Natural Language for Visual Question Answering

by

Debleena Sarkar
Roll Number: CS1605

Under the supervision of

Prof. Utpal Garain
Computer Vision and Pattern Recognition

A thesis submitted in partial fulfillment for the degree of
Masters of Technology
in
Computer Science

Indian Statistical Institute
Kolkata-700108, India

May 29, 2018

Declaration of Authorship

I, **Debleena Sarkar**, declare that this thesis titled, **"Natural Language for Visual Question Answering"** and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

CERTIFICATE

This is to certify that the dissertation entitled **"Natural Language for Visual Question Answering"** submitted by **Debleena Sarkar** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of Master of Technology in Computer Science is a bonafide record of work carried out by her under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Signed: _____

Date: _____

ABSTRACT

Visual reasoning with compositional natural language instructions, as described in the newly-released Cornell Natural Language Visual Reasoning (NLVR) dataset[1], is a challenging task, where the model needs to have the ability to create an accurate mapping between the diverse phrases and the several objects placed in complex arrangements in the image. Natural language questions are inherently compositional, and can be answered by reasoning about their decomposition into modular sub-problems. In the recently proposed End-to-End Module Networks (N2NMNs)[2] the network tries to learn to predict question specific network architecture composed of set of predefined modules. The model learns to generate network structures (by imitating expert demonstrations) while simultaneously learning network parameters. We have implemented the N2NMN model for NLVR task. By visualizing the N2NMN model on the NLVR dataset, we have found that the model is unable to find out correspondence between image feature and textual feature. We have proposed modification in the N2NMN model to capture better mapping between image and textual feature.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor Prof. Utpal Garain for encouraging me to pursue research in neural networks and for his constant guidance and support.

My sincere thanks to Akshay Chaturvedi for his valuable suggestions and discussions.

I am also thankful to my parents, sister, friends, and family for all their support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Visual Reasoning

Visual reasoning requires sophisticated understanding of the language used in question answering and the relationship with the corresponding image. The visual Question Answering [3, 4] task has significant applications to human-robot interaction, search, and accessibility, and has been the subject of a great deal of recent research attention.

1.2 The Dataset

1.2.1 NLVR dataset

Understanding complex compositional language with context is a challenging job shared by many tasks. In the paper[1] recently a challenging new NLVR (Natural Language for Visual Reasoning) task has been proposed. The dataset with natural and complex language statements that have to be classified as true or false given a multi-image set, shown in Figure 1.1. Specifically, each task instance consists of an image with three sub-images and a statement which describes the image. The task of the model is the binary prediction if the statement is true with respect to the image or not.

This data includes 92,244 sentence-image pairs with 3,962 unique sentences. It includes both images for raw visual information and the structured representation. Figure 1.1 shows two examples. Each image includes three boxes with different object types. The truth value of the top sentence is true, while the bottom is false.

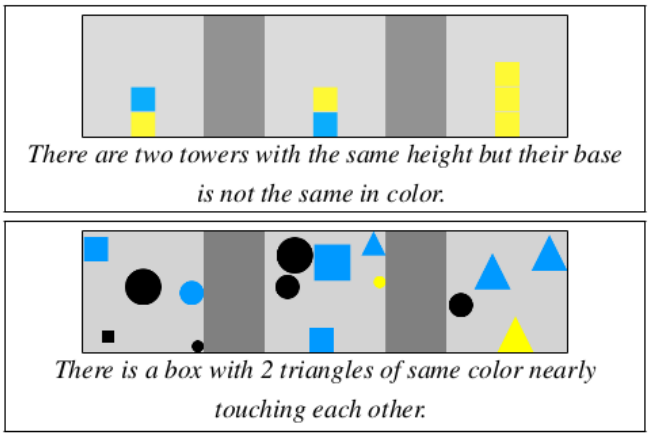


FIGURE 1.1: Example sentences and images from NLVR corpus.

The data statistics of the NLVR dataset is given in the Table 1.1. The data is divided into 4 main segments: Train (Training dataset), Dev (Development dataset), Test-P (Published test dataset), Test-U (Unpublished test dataset).

Dataset	**Unique Sentences**	**Examples**
Train	3,163	74,460
Dev	267	5,940
Test-P	266	5,934
Test-U	266	5,910
Total	3,962	92,244

TABLE 1.1: Data Statistics of NLVR dataset

1.2.2 SHAPES dataset

The SHAPES dataset for visual question answering[5] consists of 15616 image-question pairs with 244 unique questions. Each image consists of shapes of different colors and sizes aligned on a 3 by 3 grid. Despite its relatively small size, effective reasoning is needed to successfully answer questions like "is there a red triangle above a blue shape?" The example of such dataset is given in Figure 1.2. The truth value of the top figure is "yes", while the bottom is "no".

