Visual Question Answering

Aman Garg 14073
Deepanshu Bansal 150219
Gopichand Kotana 14249
Jayant Agrawal 14282
Jayendra Kumar 14283
Shubham Pandey 14679
Sunil Pandey 14723

1 Problem Statement

Visual Question Answering (VQA) involves answering natural-language visual questions about an image. VQA is a cross disciplinary research problem which requires understanding of both the image and the questions related to it. Although the task is closely related to image captioning, which is fairly well studied, VQA differs from it as VQA requires the answer to be conditioned on the image and the question. The literature survey section lists out several existing methods to solve this problem. It is interesting to note that Bayesian model such as Answer Type prediction[1] outperforms the deep models with no attention. This shows the importance of identifying parts of image and question that are important to answering. In this project, we will build on some of the existing models and, propose an architecture for VQA.

2 Datasets

2.1 VQA [7]

This is the most widely used dataset for the task. It has two parts, one part contains images from MS-COCO, and the other contains clipart images created from models of humans and animals. The evaluation can be done on MCQs, with 18 options each, or, open ended questions for which the paper mentions a method of evalution.

2.2 Visual7W [9]

This dataset is generated from images of the MS-COCO dataset. The dataset obtains it's name from the kind of questions present in it. All the multiple choice questions are of the form Who, What, Where, When, Why, How and Which. The dataset contains 47,300 images and 327,939 questions.

2.3 COCO-QA[10]

This dataset is also generated from MS-COCO dataset. The answers are one word which allows us to treat this problem as a classification problem. There are mainly three types of questions in this dataset: object type, object color, and number of objects.

3 Literature Survey

3.1 Non-Deep Learning Models

3.1.1 Answer Type Prediction

This approaches classifies different answers into few predefined type. The datasets like DAQUAR and MS-COCO doesn't have explicit categorization of the data so for training and testing we have to explicitly define categories. For DAQUAR the paper [1] created three categories by looking at the answers: Number, Color, and Other. similarly for COCO they have additional category activity which includes verbs like playing, dancing etc.

For doing this the paper used Bayesian framework. For given image features q and question features x, the method predicts the probability of particular answer k and answer type c.

$$P(A = k, T = c|x, q) = \frac{P(x|A = k, T = c, q)P(A = k|T = c, q)P(T = c|q)}{p(x|q)}$$

The paper chooses to train each of the distribution separately instead of doing it by maximum likelihood. The numerator is trained using logistic regression while denominator is not influencing the result hence ignored. The approach shared similarity with attention based models and pays attention to the parts of images which are related to question.

3.1.2 Multi world QA

The paper [2] combines natural language input with the output of the visual scene using probabilistic framework. First they start with single world approach then they extended the method for multi world. The figure 1 shows the step involved in algorithm.

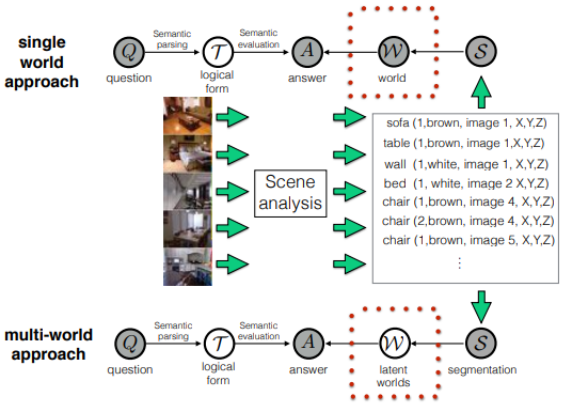

Figure 1: Multi World QA: Image Credits: [2]

The model builds over recent advancement on end to end question answering systems which trained on only question answer pairs(Q,A). The following mathematical formulation of latent features T of question and given a world W models our requirements well.

$$P(A|Q, W) = \sum_{T} P(A|T, W)P(T|Q)$$

In the perceived world we segment the image using state of the art image segmentation techniques, and hence our world is now populated with facts derived from automatic, semantic image segmentations S. Now based on our world view we assign different segments to different predicates (objects here) and then finally use this for answering. In multi world approach we do soft assignments in class labeling and sum the results over different perceived worlds.

