Conditional Kronecker Batch Normalization for Compositional Reasoning

Cheng Shi[12]
shic17@mails.tsinghua.edu.cn
Chun Yuan[2]
yuanc@sz.tsinghua.edu.cn
Jiayin Cai[3]
cai_jiayin@stu.xjtu.edu.cn
Zhuobin Zheng[12]
zhengzb16@mails.tsinghua.edu.cn
Yangyang Cheng[12]
cheng-yy13@mails.tsinghua.edu.cn
Zhihui Lin[12]
lin-zh14@mails.tsinghua.edu.cn

[1] Department of Computer Science
and Technologies
Tsinghua University
[2] Graduate School at Shenzhen,
Tsinghua University
[3] Xi'an Jiaotong University
Xi'an, China

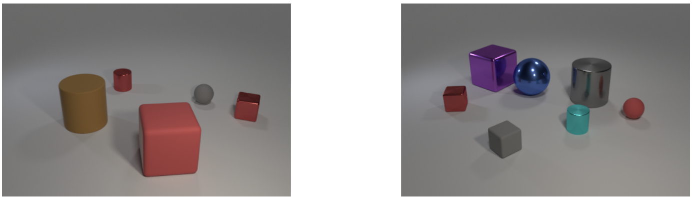
Abstract

Conditional Batch Normalization (CBN) has proved to be an effective tool for visual question answering. However, previous CBN approaches fuse the linguistic information into image features via a simple affine transformation, thus they have struggled on compositional reasoning and object counting in images. In this paper, we propose a novel CBN method using the Kronecker transformation, termed as Conditional Kronecker Batch Normalization (CKBN). CKBN layer facilitates the explicit and expressive learning of compositional reasoning and robust counting in original images. Besides, we demonstrate that the Kronecker transformation in CKBN layer is a generalization of the affine transformation in prior CBN approaches. It could accelerate the fusion of visual and linguistic information, and thus the convergence of overall model. Experiment results show that our model significantly outperforms previous CBN methods (*e.g.* FiLM) in compositional reasoning, counting as well as the convergence speed on CLEVR dataset.

1 Introduction

Visual question answering (VQA)[4] is a challenging multi-modal task that requires responding to natural language questions about images. The long-standing goal of VQA task is to design systems that can reason about the visual world like humans[16], which is central to generally intelligent behavior. However, the first generation of successful VQA models[6, 7, 12] only acquire a superficial cognition of images and questions but achieve high accuracy owing to the biased datasets. For example, a statistical learner may correctly answer the question "What's the weather like in the picture?" not because it understands the scene but because

© 2018. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.



(a) **Q:** What number of red shiny cubes are to the right of the thing that is to the left of the red metal object that is behind the gray matte sphere ? **A: 1**

(b) **Q:** Are there the same number of balls in front of the tiny cyan metallic cylinder and small gray rubber objects behind the tiny matte block ? **A: yes**

Figure 1: Two illustrative examples from the CLEVR dataset of visual reasoning.

biased datasets often ask questions about the weather when it is a rainy day[1, 25]. These statistical learning models have neither a deep understanding of images and questions nor strong reasoning process that would lead to the correct answer.

To this end, the CLEVR[13] dataset was proposed to enable detailed analysis of visual reasoning. CLEVR test the visual reasoning ability via complex question answering, as shown in Figure 1. More importantly, the information in each CLEVR image is complete and exclusive so that the chance of correctly answering question may not be increased by the external information. On the contrary, the models exploiting dataset biases may perform worse on CLEVR. Tests on CLEVR show that most traditional deep learning approaches[6, 7, 12, 13] focused on how to take full advantages of statistical biases in the data distribution[8], but failed to learn compositional reasoning ability behind complex visual questions. To solve this problem, efforts have been made to build new architecture with explicit reasoning or relational associations, such as module networks[10, 14], reasoning-augmented networks[19, 23, 24] and conditional batch normalization methods[5, 18]. Some of these[18, 23] have shown promising reasoning ability and even outperform humans.

