2021-2
Introduction to Deep Learning

Deep Modular Co-Attention Networks for Visual Question Answering

Group10: ShinSeonggu NamYeonju

CONTENTS

1. What is VQA?

2. What is MCAN?

3. CODE

4. EVALUATION

5. CONCLUSION

#1

What is VQA?

Visual
Question
Answering

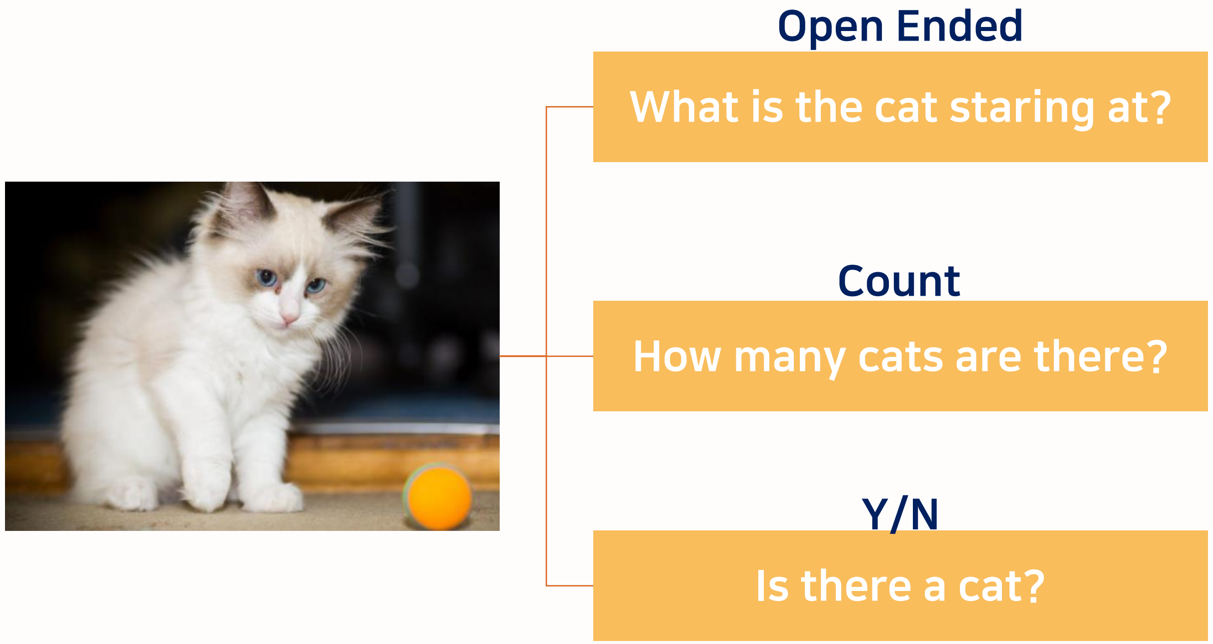
01: What is VQA?

Introduction

VQA

Visual question answering

1. Understand the Question: **Key Words**



Open Ended

What is the cat staring at?

Count

How many cats are there?

Y/N

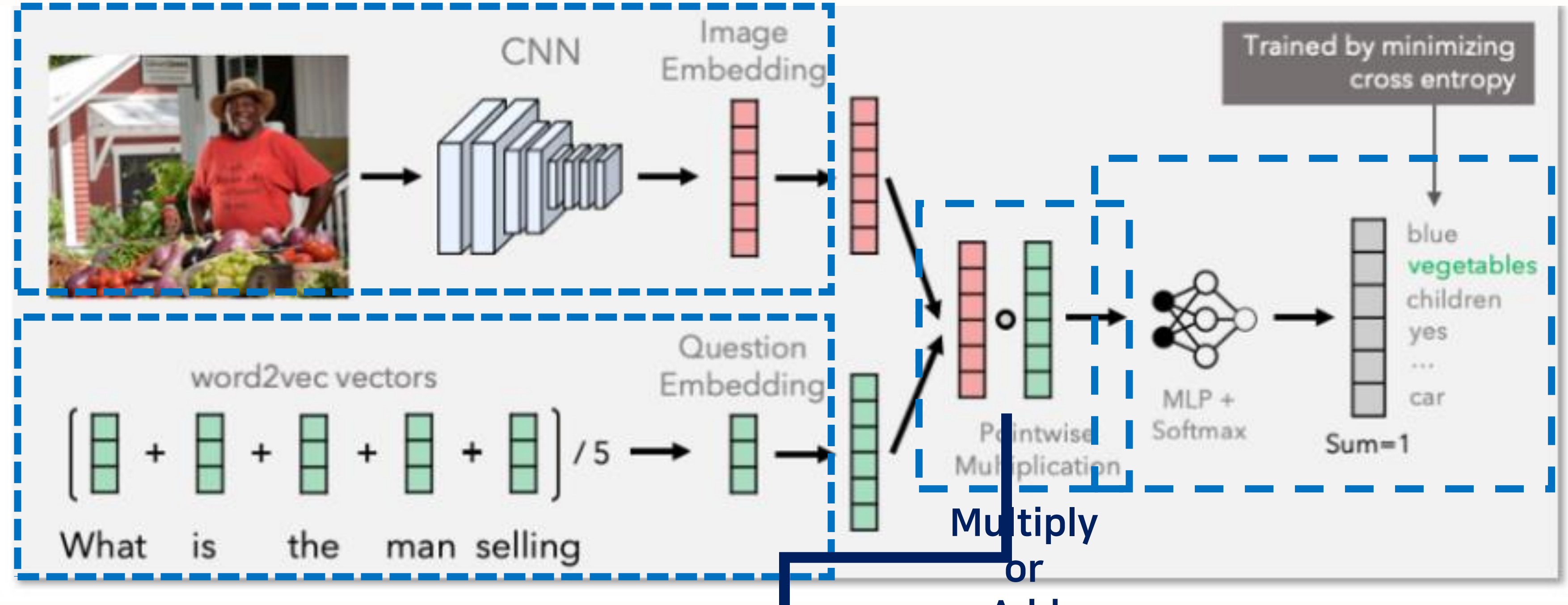
Is there a cat?

2. Understand the Image: **Key Objects**

01: What is VQA?

Introduction

CV



Open Ended

NLP

Multiply or Add

Co-Attention

To understand / extract features much better

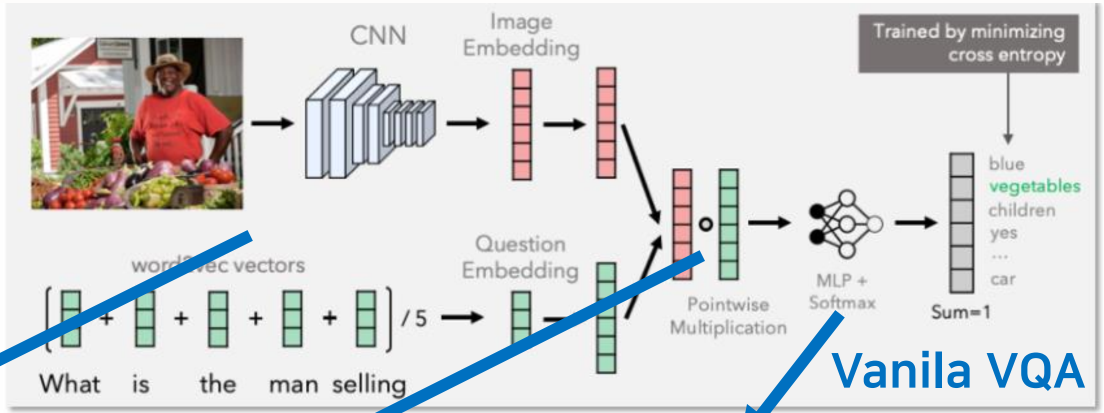
#2

What is MCAN?

deep Modular
Co-Attention
Network

02: What is MCAN?

