THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Machine Learning II
— Visual Question Answer

Group 7
Sipeng Wang
Jiafang Liu
Kangning Gao

12/11/2017

Outline

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

- Introduction
- Data Process
- VAQ Model
- Results
- Conclusion

Introduction

What is the mustache made of?

AI System

bananas

View Dataset - MSCOCO

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Image Set

204,721 Image quantities

Input: 4096 dimensions feature

Question Set

1,105,904 questions

Input: 384 dimensions feature

```
sipeng_wang@ml-class-ubuntu16:~/final/data/preprocessed$ cat questions_val2014.t
xt | head -10
Where is he looking?
What are the people in the background doing?
What is he on top of?
What website copyrighted the picture?
Is this a creamy soup?
Is this rice noodle soup?
What is to the right of the soup?
What is the man doing in the street?
How many photo's can you see?
What does the truck on the left sell?
```

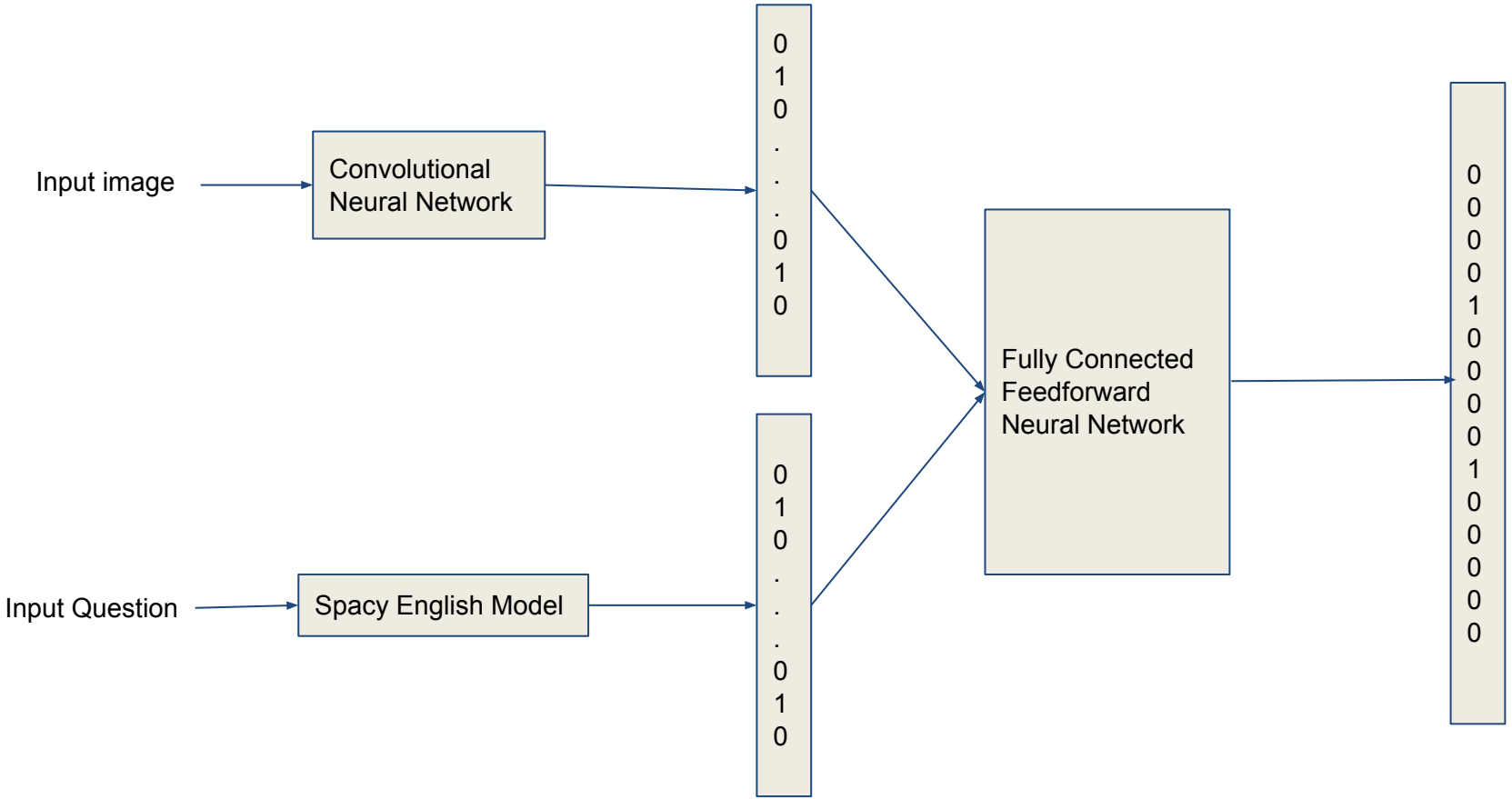
Answer Set

Top 1000 answers from

11,059,040 ground truth answer

```
sipeng_wang@ml-class-ubuntu16:~/final/data/preprocessed$ cat answers_val2014_mod
al.txt |head -10
down
watching
picnic table
foodiebakercom
no
yes
chopsticks
walking
1
ice cream
```