In this paper, we propose a novel conditional batch normalization (CBN) method, termed as Conditional Kronecker Batch Normalization (CKBN). In contrast to prior CBN approaches [5, 18], CKBN layers take the question information as conditioned input and correspondingly modulate the image features via the Kronecker transformation[17], which allows robust counting in images and retains richer information than affine transformation. Thus, CKBN model achieves a higher accuracy on CLEVR dataset than FiLM[18]—a CBN approach. Our main contributions are as follows:

(1) We propose a differentiable neural network layer CKBN that further develops the Conditional Batch Normalization techniques, and we show how CKBN layers increase the robustness of a general model as well as help it achieve stronger reasoning ability.

(2) We demonstrate the Kronecker transformation in CKBN layer is a generalization of standard affine transformation. It could accelerate the fusion of visual and linguistic information, and thus the convergence of overall model.

(3) CKBN layers take the given question as conditioned input and extract various joint features[1] from a single image feature. This process helps the overall model explicitly learn compositional reasoning and counting skills, with each joint feature capturing separate attribute or object of an image.

[1] Joint feature refers to the feature that contains information of both image and question.

2 Related Work

Neural Module Networks Neural module network (NMN)[2, 3] can be viewed as a general class of recursive neural networks[20]. It provides a framework for constructing deep neural networks with dynamic computational structure. Generally, an NMN model is composed of a semantic parser, a layout and a collection of pre-defined modules. The semantic parser maps the given question into an action plan. Based on the action plan, the layout provides a template for assembling an instance-specific network from those pre-defined modules. These modules could either be jointly trained, or be optimized independently as each module has its own set of learned parameters. Each module in the NMN model is trained for learning an elementary reasoning operation, and thus the network assembled from these modules could master the compositional reasoning ability. The recently proposed NMN approaches[10, 14] have achieved competitive VQA performance, which proves NMN a promising method.

Reasoning-Augmented Networks Conventional deep networks learn a mapping directly from inputs to outputs. Although some of these are capable of sophisticated reasoning skills, the monolithic network structures make their behavior difficult to understand, explain or optimize. To this end, a series of reasoning-augmented models[19, 22, 23] were proposed for facilitating explicit and expressive reasoning. These methods usually add components to neural networks that aid them in handling relational associations and compositional reasoning. For example, the Relational Network[19] carries out pairwise comparisons over any two pixel-wise position of extracted image features, and thus enhance the relational reasoning ability of the network. Memory networks[21, 22] and stacked-augmented recurrent networks[15] design explicit memory components for neural networks, which enable the overall model to imitate the human's reasoning process in an iterative manner.

Conditional Batch Normalization Methods Batch Normalization (BN)[11] has proved successful in improving neural network training. It accelerates training and improves generalization by reducing the covariate shift throughout the network. Inspired by BN, Vries *et al*. introduce the Conditional Batch Normalization (CBN) method for language-vision tasks in [5, 18]. These approaches first generate a variance and bias pair for each convolutional feature from a single linguistic input, and then apply an affine transformation to each feature using the generated coefficients, aiming to modulate the covariate shift like BN. However, as these CBN approaches attempt to locate all question-referenced regions via a simple affine transformation, they have struggled on compositional reasoning and object counting.

In contrast, our proposed CKBN layer could be regarded as a generalization of CBN layer. We generate multiple variance and bias pairs for each convolutional feature from a question input and apply the Kronecker transformation on each feature. Besides, CKBN layer facilitates the explicit and expressive learning of compositional reasoning and robust counting in images, which has been proved to be effective in our experiments.

3 Methodology

3.1 Conditional Kronecker Batch Normalization

The internal structure of the CKBN layer is shown in Figure 2, and we introduce the motivation and purpose on the design of CKBN layer in the following four steps.

