolution is performed, which is expected to capture the interactions and relations between the two multimodal inputs.

$$\vec{\nu}^{\,i}_{(6)} \overset{\text{def}}{=} \nu^{i}_{(6)} \parallel \nu_{im} \parallel \nu^{i+1}_{(6)}, \tag{14}$$

$$\nu^{i}_{(mm,f)} \overset{\text{def}}{=} \sigma(\mathbf{w}_{(mm,f)}\vec{\nu}^{\,i}_{(6)} + b_{(mm,f)}), \tag{15}$$

where $\vec{\nu}^{\,i}_{(6)}$ is the input of the multimodal convolution unit, $\mathbf{w}_{(mm,f)}$ and $b_{(mm,f)}$ are the parameters for the type-f feature map of multimodal convolution layer.

After the mutlimodal convolution layer, the joint representation ν_{jr} is obtained, which is fed into a softmax layer. The softmax layer generates the answer to the given image and question pair.

4. Experiments

We have evaluated our approach on the DAQUAR[Malinowski et al.(2015)] dataset that contains 12,468 questions on 1449 images of indoor scenes, and used WUPS score to evaluate the results as in [Malinowski et al.(2015)]. We have used [caffe] library and g2.2xlarge GPU instance(1,536 CUDA cores and 4GB of video memory) of Amazon web services EC2 for our experiments.

It takes around 1 hour to train a model for 1000 iterations on an Amazon EC2 instance for DAQUAR dataset with a train size of 6795 human question answer pair. We have evaluated the results by training the model for 1000, 5000 and 8000 iterations and compared them with [Malinowski et al.(2015)] on the same number of training iterations.

4.0.1 Configuration

We have used VGGNet[Simonyan et al.] to get the image CNN representation. For the sentence CNN representation, we have used three convolution layers and a maxpool layer as used by [Ma et al.]. The dimensions of these three layers are respectively 200, 300 and 300. Maximum length of allowed question is 30 and one <eos> word is added to represent the end of sentence. The multi-modal convolution neural network takes image representation as well as the sentence representation as input and finally generate a joint representation. Finally, softmax layer use this joint representation as input and generate a one hot vector encoding. It is then checked with the vocabulary to get the closest word. The vocabulary consists of all the answer words from the dataset.

#Iterations	Malinowski(%)	Our model(%)
1000	1.0701545	9.6313912
5000	1.9817677	13.1391200
8000	4.8949663	16.6270313

Table 1. Comparison of our model with [Malinowski et al.(2015)] using WUPS score

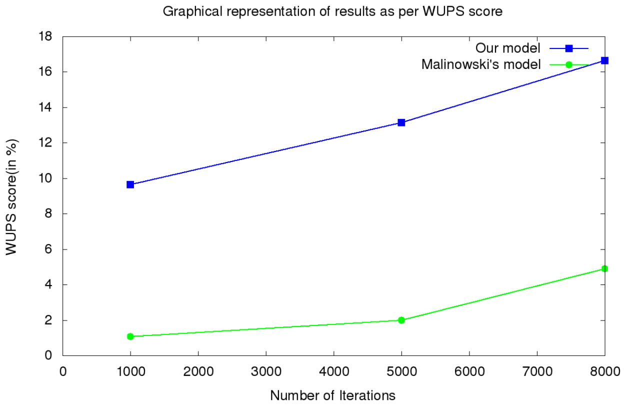


Figure 5. Comparison of results

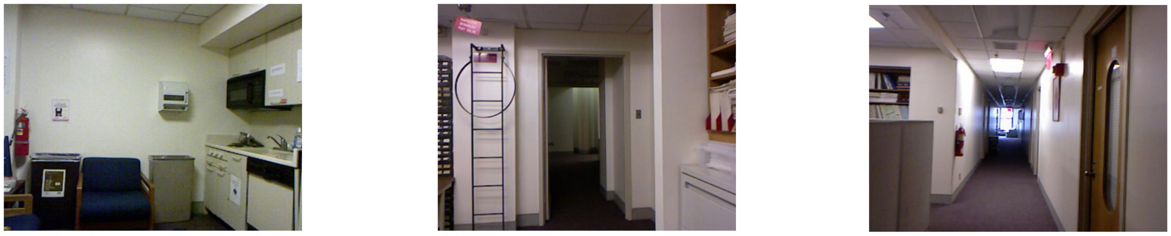
4.0.2 WUPS Score

We have evaluated the results using Wu-Palmer Similarity measure as in [Malinowski et al.(2015)]. WUPS metric is a generalization of the accuracy measure that accounts for word-level ambiguities in the answer words.