VQA Model



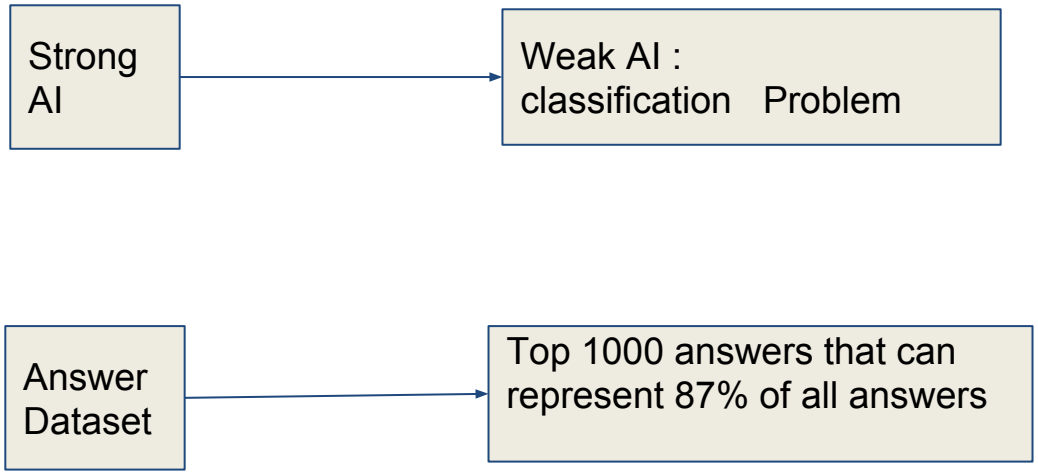
Data Pre-process

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

- Extract data
- Generate Answer
- Categorize answer data

VQA Model

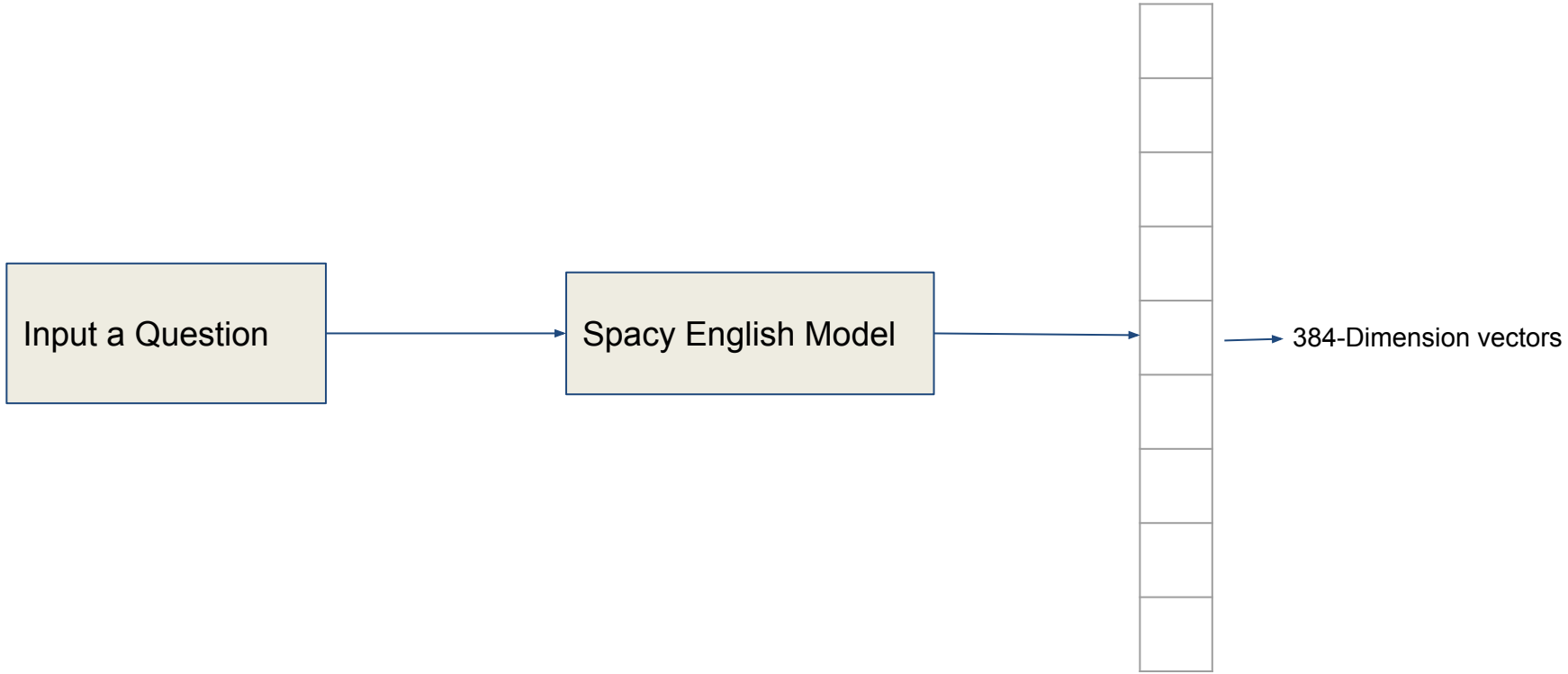
- Generate Answers



VQA Model

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

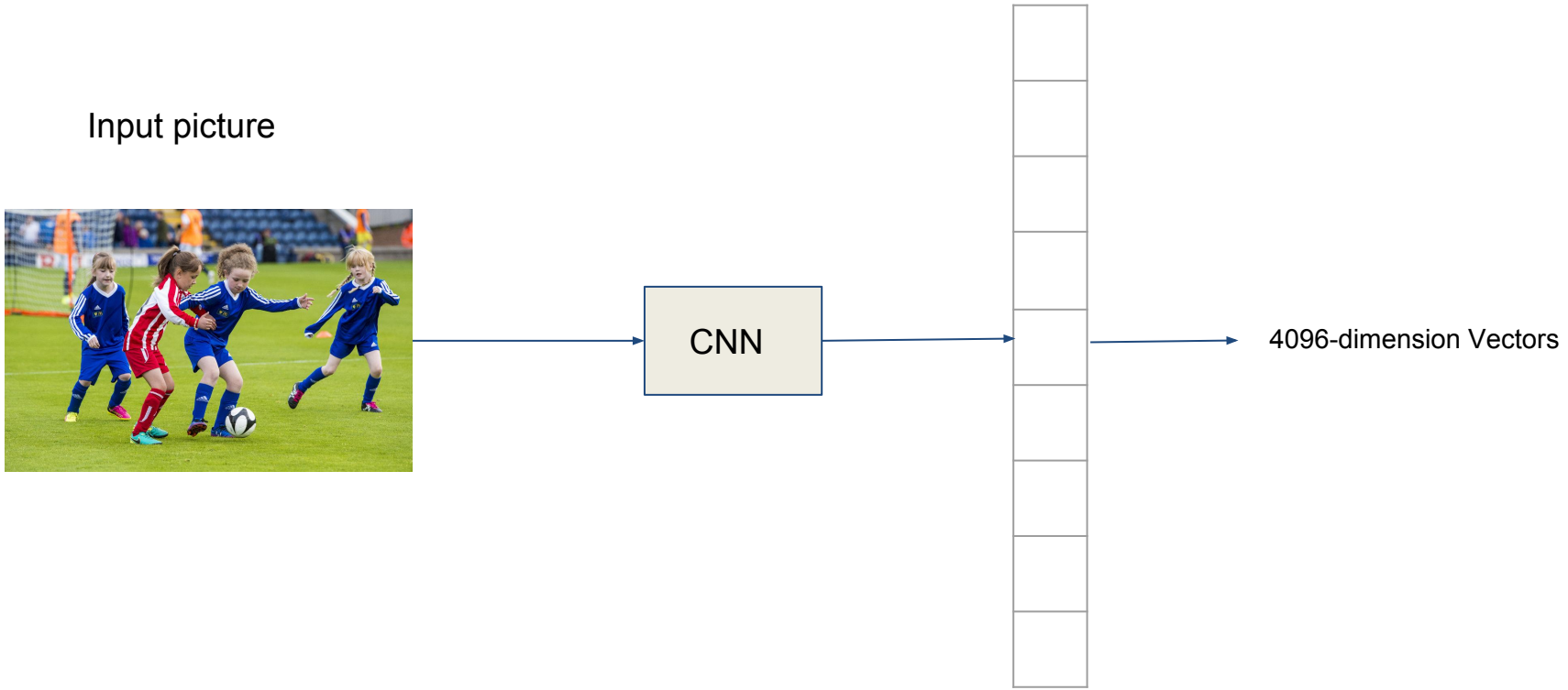
- Flatten Questions



VQA Model

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

- Flatten Pictures

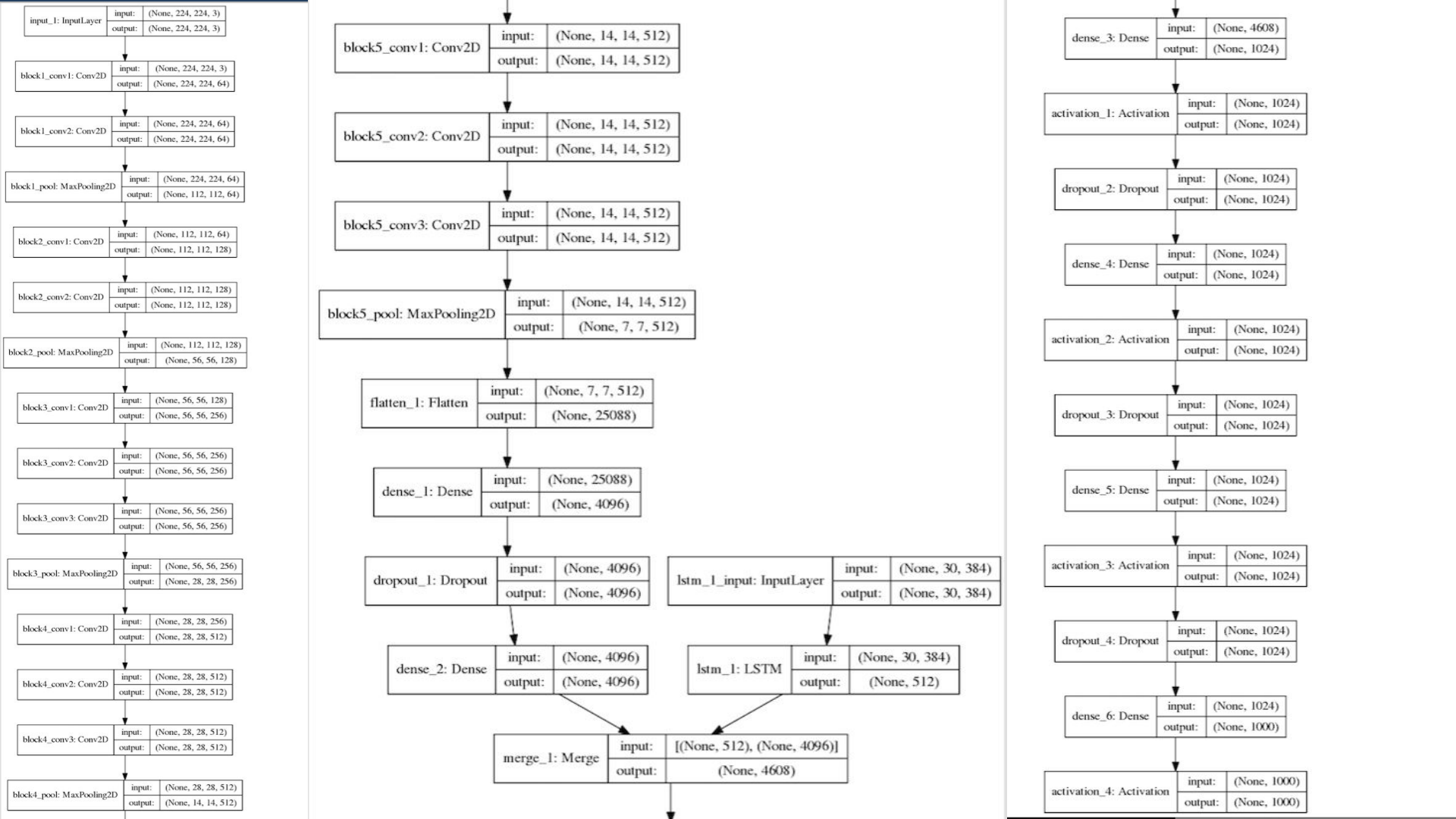


VQA Model

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

VGG 16

```
def VGG_16(weights_path=None):
    model = Sequential()
    model.add(ZeroPadding2D((1,1),input_shape=(3,224,224)))
    model.add(Convolution2D(64, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(64, 3, 3, activation='relu'))
    model.add(MaxPooling2D((2,2), strides=(2,2)))

    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(128, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(128, 3, 3, activation='relu'))
    model.add(MaxPooling2D((2,2), strides=(2,2)))