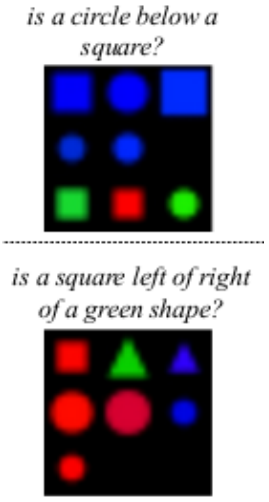


FIGURE 1.2: Example sentences and images from SHAPES corpus.

1.3 Challenges

In spite of being a binary classification, NLVR is a complex task because of the dataset. In the dataset, we have mainly 2 types of images - scatter images and tower images. In the Figure 1.1 the first box corresponds to the tower image and the next box corresponds to the scatter image. The data demonstrates a broad set of linguistic phenomena, requiring visual and set-theoretic reasoning. The example of such linguistic phenomena is explained with examples in Table 1.2. Another challenge is that the dataset has a lot of spelling mistakes which has to be taken care of in the pre-processing phase.

	Examples
Semantics	
Cardinality(hard)	There are **exactly four objects** not touching any edge
Cardinality(soft)	There is a box with **at least one** square and at least three triangles
Existential	**There is a tower** with yellow base.
Universal	There is a black item in **every box.**
Coordination	There are 2 blue circles **and** 1 blue triangle
Coreference	There is a blue triangle touching the wall with **its** side.
Spatial relation	there is one tower with a yellow block **above** a yellow block
Comparative	There is a box with multiple items and only one item **has a different color.**
Presupposition	There is a box with seven items and the **three black items** are the same in shape.
Negation	there is exactly one black triangle **not touching** the edge.
Syntax	
Coordination	There is a box with at least one square **and** at least three triangles.
PP-Attachment	There is a black block on a black block as the base of a tower **with** three blocks.

TABLE 1.2: Qualitative and empirical analysis of NLVR dataset

1.4 Outline

The main aim of this dissertation is to implement and modify the state-of-art architectures for the NLVR task. The rest of this report is organized as follows.

In **chapter 2**, we have discussed the previous work - the architecture of Neural Modular network(NMN), the already applied methods on NLVR dataset and the architecture of End-to-End Neural Modular Network(N2NMN).

In **chapter 3**, we have discussed how the N2NMN model has been applied to NLVR dataset and the results from the experiments.

In **chapter 4**, we have explained the observations from the visualization of the model on SHAPES and NLVR dataset.

In **chapter 5**, we have proposed some modification in the existing N2NMN model and explained the results obtained from the modifications.

In the **final chapter**, we have discussed the future directions of the problem.

Chapter 2

Previous Work

2.1 Neural Modular Network(NMN)

The neural module networks approach is explained in [5]. Each training datum for this task can be thought of as a 3-tuple (w, x, y), where

- w is a natural-language question
- x is an image
- y is an answer

A model is fully specified by a collection of modules $\{\mathrm{m}\}$, each with associated parameters θ_m , and a network layout predictor P which maps from strings to networks. Given (w, x) as above, the model instantiates a network based on P(w), passes x (and possibly w again) as inputs, and obtains a distribution over labels(for the VQA task, we re quire the output module to be a classifier). Thus a model ultimately encodes a predictive distribution $P(y|w, x; \theta)$.

2.1.1 Modules

A small set of modules are identified that can be assembled into all the configurations necessary for the tasks. This corresponds to identifying a minimal set of composable vision primitives. The modules operate on three basic data types: images, unnormalized attentions, and labels. The details of the modules are discussed in Table 2.1.

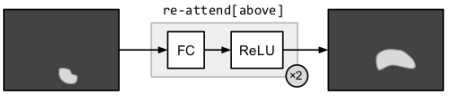
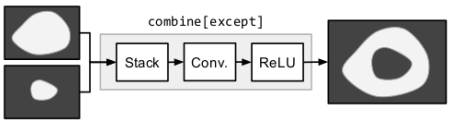
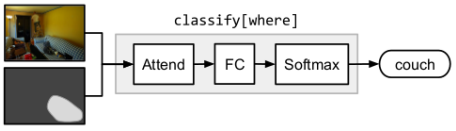
Module Name	Description	Implementation
Attention	$attend : Image \rightarrow Attention$	
Re-attention	$re - attend : Attention \rightarrow Attention$	
Combination	$combine : Attention \times Attention \rightarrow$ Attention	
Classification	$classify : Image \times Attention \rightarrow$ Label	
Measurement	$measure : Attention \rightarrow Label$	

TABLE 2.1: Modules used in Neural Modular Network

2.1.2 Model Formation

The schematic representation of the module is expressed in Figure 2.1 This approach first analyzes each question with a semantic parser, and uses this analysis to determine the basic computational units (attention, classification, etc.) needed to answer the question, as well as the relationships between the modules. All modules in an NMN are independent and composable, which allows the computation to be different for each problem instance, and possibly unobserved during training. Outside the NMN, our final answer uses a recurrent network (LSTM) to read the question.