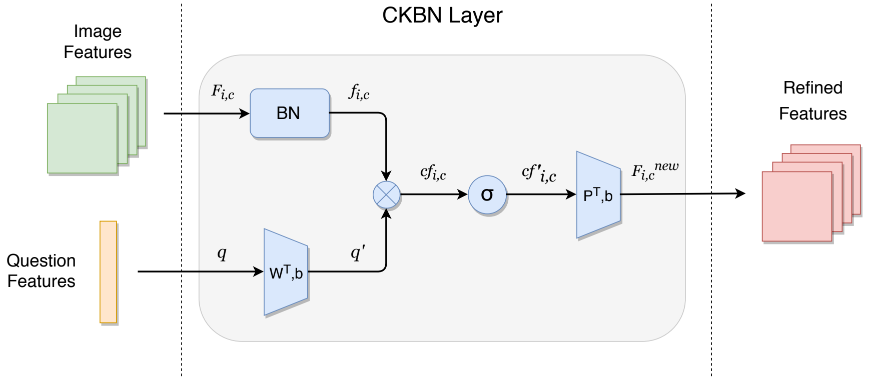


Figure 2: Overview of CKBN layer. The CKBN layer takes the features of images and questions as inputs and outputs joint refined features. Blue refers to different tensor operations.

Feature Processing Before fusing the question feature into image features, we first process both of them for better overall performance. We normalize the image features to accelerate the convergence of overall model, and we apply a linear layer on question features to generate multiple variance and bias pairs that would be used in next Kronecker transformation step.

$$\boldsymbol{f}_{i,c,h,w} = BN(\boldsymbol{F}_{i,c,h,w}) = \frac{\boldsymbol{F}_{i,c,h,w} - \mathrm{E}[\boldsymbol{F}_{\cdot,c,\cdot,\cdot}]}{\sqrt{\mathrm{Var}[\boldsymbol{F}_{\cdot,c,\cdot,\cdot}] + \varepsilon}} \tag{1}$$

$$\boldsymbol{q'} = \begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\beta} \end{bmatrix} = W_q^{\mathrm{T}} \boldsymbol{q} + b_q \tag{2}$$

Here, we define $\mathcal{B} = \{\boldsymbol{F}_{i,\cdot,\cdot,\cdot}\}_{i=1}^{I}$ as a mini-batch of I samples, and $\boldsymbol{F}$ corresponds to feature maps whose subscripts c, h, w refers to the c^{th} feture map at the spatial location (h, w).
$\boldsymbol{f}$ refers to the corresponding normalized feature maps of $\boldsymbol{F}$, and ε is a constant damping factor for numerical stability. $W_q \in \mathbb{R}^{M \times N}$ and $b_q \in \mathbb{R}^N$ stands for the weight matrix and bias for the output $\boldsymbol{q'}$, respectively. The function of $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ will be explained below.

Kronecker Transformation The Kronecker transformation facilitates the learning of compositional reasoning and robust counting in images. It enables CKBN layer to extract various joint features from a single image feature, with each joint feature capturing separate attributes or objects of an image. Also, It significantly increases the model's robustness, as it uses multiple joint features to locate all question-referenced regions instead of a single one.

$$\boldsymbol{cf}_{i,c} = \boldsymbol{\gamma}_c \otimes \boldsymbol{f}_{i,c} + \boldsymbol{\beta}_c \tag{3}$$

where $\boldsymbol{cf}_{i,c}$ is termed as *controlled feature*, $\otimes$ denotes the Kronecker product[17].
In Equation 2, $\boldsymbol{q'} \in \mathbb{R}^N$ is sliced into two column vectors $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. Let $N = (K+1)C$, $\boldsymbol{\gamma} \in \mathbb{R}^{KC}$ is composed of C different $\boldsymbol{\gamma}_c \in \mathbb{R}^K$, and $\boldsymbol{\beta} \in \mathbb{R}^C$ is composed of C different $\boldsymbol{\beta}_c \in \mathbb{R}$, where C is the number of feature maps, K is an arbitrary postive integer. For example, when $K = 1$ ($\boldsymbol{\gamma}_c \in \mathbb{R}$), Equation 3 degenerates into the standard affine transformation.