3.2 Deep Learning: Non Attention

To counter huge biases in language we will use these methods and balanced dataset (with complementary images) ie. we have pair of two similar images with different answers which reduces the bias significantly.[3] Some models we will be using are :-

Deeper LSTM Question + norm Image (d-LSTM+nI) [7]

This model used l_2 normalized last hidden layer of VGGNet as an embedding for the image. An LSTM with two hidden layers is used to obtain 2048-dim (we will try on 1024 and 4096 as well) embedding for the question. The embedding obtained from the LSTM is a concatenation of last cell state and last hidden state representations (each being 512-dim (will try 256 as well)) from each of the two hidden layers of the LSTM. Hence 2 (hidden layers) x 2 (cell state and hidden state) x 512 (dimensionality of each of the cell states, as well as hidden states). This is followed by a fully-connected layer + tanh non-linearity to transform 2048-dim embedding to 1024-dim. Basically we will use a CNN embedding of the image, a Long-Short Term Memory (LSTM) embedding of the question, combines these two embeddings via a point-wise multiplication, followed by a multi-layer perceptron classifier to predict a probability distribution over 1000 most frequent answers in the training dataset.

3.3 Deep Learning: Attention

3.3.1 Joint Attention learning on Multi-Level Attention Network

Joint attention learning works over a multi-level attention network (MLAN)[4] which has following subparts:

- Semantic Attention
- Context-aware Visual Attention
- Joint Attention Learning

Semantic Attention aims at finding important high level concepts in the image which may be related to the question. Firstly, a concept detector is trained using deep CNN, which would then produce the probability of semantic concepts for an image. After that an attention network is trained to measure the semantic relevance between each concept in the vocabulary and the question.For doing so, we represent the question using an RNN. Now, the high-level semantic information of image queried by question Q could be represented by a weighted sum over all concepts representation.

Context-aware Visual Attention aims at representing important spatial context for image regions. For this, firstly the context information is incorporated into the representation from each region by a bidirectional GRU encoder. The fine-tuned CNN model for concept detection is now used to extract visual features for local regions. A feature map of the last convolutional layer in the CNN model is used as our visual representation. These feature vectors are fed into the bidirectional GRU and the output from the forward and backward direction at each step is combined to form a new feature vector for each region.

After these two attention steps, a joint learning step is invoked which aims at searching information in the image at different levels based on questions. At low-level visual feature, it focus on question related regions by visual attention; whereas at high level semantic feature, it focus on question related concept by semantic attention.

To generate an answer to the asked question, firstly, question vector are added into attended image features extracted from different layers. Then, element-wise multiplication is done to combine the two types of attentions together. In the end, the joint feature is fed into a softmax layer to predict the probability of predefined candidate answer set.

3.3.2 Question Guided Attention based Joint Embedding

This model involves a simple joint embedding with question guided attention. Though, the model is simple in it's structure when compared to its counterparts, it still achieves state-of-the-art results

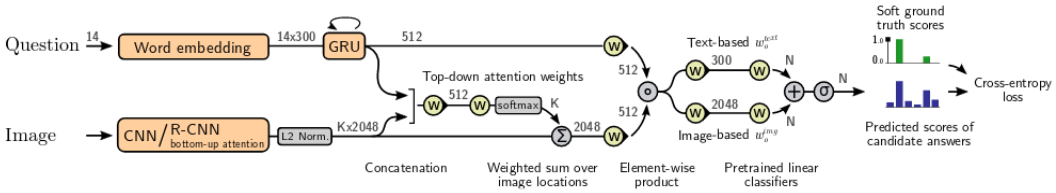

Figure 2: Question Guided Attention based joint embedding: Image Credits: [5]

on VQA. The complete model is showed in Figure 2 taken from the original paper [5].