MCAN Network Overview

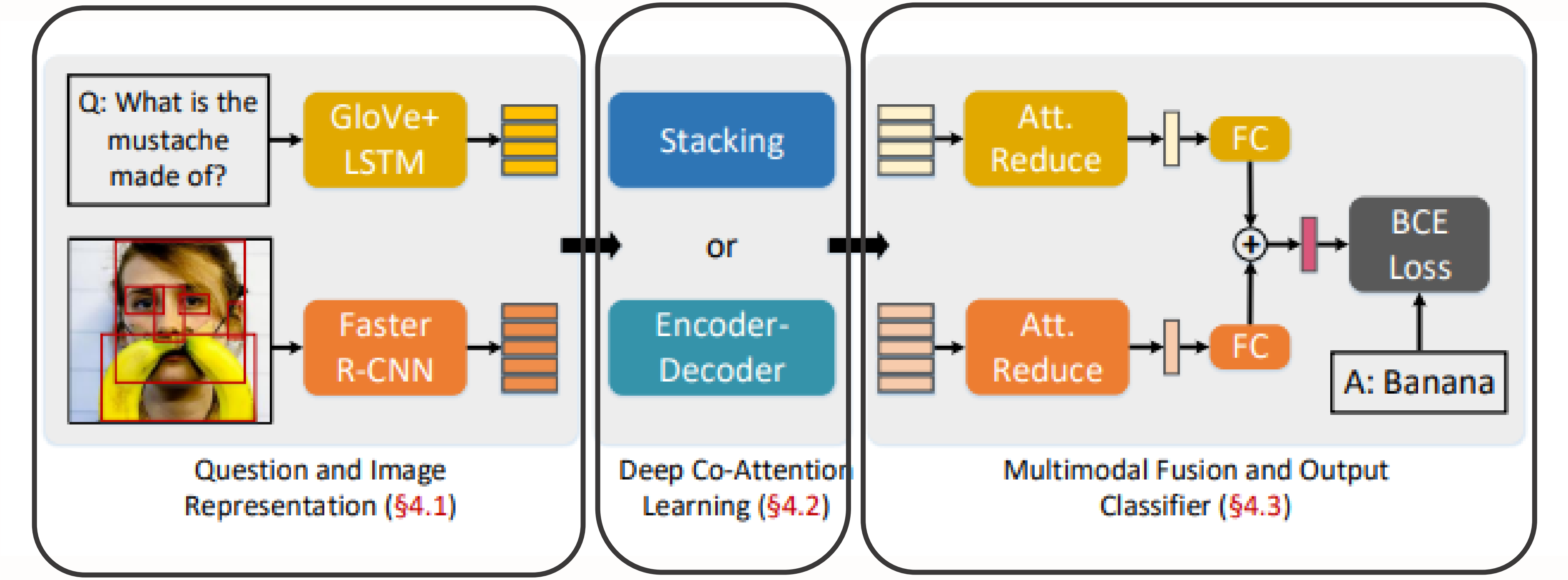


Vanila VQA

Step 1

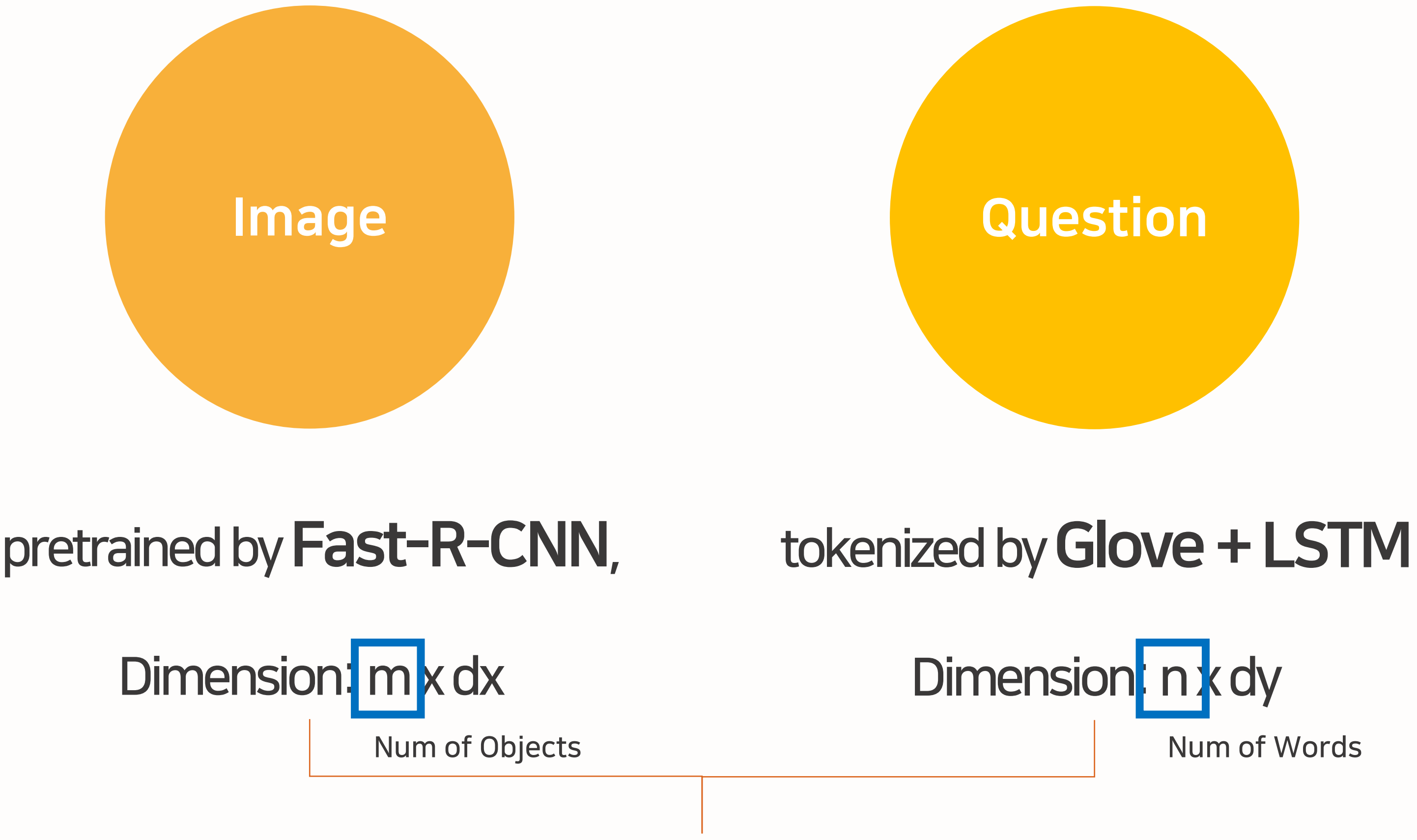
Step 2

Step 3


Q: What is the mustache made of?
GloVe+ LSTM
Faster R-CNN
Stacking
or
Encoder-Decoder
Att. Reduce
FC
Att. Reduce
FC
BCE Loss
A: Banana
Question and Image Representation (§4.1)
Deep Co-Attention Learning (§4.2)
Multimodal Fusion and Output Classifier (§4.3)

02: What is MCAN?