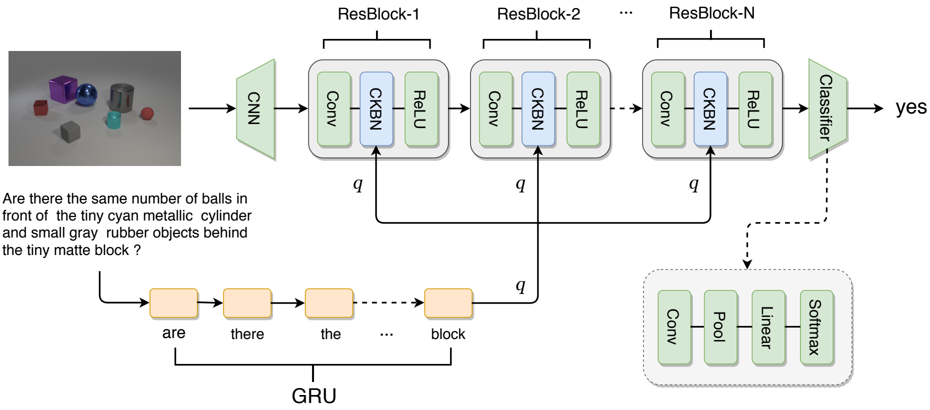

ResBlock-1
ResBlock-2
···
ResBlock-N
CNN
Conv
CKBN
ReLU
Conv
CKBN
ReLU
Conv
CKBN
ReLU
Classifier
yes
q
q
q
q
Are there the same number of balls in front of the tiny cyan metallic cylinder and small gray rubber objects behind the tiny matte block ?
are
there
the
···
block
GRU
Conv
Pool
Linear
Softmax

Figure 3: The linguistic feature generator (bottom), and the CKBN network (top).

Non-Linear Activation Apply a non-linear activation function may help to increase the representative capacity of the overall model. The first candidate is to apply the non-linear activation function right after the Kronecker transformation.

$$\boldsymbol{cf}'_{i,c} = \sigma(\boldsymbol{\gamma}_c \otimes \boldsymbol{f}_{i,c} + \boldsymbol{\beta}_c) \tag{4}$$

where σ denotes an arbitrary non-linear activation function, *e.g. ReLU* or *Sigmoid*.

Linear Projection It is sensible to reduce the rank of the feature matrices using linear projection, as it would lead to the reduction of parameters for regularization. What's more, the Kronecker product of two matrices would scale up the matrix elements. For example, the Kronecker product $A_{m\times n} \otimes B_{p\times q}$ is the $mp \times nq$ block matrix. Linear projection could be used to prevent this parameter explosion and still retain important information.

$$\boldsymbol{F}^{new}_{i,c} = P^{\mathrm{T}} \sigma(\boldsymbol{\gamma}_c \otimes \boldsymbol{f}_{i,c} + \boldsymbol{\beta}_c) + b \tag{5}$$

where $P \in \mathbb{R}^{L\times d}$ and $b \in \mathbb{R}^d$ stands for the projection matrix and bias for the jointly refined feature $\boldsymbol{F}^{new}_{i,c}$, respectively. Note that d is a hyperparameter to decide the output dimension of the CKBN layer. For example, d could be set equal to the input dimension of CKBN layer, enabling CKBN to operate like the normal CBN layer but contain richer information.

3.2 Model

The overview of network architecture is shown in Figure 3. It consists of a linguistic pipeline and a visual pipeline. The linguistic pipeline extracts question features with a Gated Recurrent Unit (GRU) which has 4096 hidden units and 200-dimensional word embeddings.