    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(256, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(256, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(256, 3, 3, activation='relu'))
    model.add(MaxPooling2D((2,2), strides=(2,2)))

    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(MaxPooling2D((2,2), strides=(2,2)))

    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(ZeroPadding2D((1,1)))
    model.add(Convolution2D(512, 3, 3, activation='relu'))
    model.add(MaxPooling2D((2,2), strides=(2,2)))

    model.add(Flatten())
    model.add(Dense(4096, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(4096, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1000, activation='softmax'))

    if weights_path:
        model.load_weights(weights_path)

    return model
```


input_1: InputLayer | input: (None, 224, 224, 3) | output: (None, 224, 224, 3)
block1_conv1: Conv2D | input: (None, 224, 224, 3) | output: (None, 224, 224, 64)
block1_conv2: Conv2D | input: (None, 224, 224, 64) | output: (None, 224, 224, 64)
block1_pool: MaxPooling2D | input: (None, 224, 224, 64) | output: (None, 112, 112, 64)
block2_conv1: Conv2D | input: (None, 112, 112, 64) | output: (None, 112, 112, 128)
block2_conv2: Conv2D | input: (None, 112, 112, 128) | output: (None, 112, 112, 128)
block2_pool: MaxPooling2D | input: (None, 112, 112, 128) | output: (None, 56, 56, 128)
block3_conv1: Conv2D | input: (None, 56, 56, 128) | output: (None, 56, 56, 256)
block3_conv2: Conv2D | input: (None, 56, 56, 256) | output: (None, 56, 56, 256)
block3_conv3: Conv2D | input: (None, 56, 56, 256) | output: (None, 56, 56, 256)
block3_pool: MaxPooling2D | input: (None, 56, 56, 256) | output: (None, 28, 28, 256)
block4_conv1: Conv2D | input: (None, 28, 28, 256) | output: (None, 28, 28, 512)
block4_conv2: Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)
block4_conv3: Conv2D | input: (None, 28, 28, 512) | output: (None, 28, 28, 512)
block4_pool: MaxPooling2D | input: (None, 28, 28, 512) | output: (None, 14, 14, 512)
block5_conv1: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
block5_conv2: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
block5_conv3: Conv2D | input: (None, 14, 14, 512) | output: (None, 14, 14, 512)
block5_pool: MaxPooling2D | input: (None, 14, 14, 512) | output: (None, 7, 7, 512)