Step 1: Question & Image Representation



Image

Question

pretrained by **Fast-R-CNN,**

tokenized by **Glove + LSTM**

Dimension: m x dx

Num of Objects

Dimension: n x dy

Num of Words

zero-padding to deal with the variable number and length.

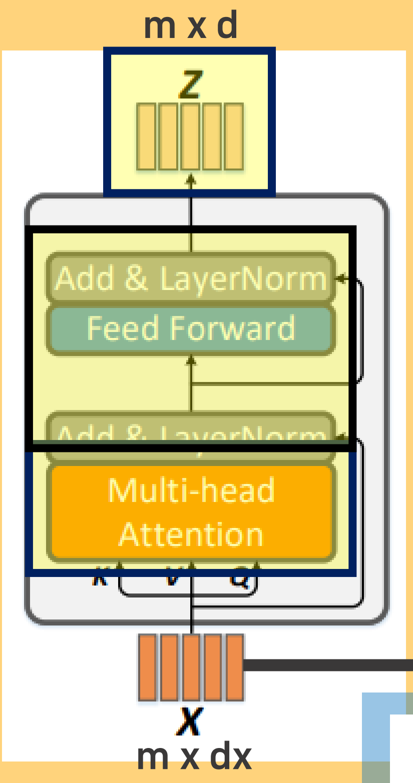
02: What is MCAN?

Step 2: Co-Attention(1) - Attention Units

Combine SA & GA

-> MCA Layer!

MCA Layer Components

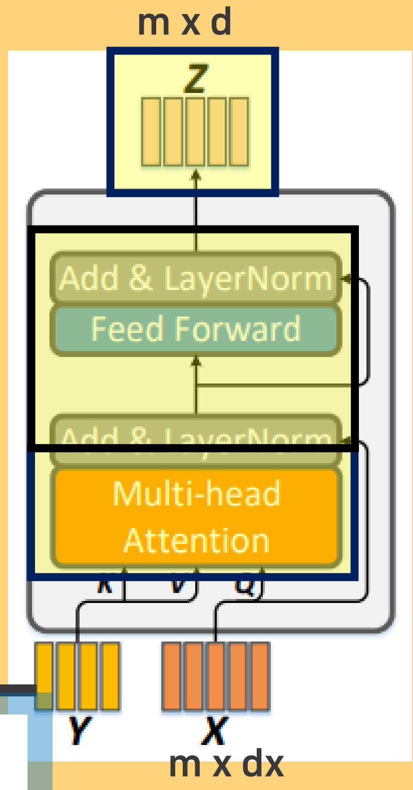


SA (Self-Attention)

- Input: X
- Q, K, V: from X

GA (Guided-Attention)

- Input: X(image)
- Input: Y(question)
- Q: from X
- K, V: from Y



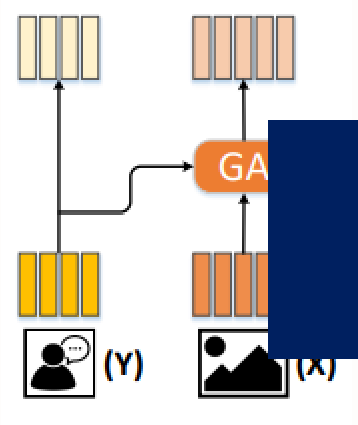
$$f = MA(q, K, V) = [\text{head}_1, \text{head}_2, ..., \text{head}_h]W^o$$

Combination of j attentions

$$\text{head}_j = A(qW_j^Q, KW_j^K, VW_j^V)$$

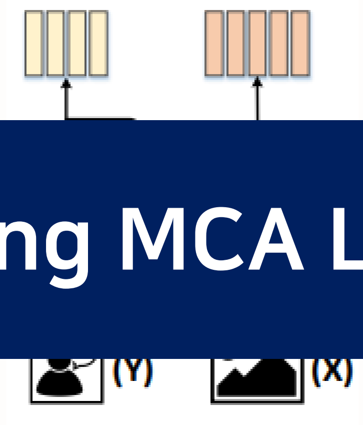
Step 2: Co-Attention(2) – MCA Layers

3 types of MCA Layers



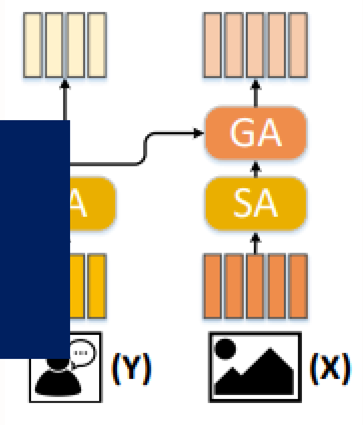
ID(Y)-GA(X, Y)

Only 1 GA for Image



SA(Y)-GA(X, Y)

SA for Question
+
GA for Image



SA(Y)-SGA(X, Y)

SA for Question
+
SA for Image
+
GA for Image

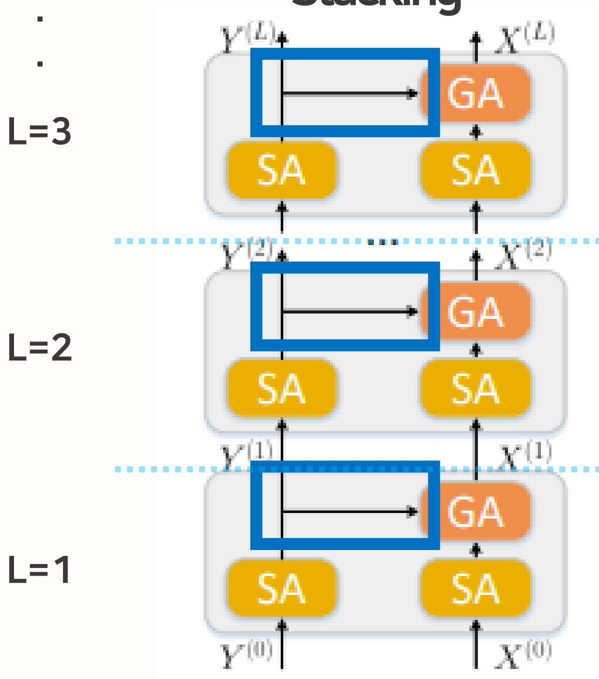
Stacking MCA Layers

02: What is MCAN?