The visual pipeline extracts the image features using the *conv4* layer of a ResNet-101 pre-trained on ImageNet with a learnable 3 x 3 convolutional layer, to match prior works on CLEVR[13]. The extracted image features are processed by several CKBN residual blocks (ResBlocks) [9] and a classifier. Similar to FiLM[18], Each CKBN ResBlock consists of a 3x3 convolutional layer that outputs 128 feature maps, a conditional batch normalization layer (CKBN in this work), and a ReLU activation. The final classifier is composed of a 1x1 convolutional layer that outputs 512 feature maps, a global max-pooling layer, and a two-layer MLP with 1024 hidden units. The second layer of MLP outputs the softmax distribution over answers. Besides, inspired by prior works on CLEVR[10, 19], we concatenate two coordinate feature maps representing relative x and y spatial position with the convolutional features, each ResBlock's input and the classifier's input to facilitate spatial reasoning.

CKBN model is trained end-to-end from scratch without data augmentation and extra supervision information. We use Adam optimizer with learning rate $3e^{-4}$, weight decay $1e^{-5}$, and set batch size 64 to match prior work on CLEVR. For our best model, we set $N = 4$ (4 ResBlocks), $K = 2$ (In Equation 3, $\gamma_c \in \mathbb{R}^2$), and train a maximum of 90 epochs.

4 Experiments

In this section, we test our CKBN model on CLEVR and its associated dataset. First, we evaluate our model on the standard CLEVR dataset and analyze what CKBN layer learns. Then, we explore the compositional reasoning capacity of CKBN model on the CLEVR Compositional Generalization Test (CLEVR-CoGenT). Finally, we examine the performance of CKBN model on more challenging CLEVR-Humans dataset, which consists of the human-sourced natural language questions on the given image.

4.1 Standard CLEVR Task

CLEVR is a synthetic dataset of 700K tuples with 3D-rendered images and automatically generated questions, as shown in Figure 1. The images feature different shapes, materials, colors and sizes. The questions measure various aspects of visual reasoning skills including attribute identification, counting, comparison, spatial relationships, and logical operations. In addition, each question has an associated machine-readable program, specifying the reasoning process that leads to the correct answer, among 28 possibilities.

We perform experiments on the original 700K CLEVR dataset[13] and achieve a competitive accuracy, as shown in Table 1. Notably, our model outperforms the FiLM model (a CBN approach) trained from pre-trained features in overall accuracy as well as in each category, which demonstrates the generalization and robustness of CKBN layer.

Counting and Numerical Comparison As shown in Table 1, CKBN model outperforms most other competing methods on questions about counting and numerical comparison, which is comparable to CAN [24]. These results demonstrate the CKBN layer's capacity on processing counting and aggregation. We stress that our model relies solely on the Kronecker transformation based on conditioned linguistic information to adaptively alter the CKBN network's behavior to answer questions. That means, a standard CNN-based approach without data augmentation, strong supervision or additional mechanisms like attention could also perform well on those intractable VQA problems, thanks to CKBN layers.

Computational Efficiency We compare the computational efficiency of our model with other competing methods (RN, FiLM), as they are all CNN-based leading approaches similar to ours. Santoro *et al.* report in [19] that the Relational Network model was trained approximately 1.4 million iterations to achieve 95.5% accuracy, which are equivalent to 125 epochs approximately, while our model achieves a higher accuracy after 15 epochs, leading

Model	**Overall**	Count	Exist	Compare Numbers	Query Attribute	Compare Attribute
Human [13]	92.6	86.7	96.6	86.5	95.0	96.0
Q-type baseline[13]	41.8	34.6	50.2	51.0	36.0	51.3
LSTM [13]	46.8	41.7	61.1	69.8	36.8	51.8
CNN+LSTM [13]	52.3	43.7	65.2	67.1	49.3	53.0
CNN+LSTM+SA [23]	76.6	64.4	82.7	77.4	82.6	75.4
N2NMN* [10]	83.7	68.5	85.7	84.9	90.0	88.7
PG+EE (9K prog.)* [14]	88.6	79.7	89.7	79.1	92.6	96.0
PG+EE (700K prog.)* [14]	96.9	92.7	97.1	**98.7**	98.1	98.9
CNN+LSTM+RN†‡ [19]	95.5	90.1	97.8	93.6	97.9	97.1
CNN+GRU+FiLM [18]	97.7	94.3	99.1	96.8	99.1	99.1
CNN+GRU+FiLM‡ [18]	97.6	94.3	99.3	93.4	**99.3**	**99.3**
CNN+GRU+CKBN	**98.4**	**96.1**	**99.4**	**97.8**	**99.3**	**99.3**

Table 1: CLEVR accuracy by baseline methods, competing methods, and our method (CKBN). (*) denotes use of extra supervisory information through program labels. (†) denotes use of data augmentation. (‡) denotes training from raw pixels.