flatten_1: Flatten | input: (None, 7, 7, 512) | output: (None, 25088)
dense_1: Dense | input: (None, 25088) | output: (None, 4096)
dropout_1: Dropout | input: (None, 4096) | output: (None, 4096)
dense_2: Dense | input: (None, 4096) | output: (None, 4096)
lstm_1_input: InputLayer | input: (None, 30, 384) | output: (None, 30, 384)
lstm_1: LSTM | input: (None, 30, 384) | output: (None, 512)
merge_1: Merge | input: [(None, 512), (None, 4096)] | output: (None, 4608)
dense_3: Dense | input: (None, 4608) | output: (None, 1024)
activation_1: Activation | input: (None, 1024) | output: (None, 1024)
dropout_2: Dropout | input: (None, 1024) | output: (None, 1024)
dense_4: Dense | input: (None, 1024) | output: (None, 1024)
activation_2: Activation | input: (None, 1024) | output: (None, 1024)
dropout_3: Dropout | input: (None, 1024) | output: (None, 1024)
dense_5: Dense | input: (None, 1024) | output: (None, 1024)
activation_3: Activation | input: (None, 1024) | output: (None, 1024)
dropout_4: Dropout | input: (None, 1024) | output: (None, 1024)
dense_6: Dense | input: (None, 1024) | output: (None, 1000)
activation_4: Activation | input: (None, 1000) | output: (None, 1000)