The question is stripped to a length of 14 words where each word vector initialized to 300-dimensional GloVe word-embeddings. This gives a 14×300 question embedding, which is then passed through a GRU with internal state of size 512 to give the final question embedding, q of size 512. Image features, v are obtained using *bottom-up* attention as described in [6]. These are of size $K \times 2048$ where K is the number of image locations and each row, v_i represents the feature vector for the i^{th} image location.

To get the **attention weights**, each location vector v_i is concatenated with the question embedding, q which is then passes through a non-linear layer followed by softmax(for normalization) to attain $\alpha = \{\alpha_1, \alpha_2 ... \alpha_K\}$ where α_i represents the attention weight for the i^{th} image location. A 1-D 2048-sized image embedding is obtained by computing the weighted sum of features of all image locations weighted by attention weights. This is then passed through another non-linear layer to obtain a 512-sized image vector, $\tilde{v}$.

A **Joint Embedding**, h is obtained by taking an element-wise multiplication of the question embedding, q and the image embedding, $\tilde{v}$. The classification scores over the candidate answers are obtained as follows:

$$\tilde{s} = \sigma(w^{text} f^{text}(h) + w^{image} f^{image}(h))$$

where f^{text} and f^{image} are non-linear transformations which bring h to sizes of 300 and 2048 respectively. w^{text} and w^{image} are initialized as follows:

- Each row of w^{text} is initialized using the GloVe embedding of the corresponding candidate answer as the classification score of that answer is determined by that row of w^{text}.
- Similarly, each row of w^{image} is initialized using an image embedding which is related to that candidate answer. Top-10 images are extracted from Google images for that answer, passed through a CNN and the final embedding is averaged over all the images to give the image embedding for that row of w^{image}.

3.3.3 Visual7W: Grounded Question Answering in Images [9]

In this model at the encoding stage, the model reads the image and the question tokens word by word. To decide which region to focus on at each word, the attention term is learned using the convolutional feature map and the previous hidden map. At the decoding stage, the answer is picked by computing the log-likelihood of an answer by a dot product between its feature map and last LSTM hidden state.
The 7W questions on which this model is trained to work roughly correspond to an array of standard vision tasks: what, where, when, who, why, how and which. The Visual7W dataset features richer questions and longer answers than VQA.
Though LSTM models have achieved state-of-the-art results in several sequence processing tasks and have also been used to tackle visual QA tasks but it has lacked the mechanism to understand attention corresponding to local regions. This model overcomes this by adding the spatial attention to the standard LSTM model for visual QA illustrated in fig. 3. The model aims to capture the scenario when the answers to image-related questions is usually associated with specific image regions. It learns to obtain the attention corresponding to local regions as it reads the question tokens in a sequence. This model has achieved state-of-the-art performance with 55.6%, and found correlations

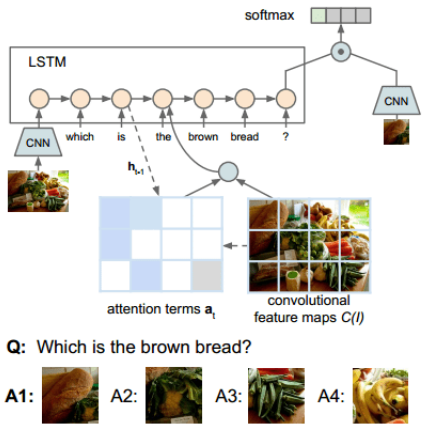

Figure 3: Diagram of the recurrent neural network model for pointing QA.

between the models attention heat maps and the object groundings.
The authors consider QA as a two-stage process. At the encoding stage, the model memorizes the image and the question into a hidden state vector(the gray box in Fig. 3). At the decoding stage, the model selects an answer from the multiple choices based on its memory (the softmax layer in Fig. 3).