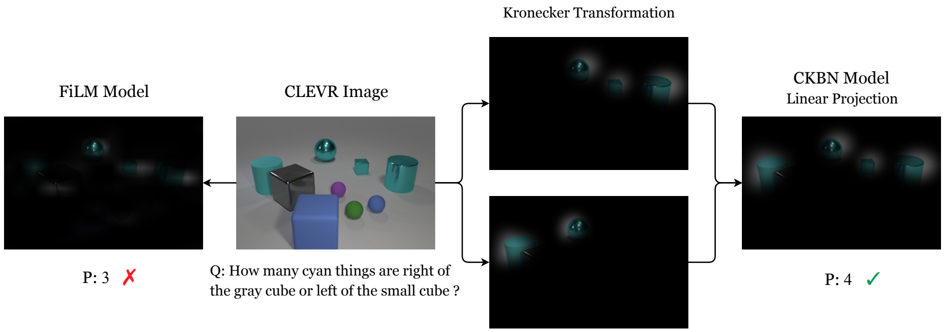


Figure 4: Comparison with FiLM (left). Features extracted from the final ResBlock of our model (right). P refers to the predicted answer. Image and question matches FiLM [18].

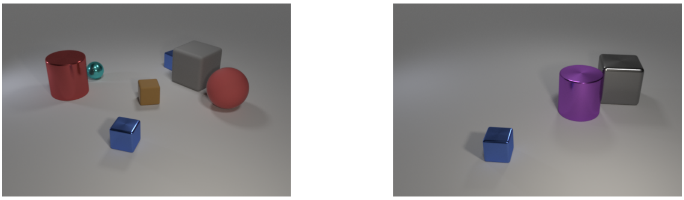
to approximately 8.5x reduction in training time. Perez *et al*. report in [18] that they train 80 epochs for their best model, achieving 97.7% accuracy. In contrast, our CKBN model achieves a comparable accuracy in 50 epochs, yielding 1.6x reduction in training time.

4.2 Why CKBN Layer Helps?

To have a deep insight of how CKBN layers work in the overall model, we visualize parts of the feature activations used to answer related CLEVR questions. Note that this visualization is based on the CKBN model with 4 ResBlocks and $\gamma_c \in \mathbb{R}^2$, aiming to have a fair comparison with best FiLM model, which could be viewed as the model with 4 ResBlocks and $\gamma_c \in \mathbb{R}$.

We ask the same question to both FiLM and CKBN model, as shown in Figure 4. FiLM model only captures parts of answer-related objects while our model locates all cyan things both "right of the gray cube" and "left of the small cube" via the Kronecker transformation.

In practical training, it is difficult or even impossible to find a global optimal solution to locate all question-related regions in a single feature activation, but it is much easier to find



(a) **Q:** How many tiny metal objects have the same shape as the tiny matte thing ? **A: 2**

(b) **Q:** Is there any other thing that is the same material as the gray object ? **A: yes**

Figure 5: Two illustrative instances sampled from CLEVR-CoGenT in Condition A.

a local optimal solution to capture parts of question-interested regions. Thus, we apply the Kronecker transformation to a single image feature, which enables the CKBN layer to extract various joint features from it. That is to say, we could find many local optimal solutions, with each joint feature only responsible for parts of question-related regions. Then, a linear projection is applied to map various local optimal solutions to the global one. This approach makes the overall model easier to be trained and converge more quickly. Also, it greatly increases the generalization and robustness of the model.