Neuraltalk2

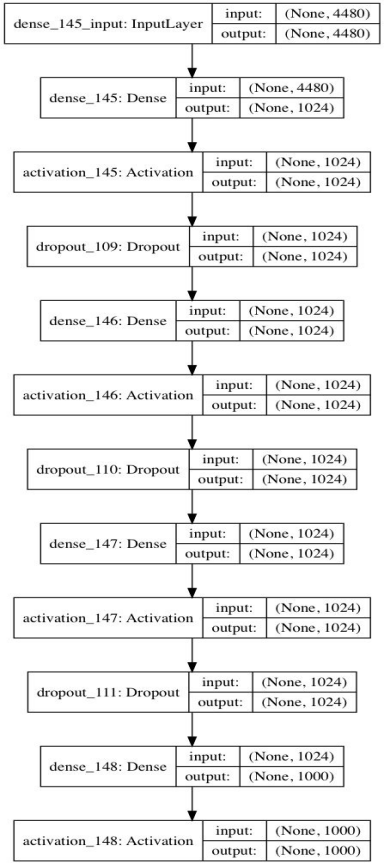
a man is playing tennis on a tennis court

a train is traveling down the tracks at a train station

a cake with a slice cut out of it

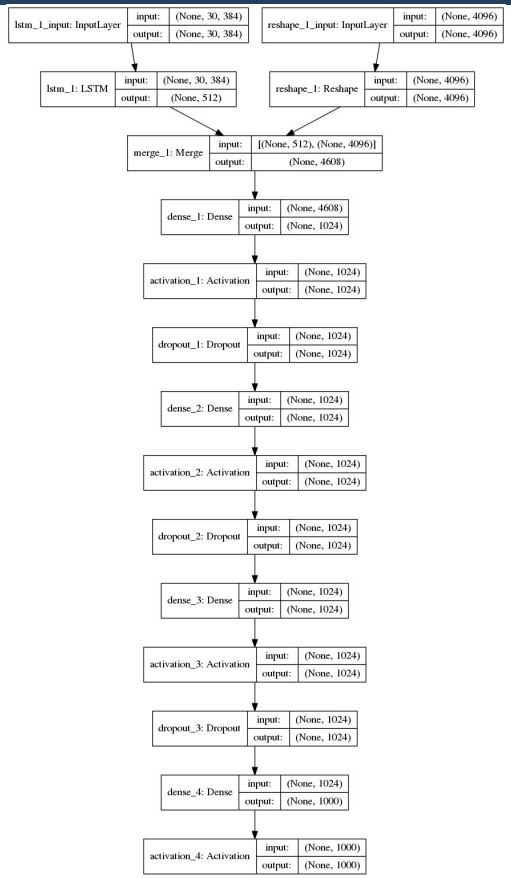
a bench sitting on a patch of grass next to a sidewalk

Model



- Multi-layer-Perceptron(MLP)

- Long-Short-Term-Memory(LSTM)



Training

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

MLP

Epochs number	1	5	10	...	20
Time(min)	36.5	182.5	420	...	1680
Loss(lr = 0.01)	5.25	3.89	3.08	...	3.07

LSTM

Epochs Number	1	5	10	...	20
Time(min)	63	315	630	...	2520
Loss(lr = 0.01)	3.52	3.01	2.90	...	3.02

Accuracy

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Our MLP:24.0%

Our LSTM:36.5%

2017 challenge 1st: 69%

2017 challenge 27th: 37.3%

what color is that train?

```
Loaded
/Users/zack wang/a
  UserWarning)
42.28 %  blue
20.94 %  red
07.67 %  orange
07.64 %  silver
07.37 %  yellow
```

Problems & Future work

THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

- The VGG Model training implement was limited by the computational power.

- The loss of both MLP/LSTM model went smooth at around 3 after 10 epochs.