Step 2: Co-Attention(3) – MCA Model

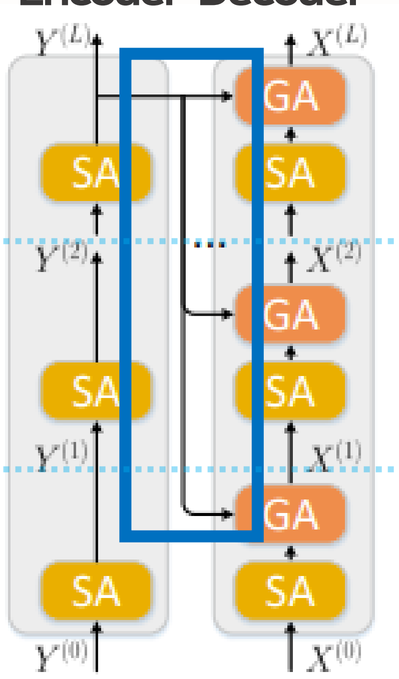
Stacking VS Encoder-Decoder

Stacking



Applying SA(Y) to SGA(X, Y)
per each layer

Encoder-Decoder



Applying the final SA(Y) to SGA(X, Y)
of each layer

$$[X^{(l)}, Y^{(l)}] = MCA^{(l)}([X^{(l-1)}, Y^{(l-1)}])$$

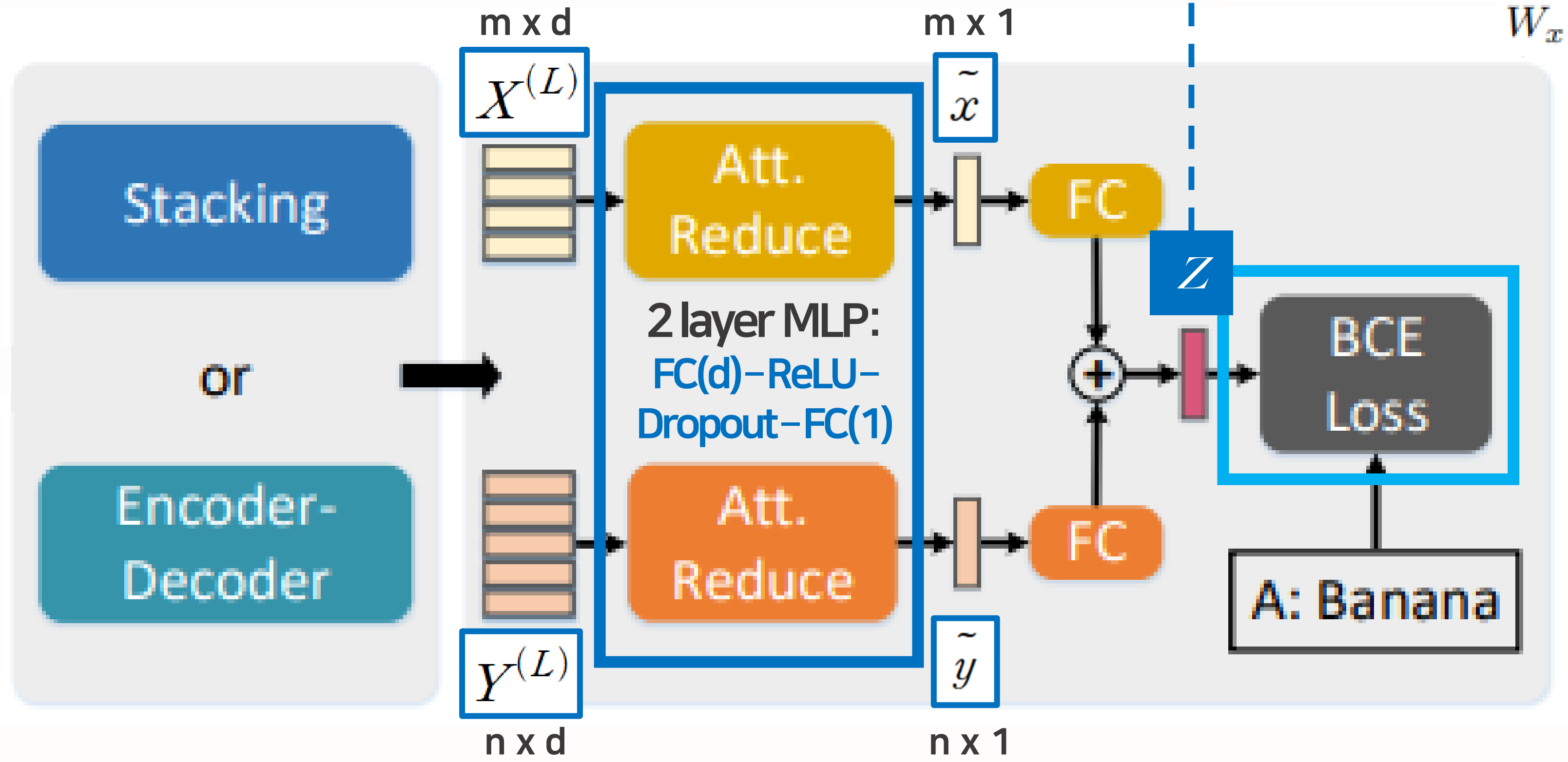
02: What is MCAN?

Step 3: Fusion & Classification

FUSION

$z = \mathrm{LayerNorm}(W_x^T \tilde{x} + W_y^T \tilde{y})$

$W_x, W_y \in \mathbb{R}^{d \times d_z}$



#3

Code Execution

Using DEMO

DataSet

Image

COCO
Common Objects in Context

Pretrained by F-R-CNN

Question

VQA

Human noted ans-ques pairs
(Related to COCO images)

Pretrained by LSTM+GLOVE

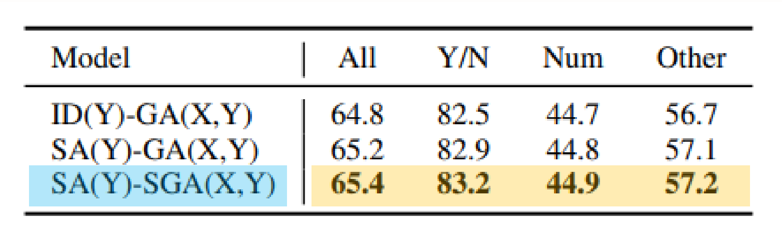
Train	Val	Test
80k / 444k	40k / 214k	80k / 448k

#4

Results

04: Results

Performance Test and Analysis

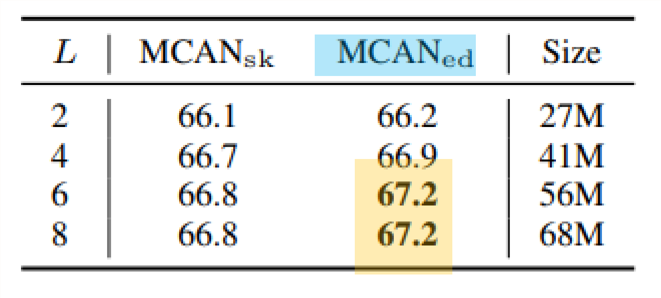


Model	All	Y/N	Num	Other
ID(Y)-GA(X,Y)	64.8	82.5	44.7	56.7
SA(Y)-GA(X,Y)	65.2	82.9	44.8	57.1
SA(Y)-SGA(X,Y)	**65.4**	**83.2**	**44.9**	**57.2**

