삼성 DS-KAIST AI Expert 프로그램

Visual Question Answering

TA: Jongheon Jeong, Jongjin Park

KAIST ALIN Lab. (Prof. Jinwoo Shin)

Aug 5, 2020

Project Overview

Goal: VQA modeling on Sort-of-CLEVR dataset

- VQA 모델링 파이프라인 이해
- Relation Network, FiLM 구현 및 비교 분석

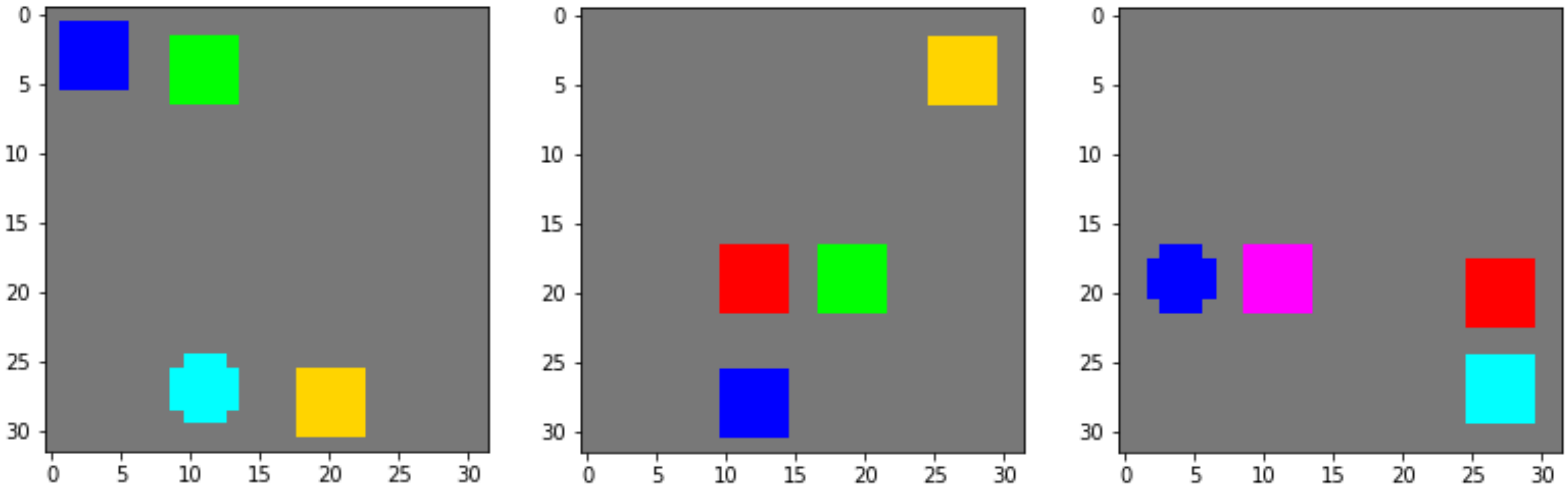
Tasks:

1. Sort-of-CLEVR 데이터셋 파악
2. Relation Network
3. FiLM-based Model
4. 구현 모델 간 성능 비교 분석
5. 모델 튜닝을 통한 성능 개선

Sort-of-CLEVR Dataset

An **even-simpler version** of CLEVR dataset

- 2 kinds of shape (rectangle, circle), 6 colors
- 10,000 images × 20 questions = 200,000 (I, q, a) samples



Sort-of-CLEVR Dataset

An **even-simpler version** of CLEVR dataset

- 2 kinds of shape (rectangle, circle), 6 colors
- 10,000 images × 20 questions = 200,000 (I, q, a) samples

Each question is hard-coded into an 11-dim vector

- No need to run language-embedding model, e.g., LSTM

0	0	0	0	0	1	0	0	0	1	0
Colors						**Q-types**				

Non-relational

```
0: 'is it a circle or a rectangle?',
1: 'is it closer to the bottom of the image?',
2: 'is it on the left of the image?',
3: 'the color of the nearest object?',
4: 'the color of the farthest object?',
```