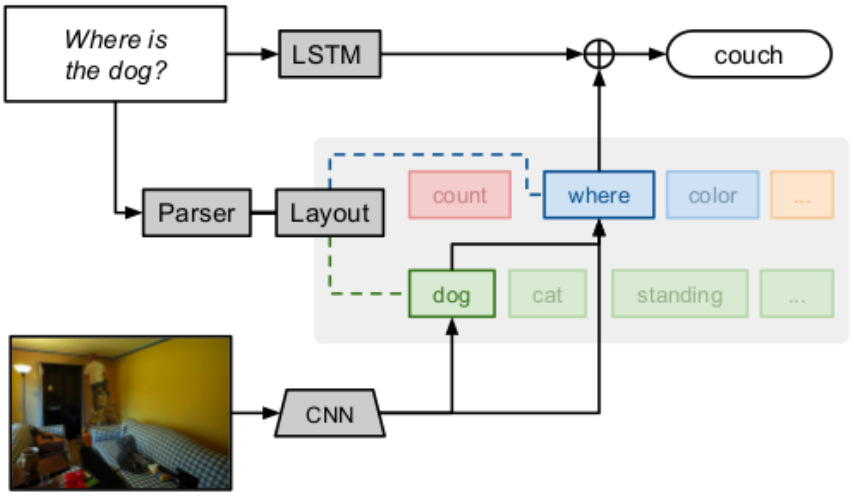


FIGURE 2.1: Schematic representation of Neural Modular Network

For supervised training, layout of each question is given to the model. Layout are the symbolic representations already determine the structure of the predicted networks, but not the identities of the modules that compose them. The layout with respect to the question "Is there a red shape above a circle?" is explained in figure 2.2. The two attend modules locate the red shapes and circles, the re-attend[above] shifts the attention above the circles, the combine module computes their intersection, and the measure[is] module inspects the final attention and determines that it is non-empty.

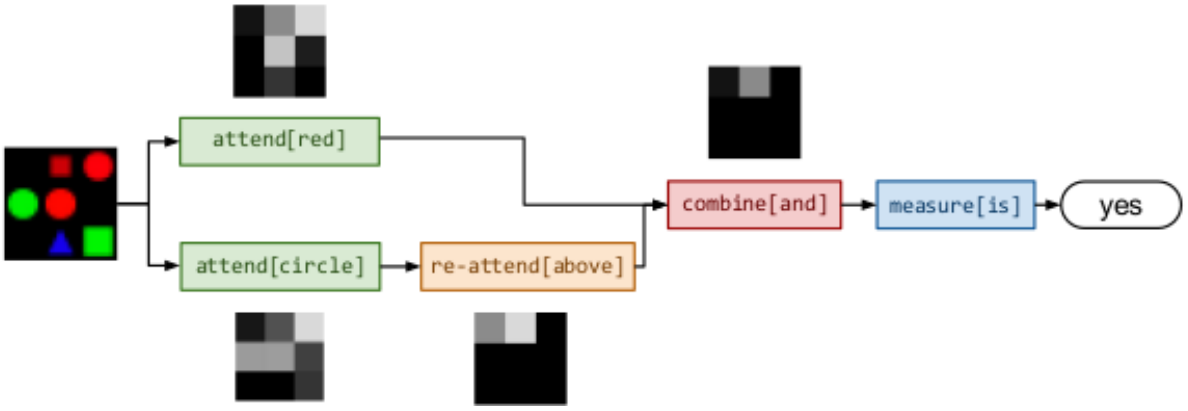


FIGURE 2.2: Example of Layout from SHAPES dataset

2.2 Methods applied on NLVR dataset

The evaluation of multiple methods on the rendered images and structured representations according to [1].The accuracy performed by different methods performed are listed in the Picture 2.3.

		Train	Dev	Test-P	Test-U
	Majority	56.37	55.31	56.16	55.43
	Text only	58.36 ±0.6	56.61 ±0.5	57.18 ±0.6	56.21 ±0.4
	Image Only	56.79 ±1.3	55.35 ±0.1	56.05 ±0.3	55.33 ±0.3
Structured representation	MaxEnt	99.99	68.04	67.68	67.82
	MLP	96.15 ±1.3	67.50 ±0.5	66.28 ±0.4	65.32 ±0.4
	Image features+RNN	59.71 ±1.0	57.72 ±1.4	57.62 ±1.3	56.29 ±0.9
Raw image	CNN+RNN	58.85 ±0.2	56.59 ±0.3	58.01 ±0.3	56.30 ±0.6
	NMN	98.37±0.6	63.06 ±0.1	66.12 ±0.4	61.99 ±0.8

FIGURE 2.3: Mean accuracy and standard deviation results applied on NLVR dataset

2.3 End-to-End Neural Modular Network(N2NMN)

This model synthesizes and extends two recent modular architectures for visual problem solving. Standard neural module networks (NMNs) [5] already provide a technique for constructing dynamic network structures from collections of composable modules. However, previous work relies on an external parser

to process input text and obtain the module layout. This is a serious limitation, because off-the-shelf language parsers are not designed for language and vision tasks and must therefore be modified using handcrafted rules that often fail to predict valid layouts.