$$\text{WUPS}(A,T) = \frac{1}{N}\sum_{i=1}^{N}\min\{\prod_{a\in A^i}\max_{t\in T^i}\mu(a,t), \\ \prod_{t\in T^i}\max_{a\in A^i}\mu(a,t)\}$$

4.1. Results

The above table1 clearly indicate that our model results are well above those in [Malinowski et al.(2015)] for the same training iterations. For 110,000 training iterations, model in [Malinowski et al.(2015)] performs with a of WUPS score of 18.55% which is 2% more than the accuracy of our model trained through only 8000 iterations.



What is the object built on top of the counter right of the stove ?
Ground truth: sink
Model result: sink
Googlenet(8000 iterations) : lamp
Googlenet(110000 iterations): dishwasher

How many blue chairs are ?
Ground truth: 2
Model result:: 2
Googlenet(8000 iterations) : 2
Googlenet(110000 iterations): 2

How many red objects are visible ?
Ground truth: 5
Model result:: 2
Googlenet(8000 iterations) : 2
Googlenet(110000 iterations): 2

What is diagonally placed in front of the ladder?
Ground truth: table
Model result:: chair
Googlenet(8000 iterations) : chair
Googlenet(110000 iterations): chair

What are the objects close to the ceiling ?
Ground truth:exit_sign, fire_alarm, light
Model result:: light
Googlenet(8000 iterations) : 2
Googlenet(110000 iterations): spot_light

What is the colour of the door?
Ground truth: brown
Model result:: yellow
Googlenet(8000 iterations) : yellow
Googlenet(110000 iterations): yellow

Figure 6. Results of question and answers generated by our model

5. Conclusions

We have presented a model which extracts a better representation of the sentence input and its relation with the image input for the task of visual question answering. LSTM captures the sequential nature of the sentence, and CNN helps to model the higher semantic representations. Also, the image CNN representation is fed into both the LSTM and mulitmodal convolution layer which was not present in any previous work in this area.

Future work The single layer LSTM can be replaced by better models e.g. bidirectional LSTM for further improvement in modelling the interactions between image and sentence.

Acknowledgements. We would like to thank Prof. Amitabha Mukherjee for his valuable guidance and for the resources provided in terms of GPU.

References

[Antol et al.] Stanislaw Antol and Aishwarya Agrawal and Jiasen Lu and Margaret Mitchell and Dhruv Batra and C. Lawrence Zitnick and Devi Parikh 2015. VQA: Visual Question Answering. *CoRR*.

[Malinowski et al.] M. Malinowski and M. Fritz 2014a. A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input. *NIPS 2014*.

[Ma et al.] Lin Ma and Zhengdong Lu and Hang Li 2015. Learning to Answer Questions From Image using Convolutional Neural Network. *CoRR*.

[Malinowski et al.(2015)] Mateusz Malinowski and Marcus Rohrbach and Mario Fritz 2015. Ask Your Neurons: A Neural-based Approach to Answering Questions about Images. *CoRR*.

[Gao et al.] Haoyuan Gao and Junhua Mao and Jie Zhou and Zhiheng Huang and Lei Wang and Wei Xu 2015. Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering. *CoRR*.

[Kiros et al.] Mengye Ren and Ryan Kiros and Richard S. Zemel 2015. Image Question Answering: A Visual Semantic Embedding Model and a New Dataset. *CoRR*.

[Zaremba et al.] W. Zaremba and I. Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

[caffe] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[Donahue et al.] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[Simonyan et al.] K. Simonyan and A. Zisserman 2014. Very Deep Convolutional Networks for Large-scale Image Recognition. *arXiv 1409.1556*.

[Dahl et al.] G. E. Dahl, and T. N. Sainath and G. E. Hinton 2013. Improving Deep Neural Networks for LVCSR Using Rectified Linear Units and Dropout. *ICASSP 2013*.

[Googlenet] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014.