3.4 Knowledge Based Model [8]

The methods discussed above only capture the information that is available from the Training Dataset, and given the vast information available, can never cover the information of the real world. Sometimes, VQA require prior information which can be in the form of encyclopedic or even 'common sense'. To overcome this, people have tried working with structured representation of knowledge in the form of a Knowledge Base which store common sense and factual knowledge in a program-interpretable format. Here, we'll talk about Wang *et al*'s paper, which uses DBpedia - one of the largest structured knowledge bases.

Knowledge-Based enhanced Dataset
The author uses KB-VQA Dataset, which was constructed in such a way that questions requiring topic-specific knowledge were present in DPpedia. 700 images were selected from the COCO image datase and 3 to 5 question/answer pairs were collected for each, for a total of 2,402 questions. Each question follows one of 23 predefined templates. The questions require different levels of knowledge, from common sense to encyclopedic knowledge.

Method
The author calls his framework the Ahab approach which begins with construction of Resource Description Framework (RDF) graph. The RDF graph detects concepts in the query image and links them to the relevant parts of the Knowledge Base. Visual concepts like Objects, Image Scenes and Image attributes detected in query image, which are then linked to the appropriate information in DBpedia. Having gathered all of the relevant information from the image and DBpedia, he then uses them to answer questions
Now give the question in Natural Language, it is translated to query format for RDBMS. Each word in the question is tagged using NLTK, tokenized, POS-tagged, and lemmatized. It is then parsed through a set of regular expressions to slot the phrases to relevant groups of information. These groups of information is then mapped to the Knowledge-based entities. Finally, for answering and reasoning, statements are generated according to the results of the queries. Post-processing of the responses is done to bring it answers to the required format.

4 Proposed Approach

4.1 Joint Embedding with Image Guided Attention on Question

The model in Figure 2 by Teney et.al focuses on Question based attention to calculate Image Attention. However, the question embedding in this model is calculated without taking the image into account which can be very important. For example: Consider two questions: "How many balls are there in the box?", and "How many balls can you see in the box?". Both the questions convey the same meaning, but the second question just introduces more noise in the final question embedding.

If we have an attention weight on each word in the question, this problem can be solved and we can have a more accurate word embedding for the question which is relevant to the image. We propose to use the image to calculate attention over each word in the question.

To achieve this, we can use the final image embedding, $\tilde{v}$ along with the word embedding of each word, w_i in the question to get the attention weights as:

$$\beta_i = W.f([w_i, \tilde{v}])$$

$$\tilde{\beta} = softmax(\beta)$$

We can then use these weights to compute the weighted question embedding and then use that as the final embedding. This is shown in Figure 4. This approach of calculating attention for question is general and can be used with any model. We experiment on this module with the model given by *Teney et.al* and show the results later in Section 5.2. Our complete model is shown in Figure 5.

