In [1]:

```
# data augmentation test: rotate different degree (pay attention to adjustable filename etc.)

import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.python.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.python.keras.models import Model
from tensorflow.python.keras import layers, Sequential,losses, metrics

image_height = 48
image_width = 48
emotions_count = 8
emotion_labels = ['neutral', 'happiness', 'surprise', 'sadness',
                  'anger', 'disgust', 'fear', 'contempt']
# !!! change sample size
samples = 99109 # 2~99110
training_samples = 28317*3  # (Training)
validation_samples = 3541*3 # (PublicTest)
test_samples = 3535         # (PrivateTest)

# !!! change npy folder name
image_path = "./dataset_r40/images.npy"
emotion_multi_path = "./dataset_r40/emotions_multi.npy"
emotion_single_path = "./dataset_r40/emotions_single.npy"
images = np.load(image_path)
emotions_multi = np.load(emotion_multi_path)
emotions_single = np.load(emotion_single_path)

# !!! change s/m dataset
#emotions = emotions_single
emotions = emotions_multi

print(images.shape)
print(emotions_multi.shape)
print(emotions_single.shape)
```

```
(99109, 48, 48, 1)
(99109, 8)
(99109, 8)
```

In [2]:

```
cce = losses.CategoricalCrossentropy()
mse = losses.MeanSquaredError()
```

```
tf.config.run_functions_eagerly(True)
def model_acc(y_true, y_pred):
    size = y_true.shape[0]
    acc = 0
    for i in range(size):
        true = y_true[i]
        pred = y_pred[i]
        index_max = tf.argmax(pred).numpy()
        if true[index_max].numpy()==tf.reduce_max(true).numpy():
            acc += 1
    return acc/size
```

In [3]:

```
images = tf.convert_to_tensor(images)
#images = tf.image.grayscale_to_rgb(images)
emotions = tf.convert_to_tensor(emotions)
#images = tf.image.resize(images, [224,224])
images = layers.Rescaling(1./127.5, offset= -1)(images)

training_size = training_samples + validation_samples
test_size = test_samples

training_images = images[:training_size]
test_images = images[training_size:]
training_emotions = emotions[:training_size]
test_emotions = emotions[training_size:]

print("training_images shape:", training_images.shape)
print("training_emotions shape:", training_emotions.shape)
print("test_images shape:", test_images.shape)
print("test_emotions shape:", test_emotions.shape)
```

```
training_images shape: (95574, 48, 48, 1)
training_emotions shape: (95574, 8)
test_images shape: (3535, 48, 48, 1)
test_emotions shape: (3535, 8)
```

In [4]:

```
from tensorflow.python.keras.applications import vgg16, resnet_v2
from tensorflow.python.keras import optimizers
from tensorflow.python.keras.optimizer_v2 import adam
```