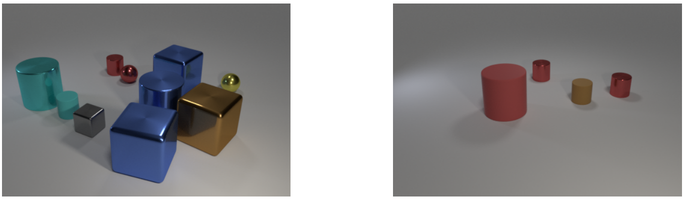
4.3 Compositional Generalization Test

To investigate the VQA model's capacity on compositional generalization, Johnson *et al.* introduced CLEVR-CoGenT[13], as shown in Figure 5. The dataset contains two different conditions: in Condition A, all cubes are gray, blue, brown, or yellow and all cylinders are red, green, purple, or cyan; in Condition B, cubes and cylinders swap color palettes. Thus, models demand the ability to learn separate representations for color and shape, rather than rote memory on all possible color/shape combinations, to achieve better performance.

Model	Train A		Finetune B	
	A	B	A	B
LSTM	55.2	50.9	51.5	54.9
CNN+LSTM	63.7	57.0	58.3	61.1
CNN+LSTM+SA+MLP	80.3	68.7	75.7	75.8
PG+EE (18K prog.)	96.6	73.7	76.1	92.7
CNN+GRU+FiLM	98.3	75.6	80.8	96.9
CNN+GRU+CKBN	**98.4**	**76.7**	**81.1**	**97.5**

Table 2: Accuracy on the CLEVR-CoGenT dataset. First, we train CKBN model on Condition A, and test them on both Condition A and Condition B (left). Then, we finetune the model on Condition B using 30K samples, and again test on both Conditions (right).

Results We perform experiments with our best CLEVR-trained model architecture on CLEVR-CoGenT, as shown in Table 2. Notably, our resulting model outperforms all prior methods both before and after finetuning on 30K ValB. This is because the design of CKBN layer allows the overall model to explicitly learn separate attribute of an image. CKBN layer takes the question as conditioned input and extracts various joint features from a single image feature based on different K ($\gamma_c \in \mathbb{R}^K$). For example, it could take one joint feature



(a) **Q:** How many of these things could be stacked on top of each other ? **A: 8**

(b) **Q:** If the largest item was removed what color would be most seen in this set ? **A: red**

Figure 6: Examples of questions from CLEVR-Humans dataset, which introduces new words and concepts. Words that do not appear in CLEVR questions are underlined.

to learn color and another to learn shape. Then, a final linear projection is applied to obtain the compositional attributes of an object. Experiment results also demonstrate our model's robustness on learning general concepts and separate representation.

4.4 CLEVR-Humans

CLEVR-Human dataset is composed of 18K human-posed natural language questions on CLEVR images, as shown in Figure 6. The questions were collected by Amazon Mechanical Turk workers who were asked to write questions about CLEVR images that would *be hard for a small robot to answer*. As the questions were proposed from different workers, the dataset has diverse vocabulary and linguistic variety. Thus, the model needs to have more varied reasoning skills to perform well on the dataset.

Model	Train CLEVR	Train CLEVR, finetune human
LSTM	27.5	36.5
CNN+LSTM	37.7	43.2
CNN+LSTM+SA+MLP	50.4	57.6
PG+EE (18K prog.)	54.0	66.6
CNN+GRU+FiLM	56.6	75.9
CNN+GRU+CKBN	**58.2**	**76.4**

Table 3: Accuracy on the CLEVR-Humans dataset after training on just the CLEVR dataset (left) and after finetuning on the CLEVR-Humans dataset (right).

Results We first train our model on CLEVR, and then finetune the model on CLEVR-Humans to make it adaptive to additional vocabulary and linguistic variety. During finetuning, our model learns to use more complex questions to freely modulate the existing feature maps. This fine-grained operation makes the model reason in a more flexible way, leading to the correct answer. Quantitatively, the results in Table 3 also demonstrate our CKBN model's robustness against linguistic variations and noise, as well as its ability to handle more diverse vocabulary and complex questions.