Relational

Sort-of-CLEVR Dataset

An **even-simpler version** of CLEVR dataset

- 2 kinds of shape (rectangle, circle), 6 colors
- 10,000 images × 20 questions = 200,000 (I, q, a) samples

Each question is hard-coded into an 11-dim vector

- No need to run language-embedding model, e.g., LSTM

10 possible answers per question

- 6 colors + 2 shapes + {yes, no}
- An answer is represented by a 10-dim 1-hot vector

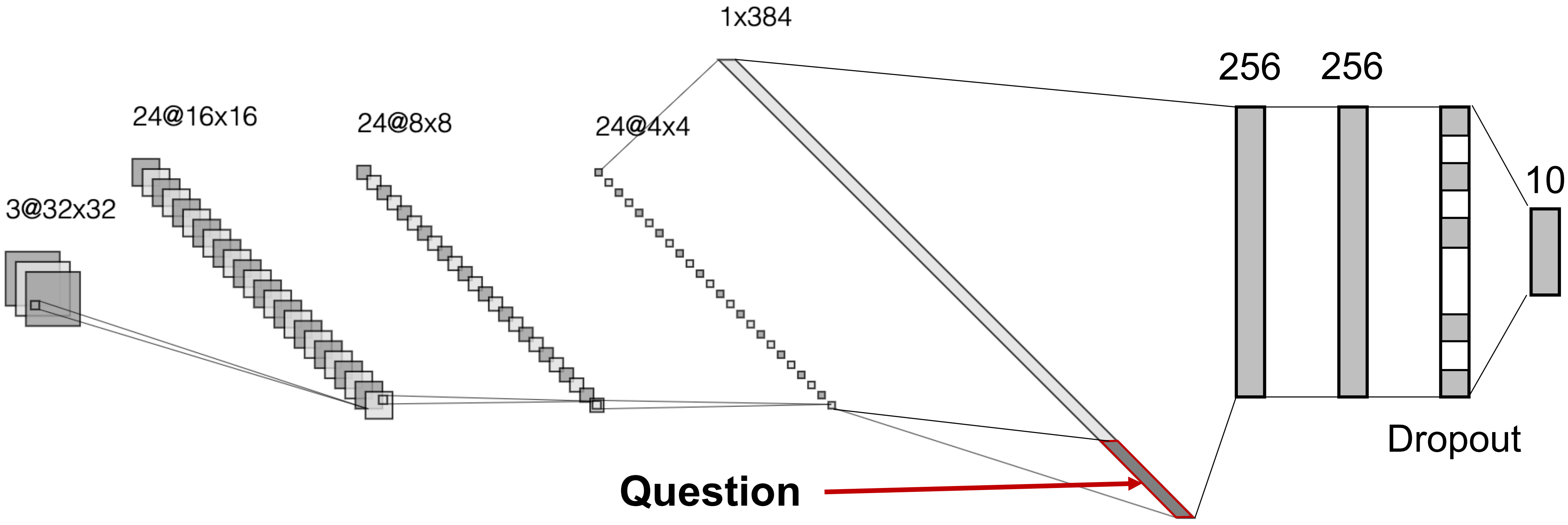
```
0: 'blue',
1: 'green',
2: 'red',
3: 'yellow',
4: 'magenta',
5: 'cyan',
6: 'circle',
7: 'rectangle',
8: 'yes',
9: 'no',
```

Sort-of-CLEVR Dataset

Baseline CNN model

- Implemented in `**models/baseline.py**`
- 4 conv. layers + 3 fully-connected layers
- Q-vector is concatenated after the final conv. layer