MCA Varients

Best model :

SA(Y)-SGA(X, Y)



L	MCAN$_{sk}$	MCAN$_{ed}$	Size
2	66.1	66.2	27M
4	66.7	66.9	41M
6	66.8	**67.2**	56M
8	66.8	**67.2**	68M

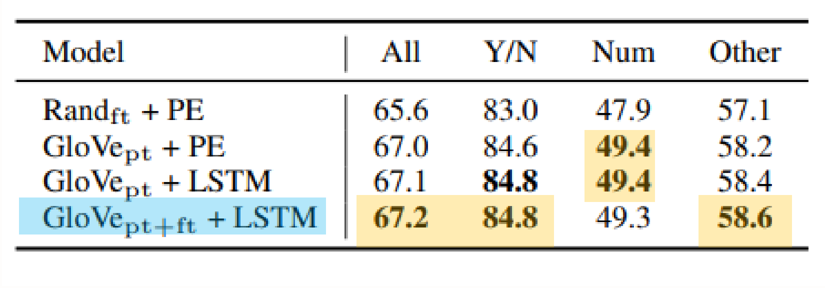
Stacking vs Encoder-decoder

Under 6 layers:

similar performance

Over 6 layers:

en/decoder shows better performance

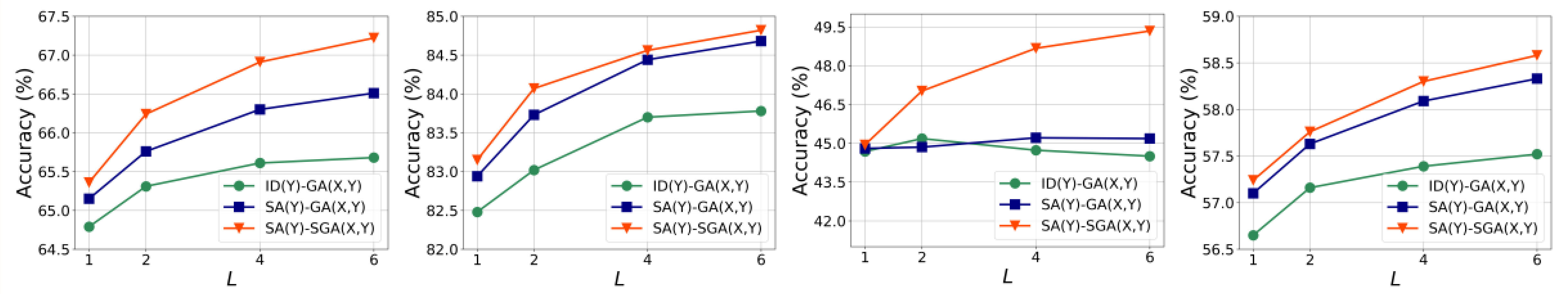


Model	All	Y/N	Num	Other
Rand$_{ft}$ + PE	65.6	83.0	47.9	57.1
GloVe$_{pt}$ + PE	67.0	84.6	**49.4**	58.2
GloVe$_{pt}$ + LSTM	67.1	**84.8**	**49.4**	58.4
GloVe$_{pt+ft}$ + LSTM	**67.2**	**84.8**	49.3	**58.6**

Question Representations

Performance depending on whether the model used Glove, PE, or LSTM

Performance Test and Analysis



(a) All (b) Y/N (c) Num (d) Other

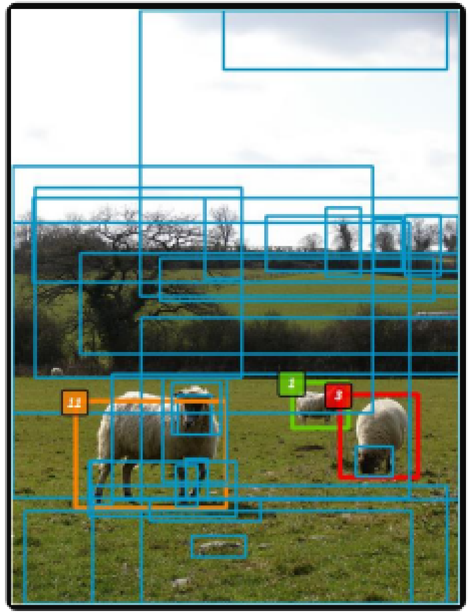
Overall and per-type accuracies of MCAN models

Best model : **SA(Y)-SGA(X, Y)**

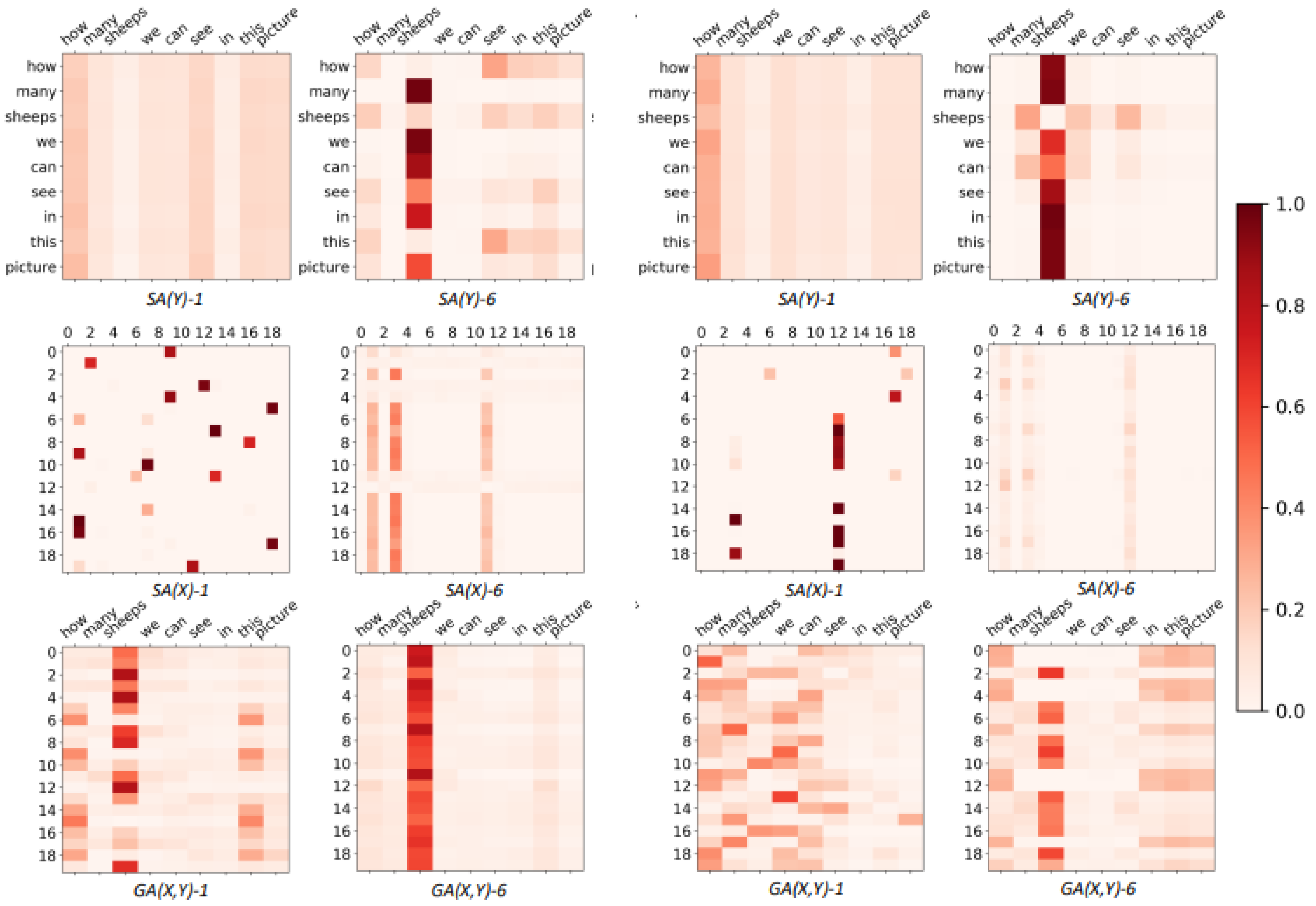
※ID(Y)-GA(X, Y), SA(Y)-GA(X, Y) in number questions
: SA(X) is important for increasing the performance of number questions