A class of models are capable of predicting novel modular network architectures directly from textual input and applying them to images in order to solve question answering tasks. In contrast to Neural Modular Network(NMN), N2NMN[2] architecture learns to both parse the language into linguistic structures and compose them into appropriate layouts.

The architecture of the model N2NMN is explained in the figure 2.4. This model first computes a deep representation of the question, and uses this as an input to a layout-prediction policy implemented with a recurrent neural network. This policy emits both a sequence of structural actions, specifying a template for a modular neural network in reverse Polish notation, and a sequence of attentive actions, extracting parameters for these neural modules from the input sentence. These two sequences are passed to a network builder, which dynamically instantiates an appropriate neural network and applies it to the input image to obtain an answer.

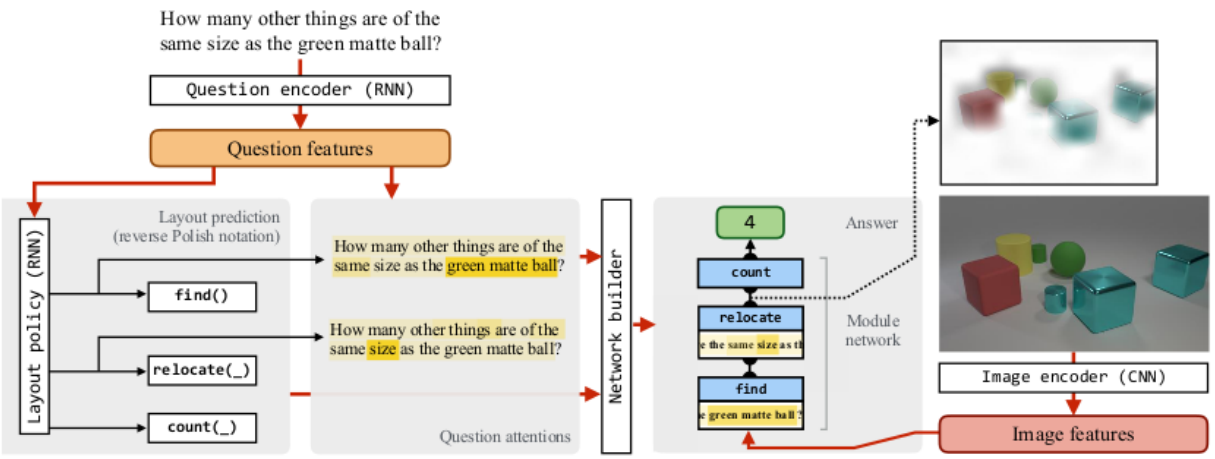


FIGURE 2.4: N2NMN Model Overview

The full list of modules used in this module is explained in the Table2.2. Each module takes 0, 1 or 2 attention maps (and also visual and textual features) as input, and outputs either an attention map a out or a score vector y for all possible answers. The operator $\otimes$ is element-wise multiplication, and sum is summing the result over spatial dimensions. The "vec" operation is flattening an attention map into a vector, and adding two extra dimensions: the max and min over attention map.

Module Name	**Att -inputs**	**Output**	**Implementation details**
find	(none)	att	$a_{out} = conv_2(conv_1(x_{vis}) \otimes Wx_{txt})$
relocate	a	att	$a_{out} = conv_2(conv_1(x_{vis}) \otimes W_1 sum(a \otimes x_{vis}) \otimes W_2 x_{txt})$
and	a_1,a_2	att	$a_{out} = minimum(a_1, a_2)$
or	a_1,a_2	att	$a_{out} = maximum(a_1, a_2)$
filter	a	att	$and(a, find[x_{vis}, x_{txt}]())$
[exist,count]	a	ans	$y = W^T vec(a)$
describe	a	ans	$y = W_1^T(W_2 sum(a \otimes x_{vis}) \otimes W_3 x_{txt})$
[eq_count, more,less]	a_1,a_2	ans	$y = W_1^T vec(a_1) + W_2^T vec(a_2)$
compare	a_1, a_2	ans	$y = W_1^T(W_2 sum(a_1 \otimes x_{vis}) \otimes W_3 sum(a_2 \otimes x_{vis}) \otimes W_4 x_{txt})$

TABLE 2.2: Modules used in N2NMN

In the paper[2] the experimental results on SHAPES, CLEVR and VQA dataset demonstrate that N2NMN model is capable of handling complicated reasoning problems, and the end-to-end optimization of the neural modules and layout policy can lead to improvement over behavioral cloning from expert layouts.

Chapter 3

End-to-End Training on NLVR Dataset

3.1 N2NMN model

This chapter contains the implementation of N2NMN model[2] for NLVR dataset. This implementation uses a library named tensorflow_fold, which is discussed in [4],which supports dynamic computation of graphs in python. The N2NMN model on NLVR dataset has been tested under 3 settings.