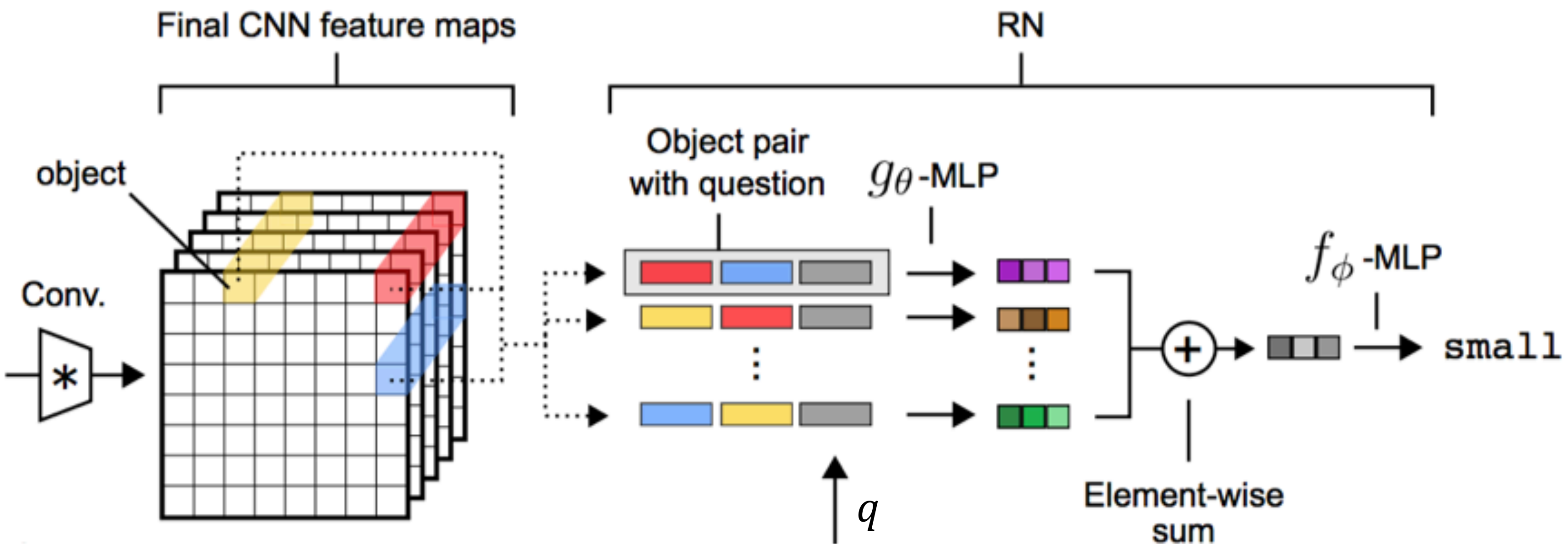
Task 1: Load a pre-trained baseline and evaluate it on Sort-of-CLEVR



Relation Network (RN)