04: Results

Performance Test and Analysis



Q: *How many sheep we can see in this picture ?*
A: *3*



(a) Encoder-Decoder (**P**: 3)

(b) Stacking (**P**: 3)

04: Results

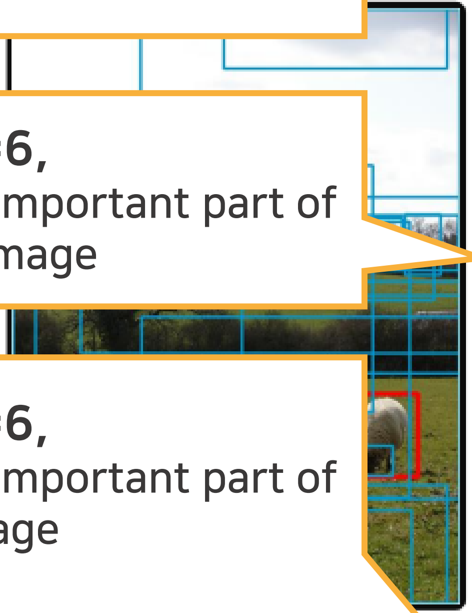
Performance Test and Analysis

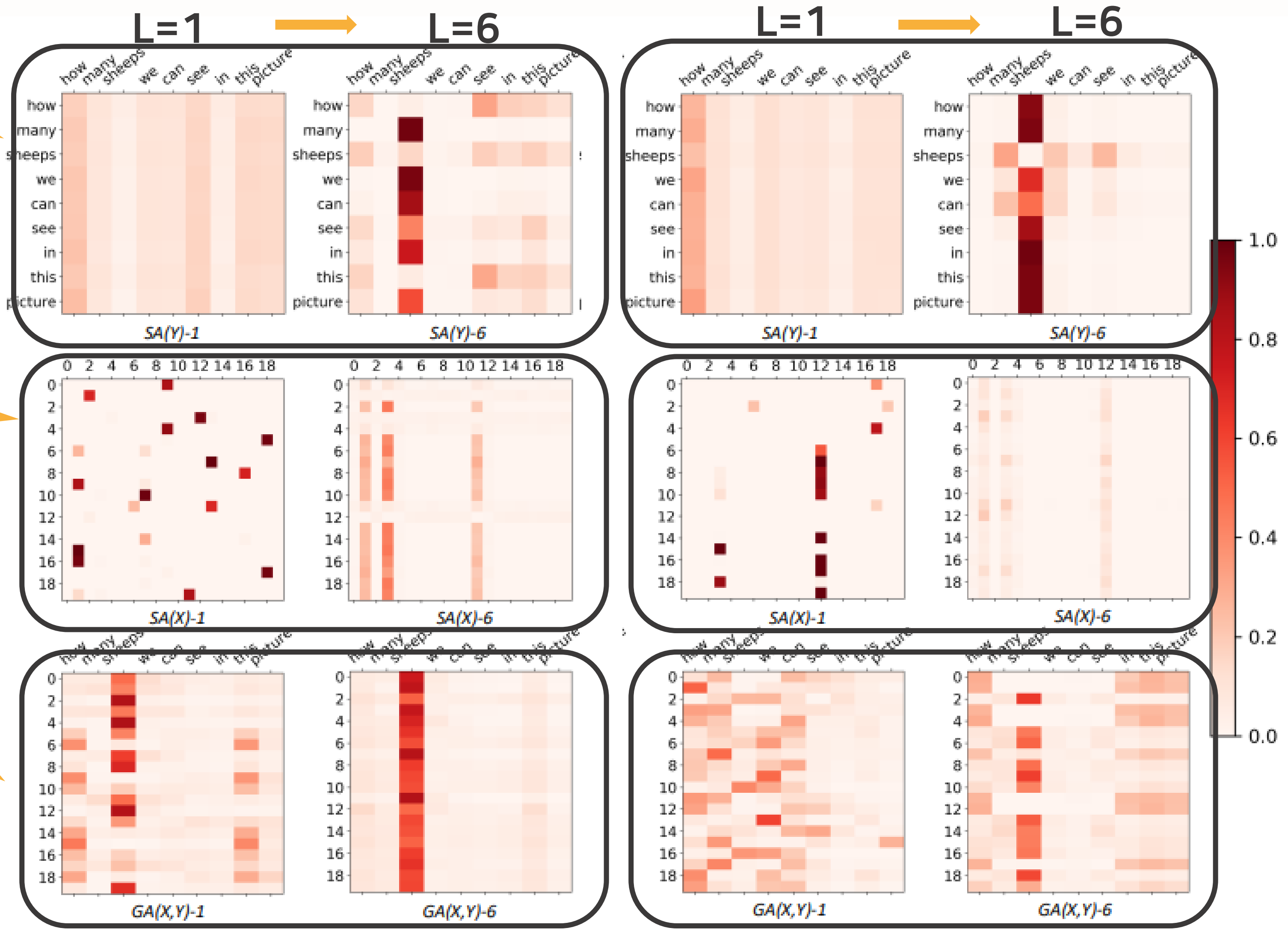
L=6,
Focusing on the important part of the question

L=6,
Focusing on the important part of the image

L=6,
Focusing on the important part of Image

By being guided from question





(a) Encoder-Decoder (**P**: 3)   (b) Stacking (**P**: 3)