In [5]:

```
base_model = vgg16.VGG16(include_top=False,
                         weights="imagenet",
                         input_shape=(48,48,3))
base_model.trainable=True
model = Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(4096, activation='relu'),
    layers.Dense(4096, activation='relu'),
    layers.Dense(emotions_count, activation='softmax'),
])

model.compile(optimizer=adam.Adam(learning_rate=1e-4),
              loss=mse,
              metrics = [model_acc])

model.fit(x=tf.image.grayscale_to_rgb(training_images),
          y=training_emotions,
          batch_size=32,
          epochs=40,
          validation_data=(tf.image.grayscale_to_rgb(test_images), test_emotions))
```

```
C:\Users\Darkl\anaconda3\lib\site-packages\tensorflow\python\data\ops\dataset_ops.py:3703: UserWarning: Even though
the `tf.config.experimental_run_functions_eagerly` option is set, this option does not apply to tf.data functions. T
o force eager execution of tf.data functions, please use `tf.data.experimental.enable.debug_mode()`.
  warnings.warn(
Epoch 1/40
2987/2987 [==============================] - 213s 69ms/step - loss: 0.0315 - model_acc: 0.6578 - val_loss: 0.0211 -
val_model_acc: 0.7489
Epoch 2/40
2987/2987 [==============================] - 220s 74ms/step - loss: 0.0189 - model_acc: 0.7783 - val_loss: 0.0174 -
val_model_acc: 0.7864
Epoch 3/40
2987/2987 [==============================] - 206s 69ms/step - loss: 0.0147 - model_acc: 0.8204 - val_loss: 0.0160 -
val_model_acc: 0.8031
Epoch 4/40
2987/2987 [==============================] - 246s 82ms/step - loss: 0.0119 - model_acc: 0.8524 - val_loss: 0.0145 -
val_model_acc: 0.8245
Epoch 5/40
2987/2987 [==============================] - 204s 68ms/step - loss: 0.0099 - model_acc: 0.8755 - val_loss: 0.0141 -
val_model_acc: 0.8284
Epoch 6/40
2987/2987 [==============================] - 201s 67ms/step - loss: 0.0082 - model_acc: 0.8946 - val_loss: 0.0141 -
val_model_acc: 0.8236
Epoch 7/40
2987/2987 [==============================] - 200s 67ms/step - loss: 0.0070 - model_acc: 0.9103 - val_loss: 0.0140 -
val_model_acc: 0.8326
```

```
Epoch 8/40
2987/2987 [==============================] - 201s 67ms/step - loss: 0.0060 - model_acc: 0.9234 - val_loss: 0.0136 -
val_model_acc: 0.8318
Epoch 9/40
2987/2987 [==============================] - 200s 67ms/step - loss: 0.0053 - model_acc: 0.9322 - val_loss: 0.0134 -
val_model_acc: 0.8313
Epoch 10/40
2987/2987 [==============================] - 201s 67ms/step - loss: 0.0047 - model_acc: 0.9406 - val_loss: 0.0134 -
val_model_acc: 0.8354
Epoch 11/40
2987/2987 [==============================] - 203s 68ms/step - loss: 0.0043 - model_acc: 0.9443 - val_loss: 0.0132 -
val_model_acc: 0.8315
Epoch 12/40
2987/2987 [==============================] - 205s 69ms/step - loss: 0.0038 - model_acc: 0.9500 - val_loss: 0.0136 -
val_model_acc: 0.8351
Epoch 13/40
2987/2987 [==============================] - 204s 68ms/step - loss: 0.0034 - model_acc: 0.9551 - val_loss: 0.0132 -
val_model_acc: 0.8366
Epoch 14/40
2987/2987 [==============================] - 202s 68ms/step - loss: 0.0032 - model_acc: 0.9577 - val_loss: 0.0133 -
val_model_acc: 0.8332
Epoch 15/40
2987/2987 [==============================] - 201s 67ms/step - loss: 0.0029 - model_acc: 0.9617 - val_loss: 0.0128 -
val_model_acc: 0.8428
Epoch 16/40
2987/2987 [==============================] - 201s 67ms/step - loss: 0.0027 - model_acc: 0.9635 - val_loss: 0.0129 -
val_model_acc: 0.8428
Epoch 17/40
2987/2987 [==============================] - 203s 68ms/step - loss: 0.0025 - model_acc: 0.9663 - val_loss: 0.0128 -
val_model_acc: 0.8442
Epoch 18/40
2987/2987 [==============================] - 206s 69ms/step - loss: 0.0024 - model_acc: 0.9666 - val_loss: 0.0130 -
val_model_acc: 0.8447
Epoch 19/40
2987/2987 [==============================] - 203s 68ms/step - loss: 0.0022 - model_acc: 0.9696 - val_loss: 0.0127 -
val_model_acc: 0.8427
Epoch 20/40
2987/2987 [==============================] - 206s 69ms/step - loss: 0.0021 - model_acc: 0.9725 - val_loss: 0.0126 -
val_model_acc: 0.8422
Epoch 21/40
2987/2987 [==============================] - 199s 67ms/step - loss: 0.0021 - model_acc: 0.9705 - val_loss: 0.0132 -
val_model_acc: 0.8399
Epoch 22/40
2987/2987 [==============================] - 199s 67ms/step - loss: 0.0018 - model_acc: 0.9754 - val_loss: 0.0130 -
val_model_acc: 0.8397
Epoch 23/40
```

```
2987/2987 [==============================] - 206s 69ms/step - loss: 0.0017 - model_acc: 0.9783 - val_loss: 0.0126 -
val_model_acc: 0.8387
Epoch 24/40
2987/2987 [==============================] - 200s 67ms/step - loss: 0.0017 - model_acc: 0.9763 - val_loss: 0.0126 -
val_model_acc: 0.8438
Epoch 25/40
2987/2987 [==============================] - 198s 66ms/step - loss: 0.0017 - model_acc: 0.9754 - val_loss: 0.0137 -
val_model_acc: 0.8325
Epoch 26/40
2987/2987 [==============================] - 200s 67ms/step - loss: 0.0015 - model_acc: 0.9790 - val_loss: 0.0130 -
val_model_acc: 0.8385
Epoch 27/40
2987/2987 [==============================] - 199s 67ms/step - loss: 0.0014 - model_acc: 0.9811 - val_loss: 0.0137 -
val_model_acc: 0.8312acc: 0
Epoch 28/40
2987/2987 [==============================] - 198s 66ms/step - loss: 0.0015 - model_acc: 0.9785 - val_loss: 0.0127 -
val_model_acc: 0.8438
Epoch 29/40
2987/2987 [==============================] - 200s 67ms/step - loss: 0.0011 - model_acc: 0.9861 - val_loss: 0.0129 -
val_model_acc: 0.8416
Epoch 30/40
2987/2987 [==============================] - 198s 66ms/step - loss: 0.0013 - model_acc: 0.9812 - val_loss: 0.0126 -
val_model_acc: 0.8438
Epoch 31/40
2987/2987 [==============================] - 195s 65ms/step - loss: 0.0012 - model_acc: 0.9834 - val_loss: 0.0127 -
val_model_acc: 0.8419
Epoch 32/40
2987/2987 [==============================] - 194s 65ms/step - loss: 0.0011 - model_acc: 0.9856 - val_loss: 0.0136 -
val_model_acc: 0.8410
Epoch 33/40
2987/2987 [==============================] - 195s 65ms/step - loss: 0.0011 - model_acc: 0.9837 - val_loss: 0.0140 -
val_model_acc: 0.8368
Epoch 34/40
2987/2987 [==============================] - 195s 65ms/step - loss: 0.0011 - model_acc: 0.9843 - val_loss: 0.0125 -
val_model_acc: 0.8481
Epoch 35/40
2987/2987 [==============================] - 194s 65ms/step - loss: 9.1851e-04 - model_acc: 0.9882 - val_loss: 0.012
8 - val_model_acc: 0.8484
Epoch 36/40
2987/2987 [==============================] - 194s 65ms/step - loss: 8.8873e-04 - model_acc: 0.9880 - val_loss: 0.013
1 - val_model_acc: 0.8396
Epoch 37/40
2987/2987 [==============================] - 194s 65ms/step - loss: 0.0011 - model_acc: 0.9833 - val_loss: 0.0128 -
val_model_acc: 0.8405
Epoch 38/40
2987/2987 [==============================] - 195s 65ms/step - loss: 8.6486e-04 - model_acc: 0.9886 - val_loss: 0.012
```

```
8 - val_model_acc: 0.8430
Epoch 39/40
2987/2987 [==============================] - 194s 65ms/step - loss: 9.7448e-04 - model_acc: 0.9857 - val_loss: 0.013
1 - val_model_acc: 0.8391
Epoch 40/40
2987/2987 [==============================] - 194s 65ms/step - loss: 6.9814e-04 - model_acc: 0.9911 - val_loss: 0.012
9 - val_model_acc: 0.8405
```