An explicit constraint on NN architecture could improve VQA

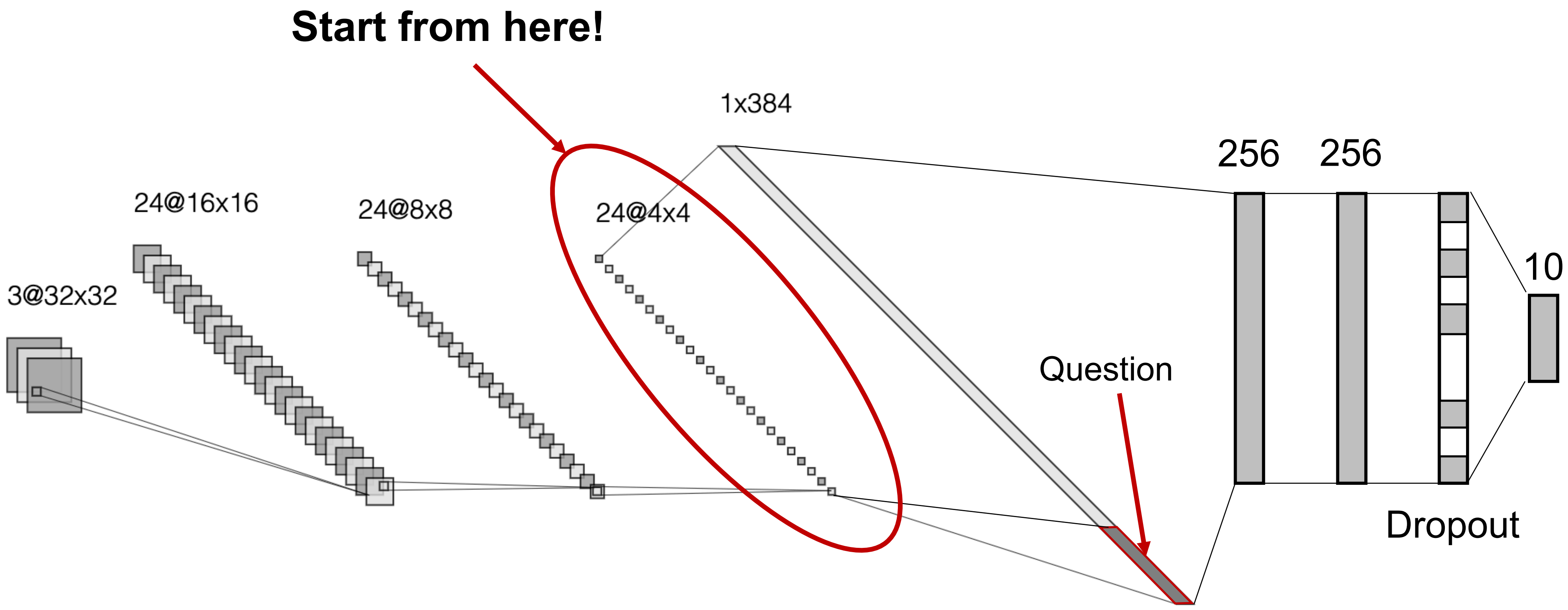
$$\underbrace{a}_{\text{answer}} = f_\phi\left(\sum_{i,j} \underbrace{g_\theta}_{\text{“relation”}}(\underbrace{o_i, o_j}_{\text{objects}}, \underbrace{q}_{\text{question}})\right)$$



Relation Network (RN)

Task 2: Implement RN

- We use the 24@4×4 features of the baseline for I-embedding



Relation Network (RN)

Task 2: Implement RN

- We use the 24@4×4 features of the baseline for I-embedding

- **Task 2-1**: Add positional encoding per feature vector
  - Append 2 feature maps that represent pixel coordinates
  - (0, 0) upper left ~ (1, 1) lower right

- **Task 2-2**: Implement $\sum_{i,j} g_\theta(o_i, o_j, q)$

- **Task 2-3**: Implement f_ϕ

$$\underbrace{a}_{\text{answer}} = f_\phi\left(\sum_{i,j} \underbrace{g_\theta}_{\text{“relation”}}(\underbrace{o_i, o_j}_{\text{objects}}, \underbrace{q}_{\text{question}})\right)$$

Relation Network (RN)

Task 2: Implement RN

- We use the 24@4×4 features of the baseline for I-embedding

- ~~**Task 2-1**: Add positional encoding per feature vector~~
  - Append 2 feature maps that represent pixel coordinates
  - (0, 0) upper left ~ (1, 1) lower right

- **Task 2-2**: Implement $\sum_{i,j} g_\theta(o_i, o_j, q)$ → **TODO**

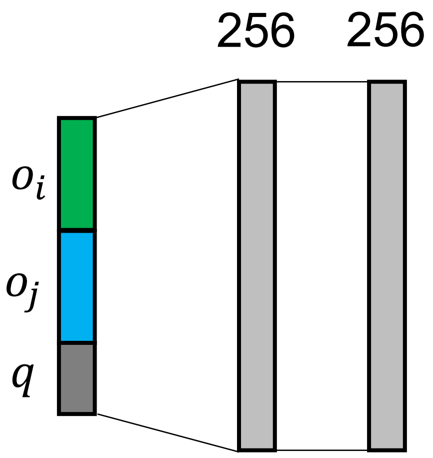
- ~~**Task 2-3**: Implement~~ $\cancel{f_\phi}$

$$\underbrace{a}_{\text{answer}} = f_\phi\left(\sum_{i,j} \underbrace{g_\theta}_{\text{“relation”}}(\underbrace{o_i, o_j}_{\text{objects}}, \underbrace{q}_{\text{question}})\right)$$