- **Behavioral Cloning from expert layout:** For this setting expert layout cloning has been used. The learning would be easier given some additional knowledge of module layout. So, in this setting, the ground truth has been provided and the model copies the ground truth layer. In this setting, for getting image feature, a simple CNN with 2 convolution layers has been used and the ground truth was generated using Stanford tagger.
- **Reinforcement learning:** In this setting, with the additional layout information provided earlier, the model can also explored more possible valid layout tokens using reinforcement learning [6]. This setting allows to explore more possible valid layouts which can have more information than the existing ground truth.
- **Training from scratch:** In this setting, no external ground truth has been provided to train the model. Optimizing the loss function from scratch is a challenging reinforcement learning problem: one needs to simultaneously learn the parameters in the sequence-to-sequence RNN to optimize the layout policy and textual attention weights to construct the textual features x_{txt}^{m} for each module, and also the parameters in the neural modules. This is more challenging than a typical reinforcement learning scenario where one only needs to learn a policy. Here, some additional constraints are applied to make sure that the predicted layout is valid.

The accuracy obtained by using these three settings are explained the the table 3.1.

Policy Used	train set	Dev set	test set
Behavioral Cloning from expert layout	94.17%	57.7%	60.15 %
Reinforcement learning	95.9%	57.35%	60.97%
Training from scratch	86.65%	53.4%	56.8%

TABLE 3.1: Accuracy obtained after implementing N2NMN on NLVR dataset

3.2 N2NMN model with VGG image feature

Till the previous experiment setup, for getting image feature, a simple CNN network has been used which does not provide very high level image feature. To get high level image feature, we have used pretrained VGG-16[7] model and trained the model using above 3 settings. But it was found that accuracy has not changed noticeably.

The accuracy obtained by using these three settings are explained the the table 3.2.

Policy Used	train set	Dev set	test set
Behavioral Cloning from expert layout	98.3%	57%	60.8 %
Reinforcement learning	99.7%	56.4%	60.2%
Training from scratch	99.62%	58.57%	58.5%

TABLE 3.2: Accuracy obtained after using VGG net on NLVR dataset

3.3 Changing LSTM dimensions

Changing the number of hidden units in the LSTM changes its ability to learn the textual feature from the given question.

We have experimented by changing number of hidden units in the LSTM for this architecture. The result obtained from such experiment is explained in Table 3.3.

LSTM dimension	Policy Used	train set	Dev set	test set
1000	Behavioral Cloning from expert layout	98.3%	57%	60.8 %
	Reinforcement learning	99.7%	56.4%	60.2%
512	Behavioral Cloning from expert layout	87.6%	60.4%	57%
	Reinforcement learning	90.9%	58.7%	57.8%
256	Behavioral Cloning from expert layout	92.8%	58.7%	57.8 %
	Reinforcement learning	95.2%	57.4%	59.1%

TABLE 3.3: Accuracy obtained after changing the number of hidden units in LSTM

3.4 Freezing CNN weights

Whenever a new input instance comes, the model tries to learn the image feature using the CNN. In this experiment setting, we freeze the weights of the CNN using the weights of the best available result's CNN weight. The result by this setting is explained in the Table 3.4.

LSTM dimension	**Policy Used**	**train set**	**Dev set**	**test set**
512	Behavioral Cloning from expert layout	89.4%	56.7%	56.79%
	Reinforcement learning	90.4%	57.3%	56.96%
256	Behavioral Cloning from expert layout	94.2%	56.7%	57.1 %
	Reinforcement learning	95.3%	56%	57.3%
128	Behavioral Cloning from expert layout	92.9%	55.3%	57.49 %
	Reinforcement learning	94.37%	56%	57.3%

TABLE 3.4: Accuracy obtained after freezing the CNN weights

Chapter 4

Visualization of the model on different datasets

In this chapter, we would discuss about the visualization of the correspondence of the image feature and textual feature for a particular module. The visualization is needed to understand the attention of the modules in the words of the sentence and image. We have visualized two datasets: SHAPES and NLVR.

4.1 SHAPES dataset

SHAPES dataset produces best results after using N2NMN architecture. An example of the visualization of this dataset with the question "is a circle below below a red shape" is explained in Figure 4.1. In this Figure, the first sub-image corresponds to the image and the question from the SHAPES dataset. Right to that, predicted layout and original and predicted label is shown. At the rightmost, textual attention for each module used in the predicted layout is shown. Next, there are the attention map from each of the modules used in the predicted layout.