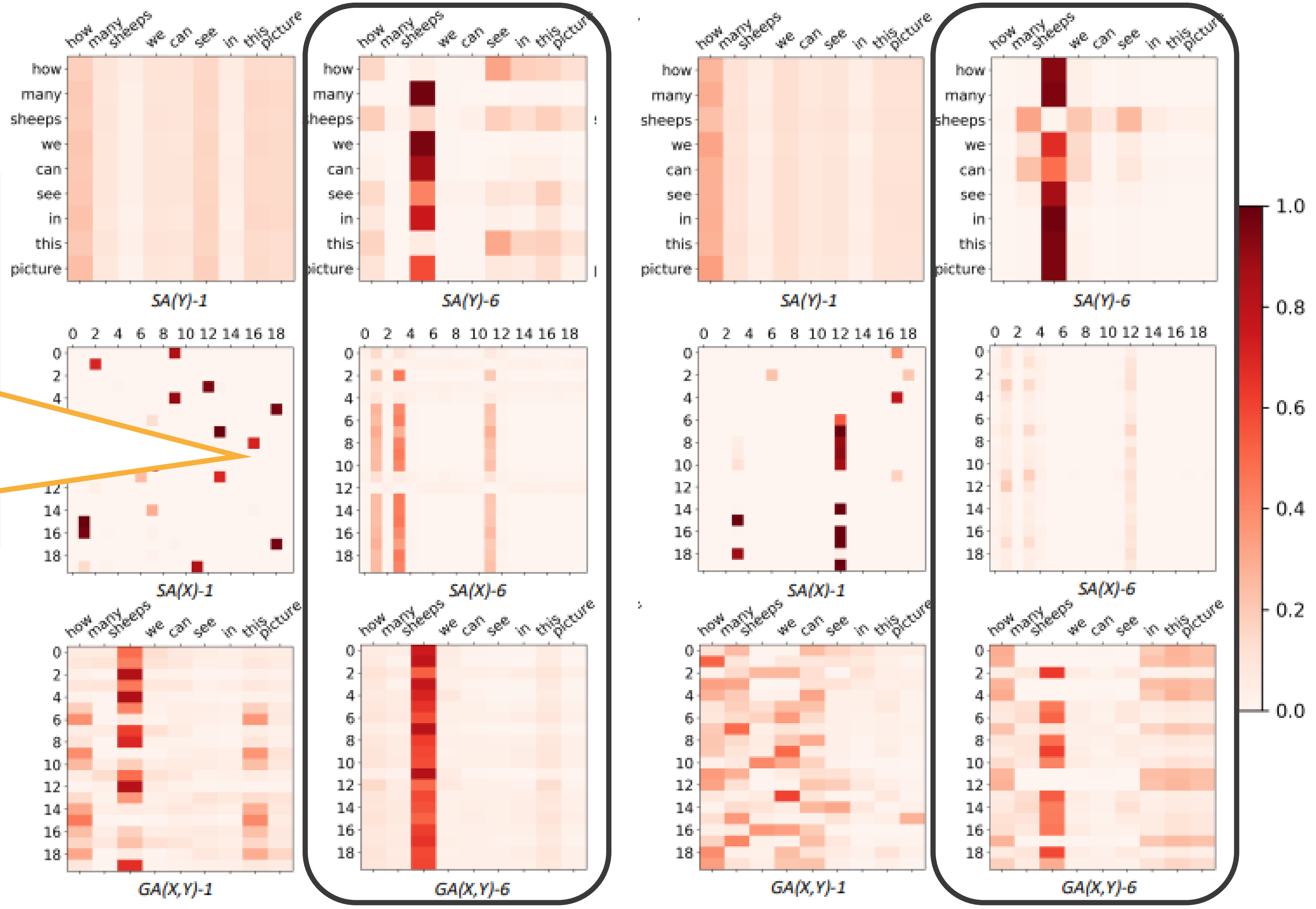
=> The layer should be deep enough(=6) for better performance

04: Results

Performance Test and Analysis

L=6,
Encoder-Decoder shows better performance than stacking in SA(Y), SA(X), GA(X, Y)

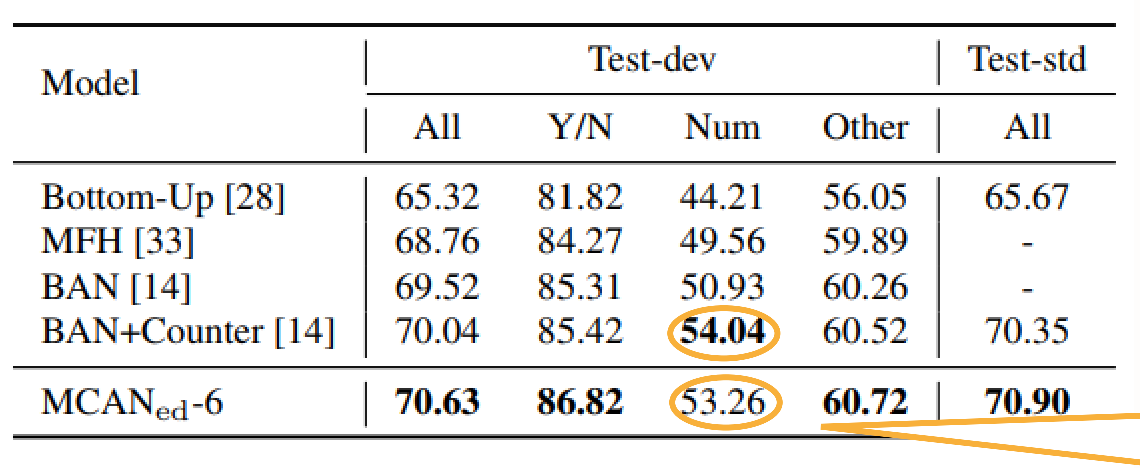
⇒ Encoder-Decoder has better performance than stacking
as the layer become deep enough(=6)



(a) Encoder-Decoder (**P**: 3)

(b) Stacking (**P**: 3)

Comparison with state-of-the-art



Model	Test-dev				Test-std
	All	Y/N	Num	Other	All
Bottom-Up [28]	65.32	81.82	44.21	56.05	65.67
MFH [33]	68.76	84.27	49.56	59.89	-
BAN [14]	69.52	85.31	50.93	60.26	-
BAN+Counter [14]	70.04	85.42	**54.04**	60.52	70.35
MCAN_{ed}-6	**70.63**	**86.82**	53.26	**60.72**	**70.90**

MCAN: the best model compared with the current state-of-the-art models

Just little bit lower in 'object counting performance'

#5

Conclusions

With Limitations

05: Conclusion

Conclusion

[Modular Co-Attention Network(MCAN) presented for VQA is **effective**.]

[MCAN consists of a **cascade of MCA layers**,
each of which consists of **SA** and **GA** units.]

[Using the **encoder-decoder** ,
and cascading MCA layers in depth(>=6) makes better performance for VQA.]