Out[5]:

```
<tensorflow.python.keras.callbacks.History at 0x1f700a74d00>
```

In [6]:

```python
image_path = "./dataset_rran/images.npy"
emotion_multi_path = "./dataset_rran/emotions_multi.npy"
emotion_single_path = "./dataset_rran/emotions_single.npy"
images = np.load(image_path)
emotions_multi = np.load(emotion_multi_path)
emotions_single = np.load(emotion_single_path)

# !!! change s/m dataset
#emotions = emotions_single
emotions = emotions_multi

print(images.shape)
print(emotions_multi.shape)
print(emotions_single.shape)
images = tf.convert_to_tensor(images)
#images = tf.image.grayscale_to_rgb(images)
emotions = tf.convert_to_tensor(emotions)
#images = tf.image.resize(images, [224,224])
images = layers.Rescaling(1./127.5, offset= -1)(images)

training_size = training_samples + validation_samples
test_size = test_samples

training_images = images[:training_size]
test_images = images[training_size:]
training_emotions = emotions[:training_size]
test_emotions = emotions[training_size:]

print("training_images shape:", training_images.shape)
print("training_emotions shape:", training_emotions.shape)
print("test_images shape:", test_images.shape)
print("test_emotions shape:", test_emotions.shape)
base_model = vgg16.VGG16(include_top=False,
                         weights="imagenet",
                         input_shape=(48,48,3))
base_model.trainable=True
```