Figure 4: Image Guided Attention on Question

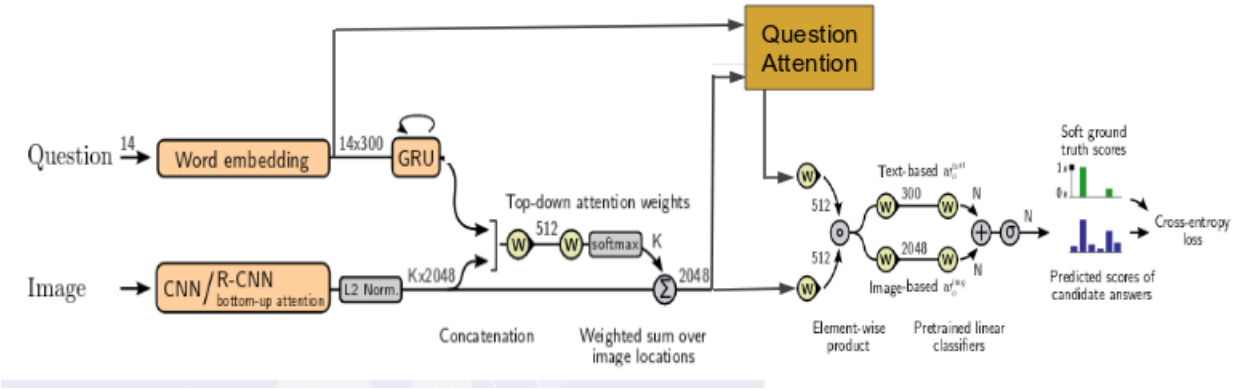

Figure 5: Joint Embedding Model with attention on question and image

4.2 Answer Type Prediction

Bayesian framework to VQA would give us the most probable answer given the image and the question. Our aim is to find $P(A = k|x, q)$, through which we find the k with the largest probability conditioned on the image and the question
[1] proposes a bayesian approach to VQA where we model the following probability

$$P(A = k, T = c|x, q) = \frac{P(x|A = k, T = c, q)P(A = k|T = c, q)P(T = c|q)}{P(x|q)}$$

where A is the answer, T is the type of question (in our case - object, number, location and colour), x is the feature vector of the Image and q is the feature vector of the Question asked.

$P(A = k|x, q)$ is obtained, by marginalizing out the random variable T

$$P(A = k|x, q) = \sum_{c \in T} P(A = k, T = c|x, q)$$

RHS of the equation is obtained by applying the Bayes' rule and the chain rule of probability, so we model the probability on the LHS by modelling the probabilities on the RHS

$P(A = k|T = c, q)$ and $P(T = c|q)$ are modelled using Logistic Regression classifiers. $P(x|q)$ does not influence the prediciton of A, so it is treated as a constant.

We model each $P(x|A = k, T = c, q)$ with a conditional multivariate Gaussian,

$$P(x|A = k, T = c, q) = \mathcal{N}(x|\bar{\mu}_{k,c,q}, \bar{\Sigma}_{k,c})$$

Let the sample mean and covariance for the training data with answer k and answer-type c, in which the image features x are concatenated with the question features q, be

$$\mu_{k,c} = [\mu_{k,c,x}, \mu_{k,c,q}]$$

and

$$\Sigma_{k,c} = \begin{bmatrix} \Sigma_{k,c,1,1} & \Sigma_{k,c,1,2} \\ \Sigma_{k,c,2,1} & \Sigma_{k,c,2,2} \end{bmatrix}$$

Then, the mean of the Gaussian given q will be

$$\bar{\mu}_{k,c,q} = \mu_{k,c,x} + \Sigma_{k,c,1,2}\Sigma^{-1}_{k,c,2,2}(q - \mu_{k,c,q})$$

and the covariance will be

$$\bar{\Sigma}_{k,c} = \Sigma_{k,c,1,1} - \Sigma_{k,c,1,2}\Sigma^{-1}_{k,c,2,2}\Sigma_{k,c,2,1}$$

Using these probability estimates, we can find the k for which $P(A = k|x, q)$ is maximized.

This model is not technically a Bayesian model, as we are not calculating the exact form of the posterior distribution, but rather doing an MLE estimate. We used this simple probabilistic model as a Baseline for other models.

4.3 Deep Bayesian Model

Intuition: When you are asked a question of the kind present in COCO-QA, you build up an image in your head to be able to answer it. You don't build a clear picture with details, but you essentially try to capture the semantics that are needed to answer it. We wanted to test if we can achieve similar effect using a neural network. This is a novel approach proposed by us, and the architecture is inspired by that of a VAE.

Training: We train using the question embedding and answer embedding passed together as input. We expect the network to give an image embedding close to that of the one we obtained from the fc7 layer of VGG16. This is reflected in the loss which is a sum of KL divergence which ensures the mu, sigma is close to that of a unit gaussian and a mean square error between the image embedding given by the network and the one obtained through VGG16.