05: Conclusion

Limitations

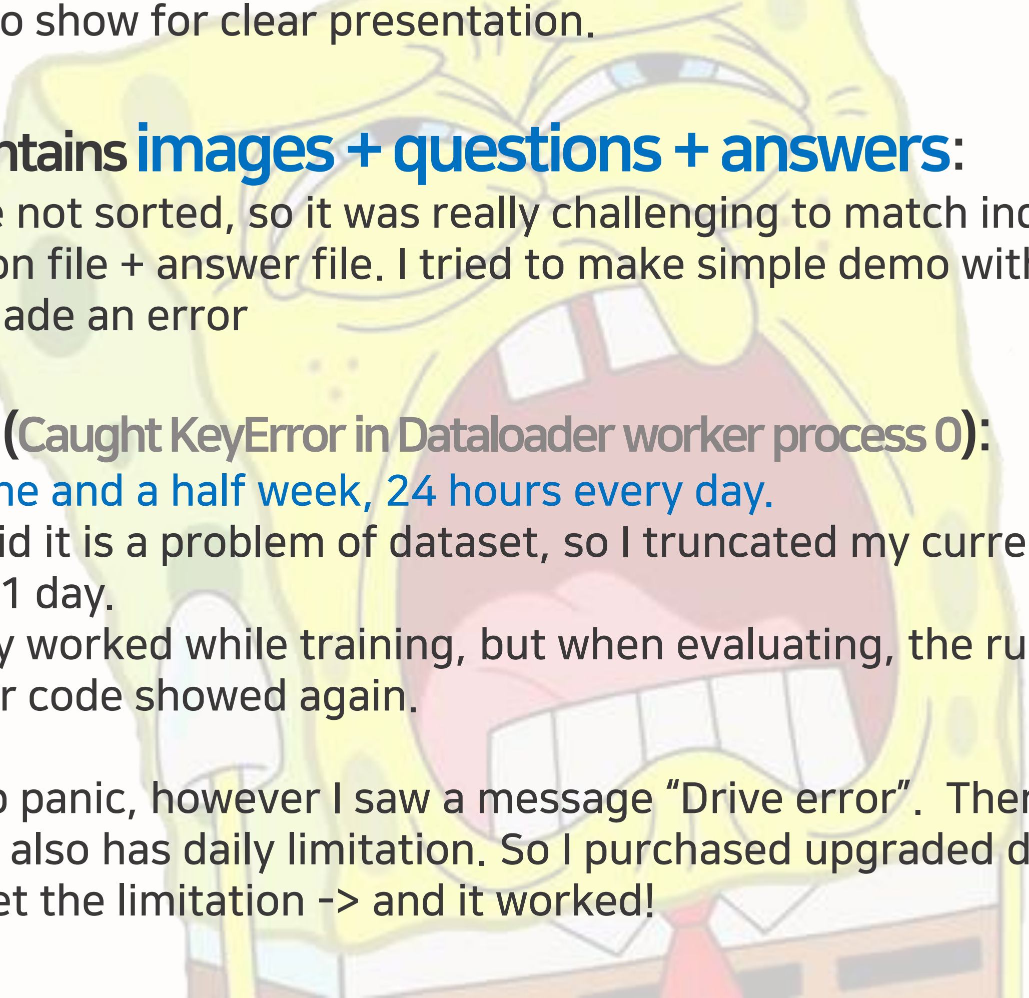


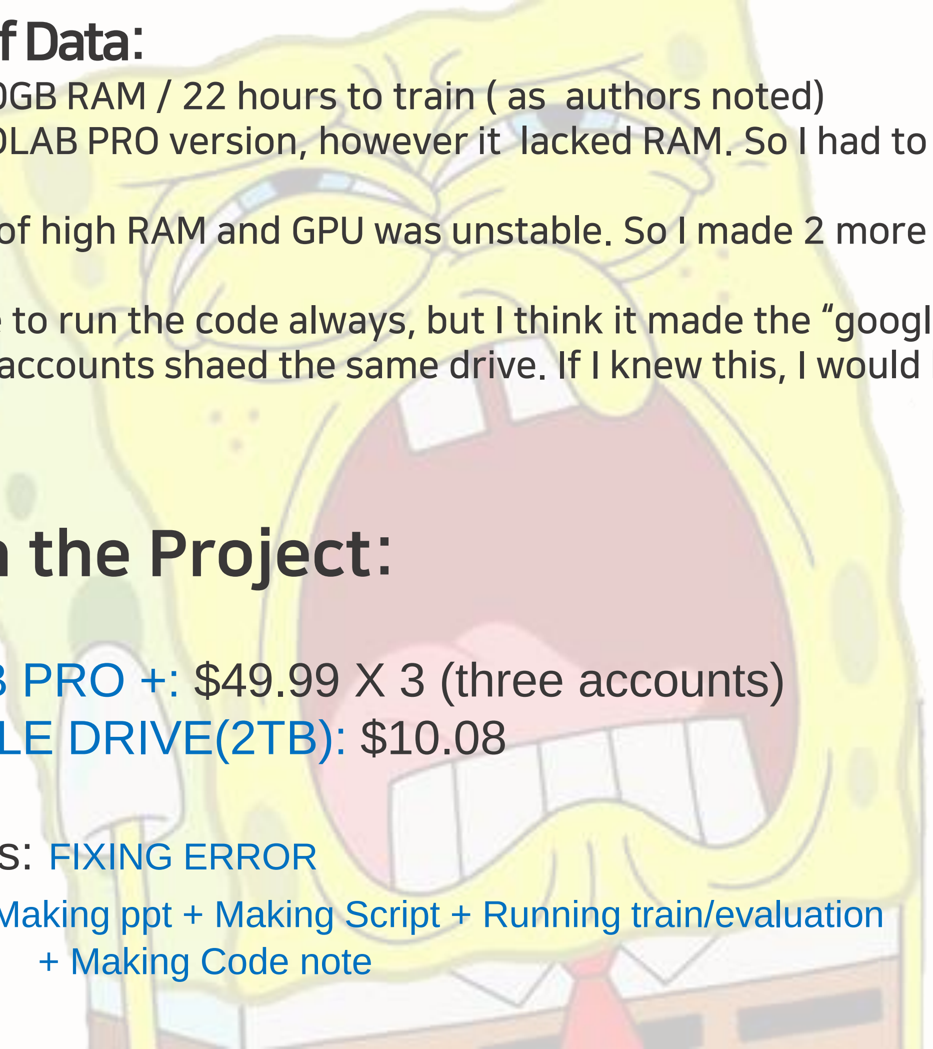
Image data is provided as pre-trained features:
It was hard to show for clear presentation.

Dataset contains images + questions + answers:
Datasets are not sorted, so it was really challenging to match indexes between image file + question file + answer file. I tried to make simple demo with small dataset, but I failed…. It made an error

Code Error (Caught KeyError in Dataloader worker process 0):
Spent over one and a half week, 24 hours every day.
Someone said it is a problem of dataset, so I truncated my current project and restarted in 1 day.
It fortunately worked while training, but when evaluating, the runtime shut down and the error code showed again.

I was in deep panic, however I saw a message "Drive error". Then I found out google drive also has daily limitation. So I purchased upgraded drive version in order to reset the limitation -> and it worked!

Limitations



Huge amount of Data:

It requires at leat 30GB RAM / 22 hours to train (as authors noted)
I first purchased COLAB PRO version, however it lacked RAM. So I had to purchase COLAB PRO + version.
But the connection of high RAM and GPU was unstable. So I made 2 more accounts with COLAB PRO +.
It made me possible to run the code always, but I think it made the "google drive error" because all these 3 accounts shaed the same drive. If I knew this, I would have used just my original account…

Cost to Run the Project:

MONEY)

COLAB PRO +: $49.99 X 3 (three accounts)
GOOGLE DRIVE(2TB): $10.08

TIME)

2 weeks: FIXING ERROR
1 day: Making ppt + Making Script + Running train/evaluation + Making Code note

Thank you