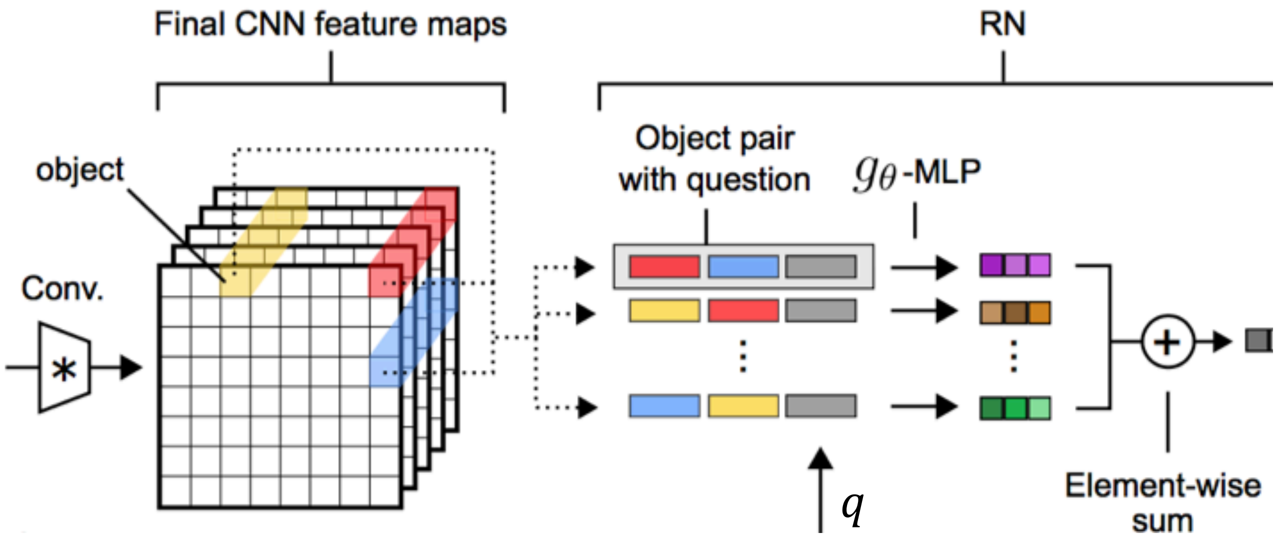
Relation Network (RN)

Task 2: Implement RN

- We use the 24@4×4 features of the baseline for I-embedding



- **TODO**: **Implement** $\sum_{i,j} g_\theta(o_i, o_j, q)$
    1. i and j ranges from **1 ~ 4x4 (=16)**
    2. θ is **shared** across i, j, and assumed to be a **2-layer MLP**
    3. g_θ is specified as given in the left (MLP: 63-256-256)