Testing: The COCO-QA can have 430 possible answers for a question. We append the embedding of each answer to the question embedding and we check the similarity of the produced embedding to the true embedding. We also have $P(Answer|Question)$ prior present which we

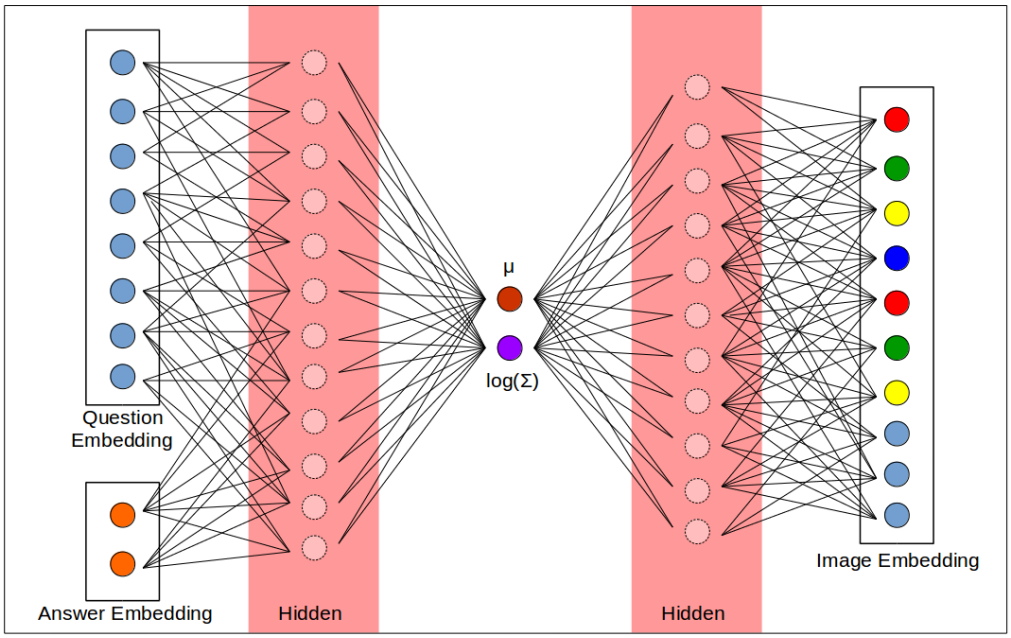

Figure 6: Model proposed

learn by logistic regression. We multiply this with the similarity and the answers with top 10 highest probabilities are chosen. We check if the answer exists in the ten. This evaluation gives us an accuracy around 23% and a slight improvement of 2% when done using skip thought vectors.

Suggested improvements: Accuracy takes a major hit because of the sampling step in between. Sample more than one time, k times, depending on the computing resources available. Discard the ones which has too much variance. Average the rest. This might produce a better embedding than sampling once. Training the model, use dropout, more data and for more epochs. Training is currently done with 50000 examples for 10 epochs. Construct a better loss function than mean squared error. We couldn't incorporate these due to shortage of computing resources and time.

5 Experiments

5.1 Embeddings

The following embeddings for image and questions were used for our different models:

5.1.1 Image Features

We used Fc7 layer of pre trained VGG16 are used as the features for the generative model.

5.1.2 Question Features : using GLoVe

We took GLoVe vector for each word of size 50. Question length was taken to be of 7 words. First we consider the important words in the question and then add the stop words to make for a length of 7. We then combine their GLoVe vectors to represent the question feature.

5.1.3 Question Features : Skip-Thought Vectors

Skip-Thought Vectors [11] are inspired from skip-gram structure used in word2vec. In contiguous text, nearby sentences provide rich semantic and contextual information. The learned sentence vectors are highly generic, and can be reused for many different tasks by learning an additional mapping, such as a classification layer.

The Skip-Thought model is based on an encoder-decoder architecture. All variants of this architecture share a common goal: encoding source inputs into fixed-length vector representations, and then feeding such vectors through a narrow passage to decode into a target output.