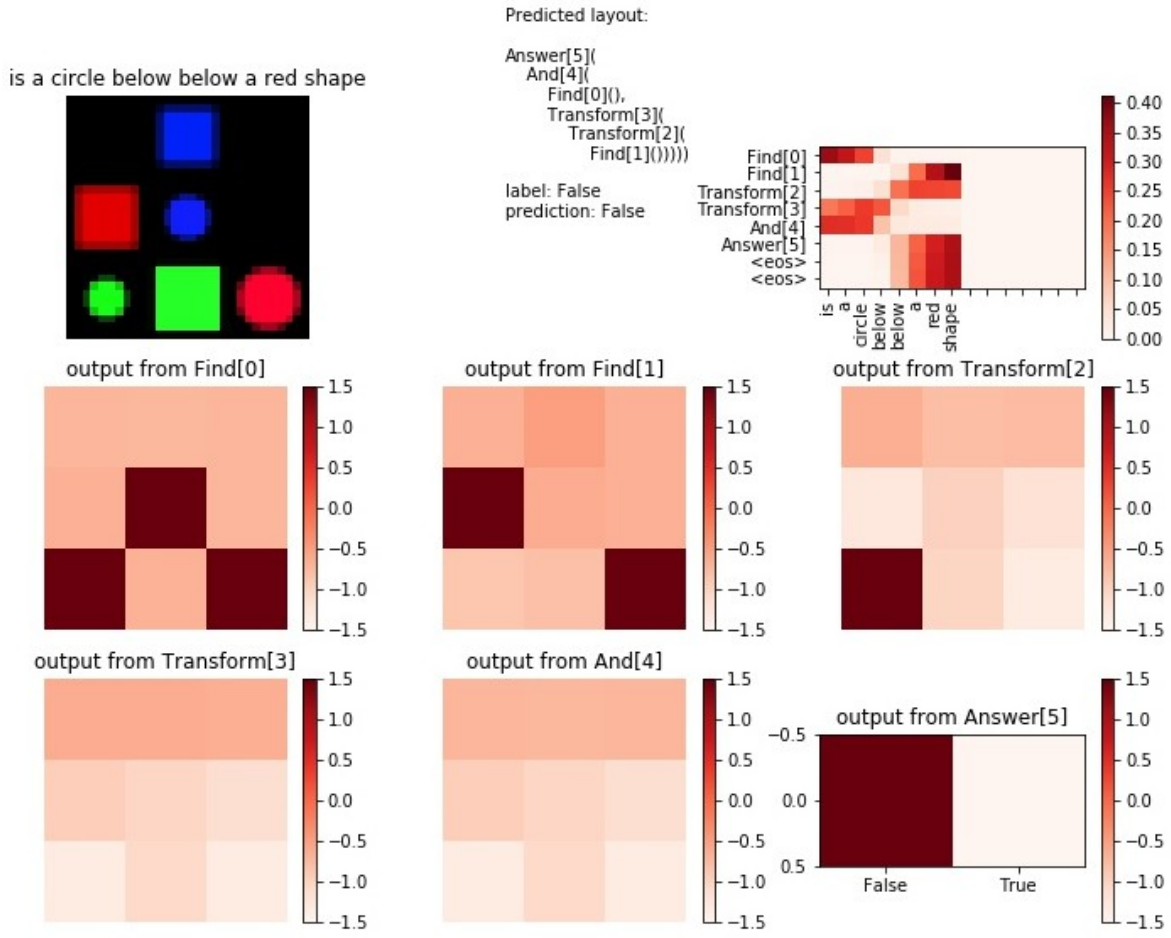


FIGURE 4.1: Visualization of one particular example from SHAPES dataset

4.2 NLVR dataset

An example of the visualization of NLVR dataset with the question "there is 1 box with exactly 2 items" is explained in Figure 4.2. In this Figure, the first sub-image corresponds to the image and the question from the NLVR dataset. Right to that, predicted layout and original and predicted label is shown. At the rightmost, textual attention for each module used in the predicted layout is shown. Next, there are the attention map from each of the modules used in the predicted layout. From the figure, it can be seen that the modules do not have proper textual attention or image attention.