5 Conclusion

In this paper, we introduce a novel CKBN layer that further develops the Conditional Batch Normalization techniques, and we show how CKBN layers increase the generalization and robustness of a general model as well as accelerate its convergence. Also, we analyze the reasons behind the competitive performance of the CKBN model. In further work, we would further explore the Condtional Normalization (CN) techniques in following two directions. First, as our design of CKBN was based on the concept of compositionality in visual question answering (VQA), we wish to generalize this work to the non-conditional version called Kronecker Batch Normalization to accelerate neural network's training. Second, we would extend the series of Conditional Normalization Techniques (both prior works [5, 18] and ours) to different tasks and domains, including real-world VQA, speech recognition and machine comprehension.

6 Acknowledgement

This work is supported by the National High Technology Research and Development Plan (863 Plan) under Grant No. 2015AA015803, the NSFC project under Grant No. U1433112, and the Joint Research Center of Tencent & Tsinghua University.

We thank Perez *et al.* for the inspiring work FiLM and their open-source code on FiLM architecture. We also thank Johnson *et al.* for their proposed CLEVR dataset and open-source code, which our work were based off.

References

[1] A. Agrawal, D. Batra, and D. Parikh. Analyzing the behavior of visual question answering models. In *EMNLP*, 2016.

[2] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *CVPR*, 2016.

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Learning to compose neural networks for question answering. In *NAACL*, 2016.

[4] S. Antol, A. Agrawal, and Jiasen Lu. VQA: Visual Question Answering. In *ICCV*, 2015.

[5] H. de Vries, F. Strub, and J. Mary. Modulating early visual processing by language. In *NIPS*, 2017.

[6] J. Devlin, S. Gupta, and R. Girshick. Exploring nearest neighbor approaches for image captioning, 2015. arXiv preprint arXiv:1505.04467.

[7] A. Fukui, D. H. Park, D. Yang, and A. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *EMNLP*, 2016.

[8] Y. Goyal, T. Khot, D. Summers-Stay, and D. Batra. Making the v in vqa matter: Elevating the role of image understanding in visual question answering, 2017. CoRR, abs/1612.00837,.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[10] R. Hu, J. Andreas, and M. Rohrbach. Learning to reason: End-to-end module networks for visual question answering. In *ICCV*, 2017.

[11] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[12] A. Jabri, A. Joulin, and L. van der Maaten. Revisiting visual question answering baselines. In *ECCV*, 2016.

[13] J. Johnson, B. Hariharan, and L. van der Maaten. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2016.

[14] J. Johnson, B. Hariharan, and L. van der Maaten. Inferring and executing programs for visual reasoning. In *ICCV*, 2017.

[15] A. Joulin and T. Mikolov. Inferring algorithmic patterns with stack-augmented recurrent nets. In *NIPS*, 2015.

[16] B. M. Lake, T. D. Ullman, and J. B. Tenenbaum. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 2016.

[17] Charles F. Van Loan. The ubiquitous kronecker product. *Journal of Computational and Applied Mathematics*, 2000.

[18] E. Perez, F. Strub, and H. de Vries. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

[19] A. Santoro, D. Raposo, and D. G. T. Barrett. A simple neural network module for relational reasoning, 2017. arXiv preprint arXiv:1706.01427.

[20] R. Socher, A. Perelygin, J. Y. Wu, and J. Chuang. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.

[21] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. End-to-end memory networks. In *NIPS*, 2014.

[22] J. Weston, S. Chopra, and A. Bordes. Memory networks. In *ICLR*, 2015.

[23] Z. Yang, X. He, and J. Gao. Stacked attention networks for image question answering. In *CVPR*, 2016.

[24] Z. Yang, X. He, and J. Gao. Compositional attention networks for machine reasoning. In *ICLR*, 2018.

[25] P. Zhang, Y. Goyal, and D. Summers-Stay. Yin and yang: Balancing and answering binary visual questions. In *CVPR*, 2016.