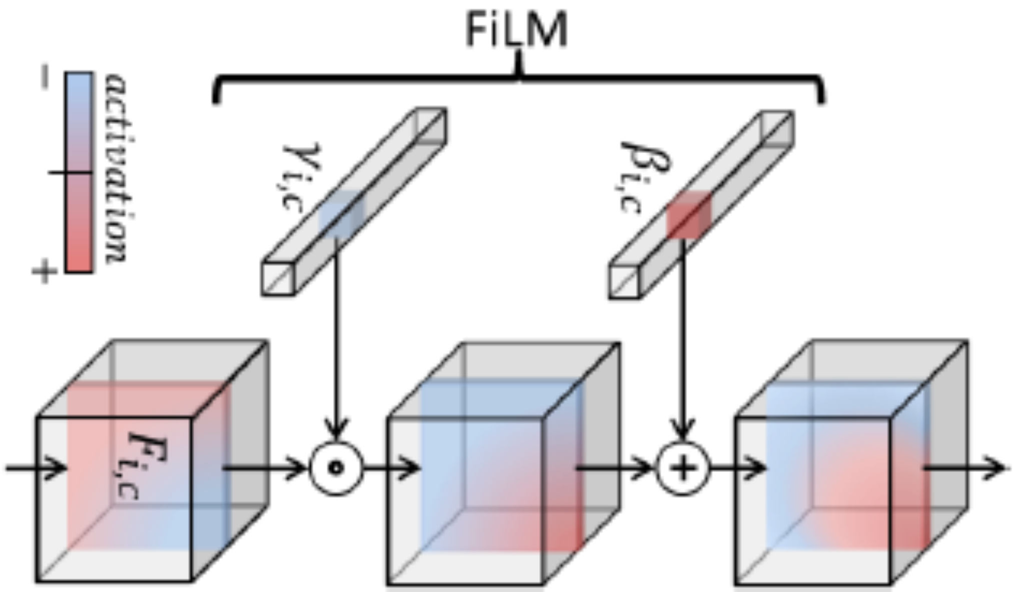
$$\underbrace{a}_{\textbf{answer}} = f_\phi\left(\sum_{i,j} \underbrace{g_\theta}_{\textbf{“relation”}}(\underbrace{o_i, o_j}_{\textbf{objects}}, \underbrace{q}_{\textbf{question}})\right)$$

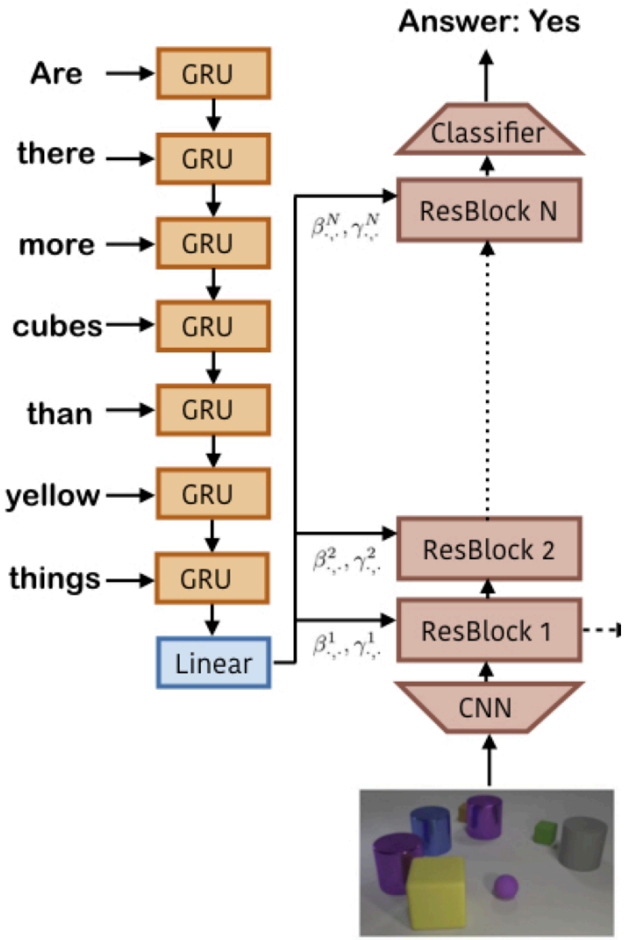
FiLM

Feature-wise affine transform is enough for conditioning a question

$$FiLM(\boldsymbol{F}_{i,c}|\gamma_{i,c}, \beta_{i,c}) = \gamma_{i,c}\boldsymbol{F}_{i,c} + \beta_{i,c}$$

$$(\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i) = \mathrm{Linear}_i(\mathrm{GRU}(q_i))$$