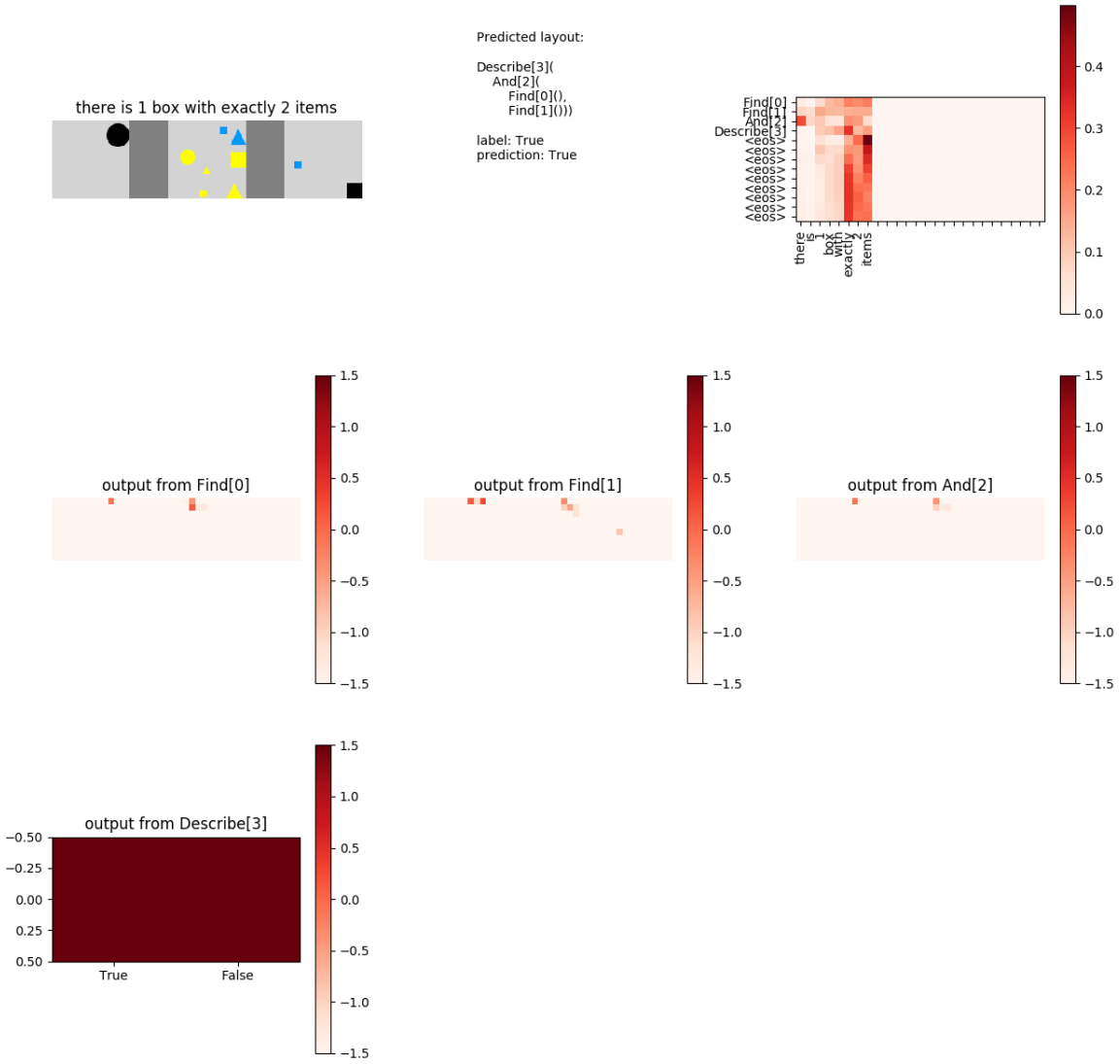


FIGURE 4.2: Visualization of one particular example from NLVR dataset

4.3 Observations

As we can see from the previous sections that in SHAPES dataset, the attention on the image for different modules is better than that of NLVR dataset. So, from that we came to the conclusion that in NLVR dataset, even though the model is not learning much from the modules, it is predicting right answers randomly. This observation becomes the motivation for the next chapter where we have discussed the modifications in the modules.

Chapter 5

Proposed Modification in N2NMN Model

5.1 New proposed Modules

From the previous chapter, we can see that the existing modules are unable to capture proper textual as well as image feature because of the complexity of the dataset. So, in this chapter, we have proposed some modification in the existing architecture to better capture the textual feature from the given question. The modifications consisting of inclusion of new modules in the architecture. The new modules are described in the Table 5.1. Here x_{txt} corresponds to the textual feature, x_{vis} corresponds to the visual feature, "att" corresponds to the attention map, "att_summary" corresponds to the attention summary which consists of minimum, average and maximum of an attention, "vector" corresponds to the vectors and "ans" corresponds to the answer score.

Module Name	**Input**	**Output**	**Description**
Break	None	None	Sets a flag true, which indicates the next modules to break the image feature into three parts (one part corresponding to each box).
Find	x_{txt}, x_{vis}	att	If the flag is set as true, then it finds the attention map on individual box, otherwise it finds the attention map over the full image.
Transform	x_{txt}, x_{vis}, att	att	If the flag is true, it finds the re-attention map on the given attention map of individual box or it finds the re-attention on the given attention map over the full image.
Describe	att	ans	It finds the answer score from the given attention map.
And	att, att	att	Performs the AND operation on two attention maps.
Or	att, att	att	Performs the OR operation on the given two attention maps.
Not	att	att	Finds the complement attention map of the given attention map.
Count	att	vector	If the flag is true, it counts the number of individual attention in the attention map of one box, else it counts the number of individual attention from the attention map of the whole image.
Compare	x_{txt}, att	vector	If the flag is true, it compares the vector output of one box from count with the textual feature, else it compares the vector output of the full image from count with the textual feature.
Find_SameProperty	att, att	vector	Finds if the two attention has same property or not.
CompareAtt	att, att	ans	If the flag is true, measures the similarity between two input attention within a box, else measures the similarity between two input attention over the full image and return the score of truth values accordingly.
CompareReduce	vector, vector	vector	If the flag is true, measures the similarity between two input vectors within a box, else measures the similarity between two input vectors over the full image.