Like uni-skipgram and bi-skipgram variants of skip-gram structure we used uni-skipThought and bi-skipThought variants for embedding questions. The feature vectors were 4800 dimensional, where the first 2400 dimensions were of the uni-skip model, and the last 2400 were of the bi-skip model.

5.2 Joint Embedding with Image Guided Attention on Question

The length of each question is fixed and trimmed to 14 words. The first 14 words are considered after stop-word removal and if the length falls short, appropriate padding is done. Each word embedding of size 300. We use GLoVe Embeddings for words.

For images, COCO Images are passed through a Faster RCNN based framework as done by *Teney et.al* to obtain region embeddings. Number of regions is 36 each with each region embedding having a feature vector of size 2048. The size of the final joint embedding is 512. Our final model can be seen in Figure 5. For these experiments, 10k examples (Question-Image) were used for training and 512 were used for validation. The results are shown in Figure 7 and an improvement of 3% is observed under the given constraints.

Model	Accuracy (3 epochs)
Joint Embedding with attention on Image Only (Baseline)	25%
Joint Embedding with attention on Image and Question	28%

Figure 7: Results:Joint Embedding Model with attention on question and image

5.3 Answer Type Prediction Model

Image Features were used as mentioned in Section 5.1.1, and Question Features were obtained using GLoVe as mentioned in Section 5.1.2.
Sklearn's Logistic Regression classifier and Scipy.stats' multivariate normal distribution were used for the probabilities.
An accuracy of approx 18% was obtained with a 10% test-train split on 23,516 question-image pair.

6 Conclusion

We experimented with three different kind of models. First, we experimented with an attention based approach and proposed a method to compute attention on question using image(Figure 4). This module is simple and general enough, so it can be used with all kinds of models. We have shown the effectiveness of the same with a joint-attention model(Figure 5).

Then we consider a bayesian approach in the answer type prediction framework. We also look at a deep bayesian model and show its effectiveness. VQA is a highly difficult problem and remains largely unsolved. We present some approaches which might help researchers working on this problem in future.

7 Future Work

Since the initial experiments show promise, we would like to conduct more experiments on the image-guided question attention model. We plan to train and test on the entire COCO-QA dataset and compare our results with other attention based models.

8 Contribution Distribution

Name	Contribution
Aman Garg	14%
Deepanshu Bansal	14%
Gopichand Kotana	15%
Jayant Agrawal	15%
Jayendra Kumar	14%
Shubham Pandey	14%
Sunil Pandey	14%

References

[1] Kushal Kafle and Christopher Kanan. 2016. *Answertype prediction for visual question answering.* In CVPR.

[2] Mateusz Malinowski and Mario Fritz. 2014. *A multiworld approach to question answering about realworld scenes based on uncertain input.* In NIPS.

[3] Making the V in VQA Matter, Yash Goyal *et al*.

[4] Multi-level Attention Networks for Visual Question Answering Dongfei Yu *et al*.

[5] *Tips and Tricks for Visual Question Answering: Learnings from the 2017 Challenge* Damien Teney, Peter Anderson, Xiaodong He , Anton van den Hengel arXiv Pre-Print arXiv:1708.02711v1 [cs.CV]

[6] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. *Bottom-up and top-down at- tention for image captioning and vqa.* arXiv preprint arXiv:1707.07998 , 2017.

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

[8] *Explicit Knowledge-based Reasoning for Visual Question Answering*, Peng Wang, Qi Wu , Chunhua Shen, Anton van den Hengel, Anthony Dick. 2015. arXiv:1511.02570

[9] *Visual7w: Grounded question answering in images* Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei- Fei. In CVPR, 2016.

[10] *Exploring Models and Data for Image Question Answering*, Mengye Ren, Ryan Kiros, and Richard Zemel, NIPS 2015.

[11] *Skip-Thought Vectors*, Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, Sanja Fidler, 2015