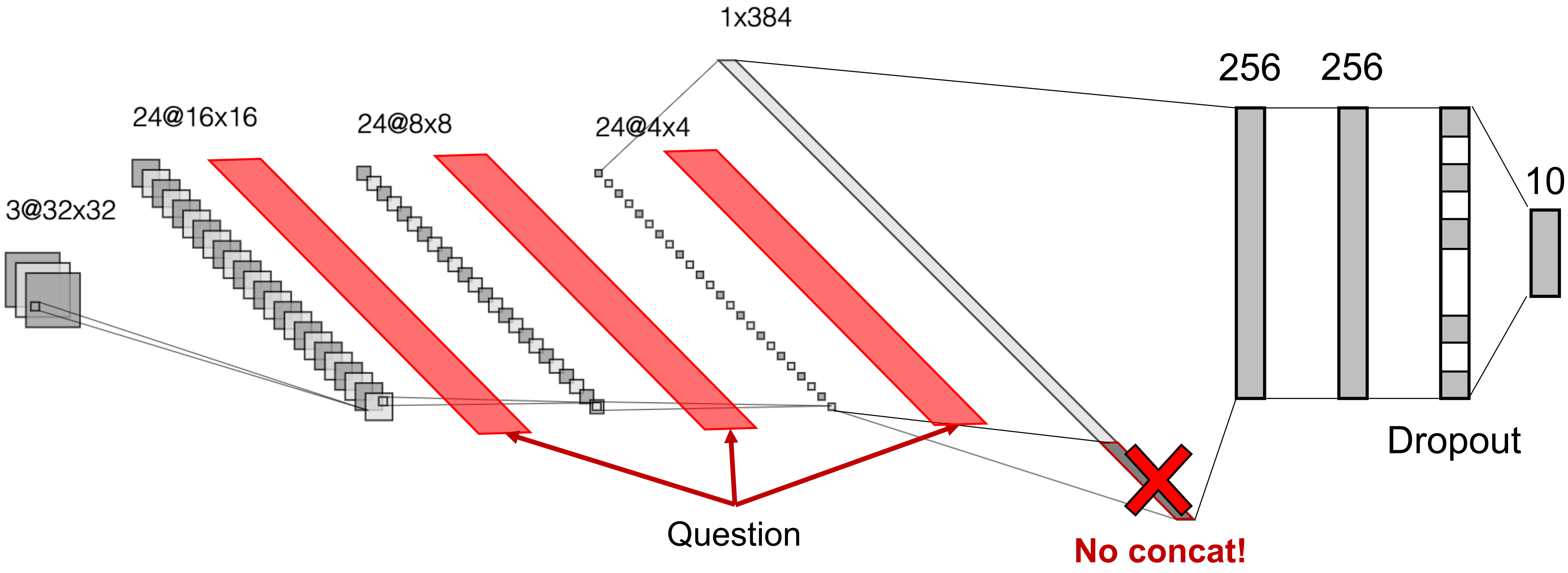


FiLM

$$FiLM(\boldsymbol{F}_{i,c}|\gamma_{i,c}, \beta_{i,c}) = \gamma_{i,c}\boldsymbol{F}_{i,c} + \beta_{i,c}$$

Task 3: Implement FiLM

- **TODO**: Add FiLM layers for each convolutional outputs of the baseline
- Use a 2-layer MLP to model (γ, β): $(\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i) = \mathrm{MLP}(q_i)$



Project Overview

Task 4: Compare and evaluate Baseline, RN and FiLM

- Can you say which method is superior to the others?
- Do they transfer to other Sort-of-CLEVR datasets, e.g., different # objects?

Task 5: Further improvements?

- Hyperparameter tuning using the validation dataset
- Data augmentation
- Training details
- Advanced models
- …

Project Overview

One can get the skeleton code in the following link:

- `git clone https://github.com/alinlab/0805_vqa.git`

The tutorial is driven in jupyterlab

- Make sure the material is accessible by jupyter console
- Read & follow the instructions in the project file

One can also track more training stats via Tensorboard

- `tensorboard --logdir=train_dir`