Module Name	**Input**	**Output**	**Description**
AttReduce	att	att_summary	Finds the minimum, average and maximum of the given attention.
Combine	3 vectors, x_{txt}	ans	With the help of the textual feature, it decides how many vectors fulfill the condition and calculate scores of the truth values according to that.
ExistAtt	3 att_summary, x_{txt}	ans	With the help of the textual feature, it decides how many att_summary fulfill the condition and calculate scores of the truth values according to that.
Exist	vector	ans	Calculates the final scores corresponding to the truth values.

TABLE 5.1: Proposed Modified Modules

5.2 Preparing new ground truth layout

Addition of new modules in the architecture leads to the preparation of new ground truth layout for each questions. For this purpose, we have replaced each key word in one sentence with some tokens, which leads to new ground truth layout. The replacement of each key word with token words is done as explained in the paper[8]. Unique token sequence is generated by taking only the token words from the sentenced replaced with token words. For each unique token sequence, we have generated the ground truth manually. Using this policy, we have manually generated the ground truth of 212 unique token sequence, which covers almost 85% of the total training data. For the rest, we have kept the same ground truth which has been used previously. The procedure from replacing the key words with token words to generate the ground truth is explained with some examples in the Table 5.2 .

Sentence	**Replaced Sentence**	**Token Sequence**	**Ground truth layout**
there are 4 yellow items	there are T_INT T_COLOR item	[‘T_INT’, ‘T_COLOR’]	[‘_Find’, ‘_Count’, ‘_Compare’, ‘_Exist’]
there are 2 boxes that have at least 1 black circle	there are T_INT box that have T_QUANTITY _COMPARE T_ONE T_COLOR T_SHAPE	[‘T_INT’ ,‘T_QUANTITY _COMPARE’, ‘T_ONE’, ‘T_COLOR’, ‘T_SHAPE’]	[‘_Break’, ‘_Find’, ‘_Count’, ‘_Compare’, ‘_Combine’]
there are exactly two towers with a yellow block at the top	there are T_QUANTITY _COMPARE T_INT tower with a T_COLOR block at the T_LOC	[‘T_QUANTITY _COMPARE’, ‘T_INT’, ‘T_COLOR’, ‘T_LOC’]	[‘_Break’, ‘_Find’, ‘_Find’, ‘_And’, ‘_AttReduce’, ‘_ExistAtt’]
there is 1 box with exactly 2 items of the same color	there is T_ONE box with T_QUANTITY _COMPARE T_INT item of the T_SAME color	[‘T_ONE’, ‘T_QUANTITY _COMPARE’, ‘T_INT’, ‘T_SAME’]	[‘_Break’, ‘_Find’, ‘_Find’, ‘_Find_SameProperty’, ‘_Find’, ‘_Count’, ‘_Compare’, ‘_CompareReduce’, ‘_Combine’]

Table 5.2: Generation of ground truth from question sentences

5.3 Results using the new proposed model

Using this new proposed model, we have tested the NLVR dataset using behavioral cloning from expert layout with different LSTM dimensions. The result of this experiment is explained in Table 5.3.

LSTM dimension	**Policy used**	**Train**	**Dev**	**Test**
128	Behavioral Cloning from expert layout	95.6%	58.52%	60.32%
256	Behavioral Cloning from expert layout	95.9%	57.93%	59.61%
512	Behavioral Cloning from expert layout	96.13%	57.6%	57.18%
1000	Behavioral Cloning from expert layout	94.89%	59.13%	58.3%

Table 5.3: Results from new proposed model

Chapter 6

Conclusion and Future Work

In this thesis, we looked into the implementation of N2NMN model for NLVR task. We have proposed modifications in the modules and layouts to better capture the correspondence between textual and image feature. The final result shows that the proposed model gives comparable result with the state-of-art architecture.

There is a lot of scope for the improvement of the proposed model on NLVR dataset.

- As the number of modules has increased, the training of the model from scratch and the training using reinforcement learning needs to have some more constraints in addition to the existing ones.
- We can use deep architecture e.g. ResNet-50[9] to generate image features.

Bibliography

[1] Alane Suhr, Mike Lewis, James Yeh, and Yoav Artzi. A corpus of natural language for visual reasoning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 217–223, 2017.

[2] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. *CoRR, abs/1704.05526*, 3, 2017.

[3] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C Lawrence Zitnick, Devi Parikh, and Dhruv Batra. Vqa: Visual question answering. *International Journal of Computer Vision*, 123(1):4–31, 2017.

[4] Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question. In *Advances in neural information processing systems*, pages 2296–2304, 2015.

[5] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48, 2016.

[6] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[8] Omer Goldman, Veronica Latcinnik, Udi Naveh, Amir Globerson, and Jonathan Berant. Weakly-supervised semantic parsing with abstract examples. *arXiv preprint arXiv:1711.05240*, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.