Outline

Deep Learning

Midterm Assignment

P.L.D. Tien (520K0220)

Ton Duc Thang University

March 10, 2023

Outline

Introduction

Overview

ResNet50

ResNet50 (Overfit)

ResNet50 (Vanishing)

CNN

CNN (Overfit)

CNN (Vanishing)

Overall results

End

What is this?

An image captioning system, using ResNet50 and CNN, with independent handling of overfitting and vanishing problems. This submission uses the CIFAR-10[1] for it's dataset.

Note

The snippets of code are coming from the original notebook. It's advised to look at it for a better understanding.

[1] https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz

What does it cover?

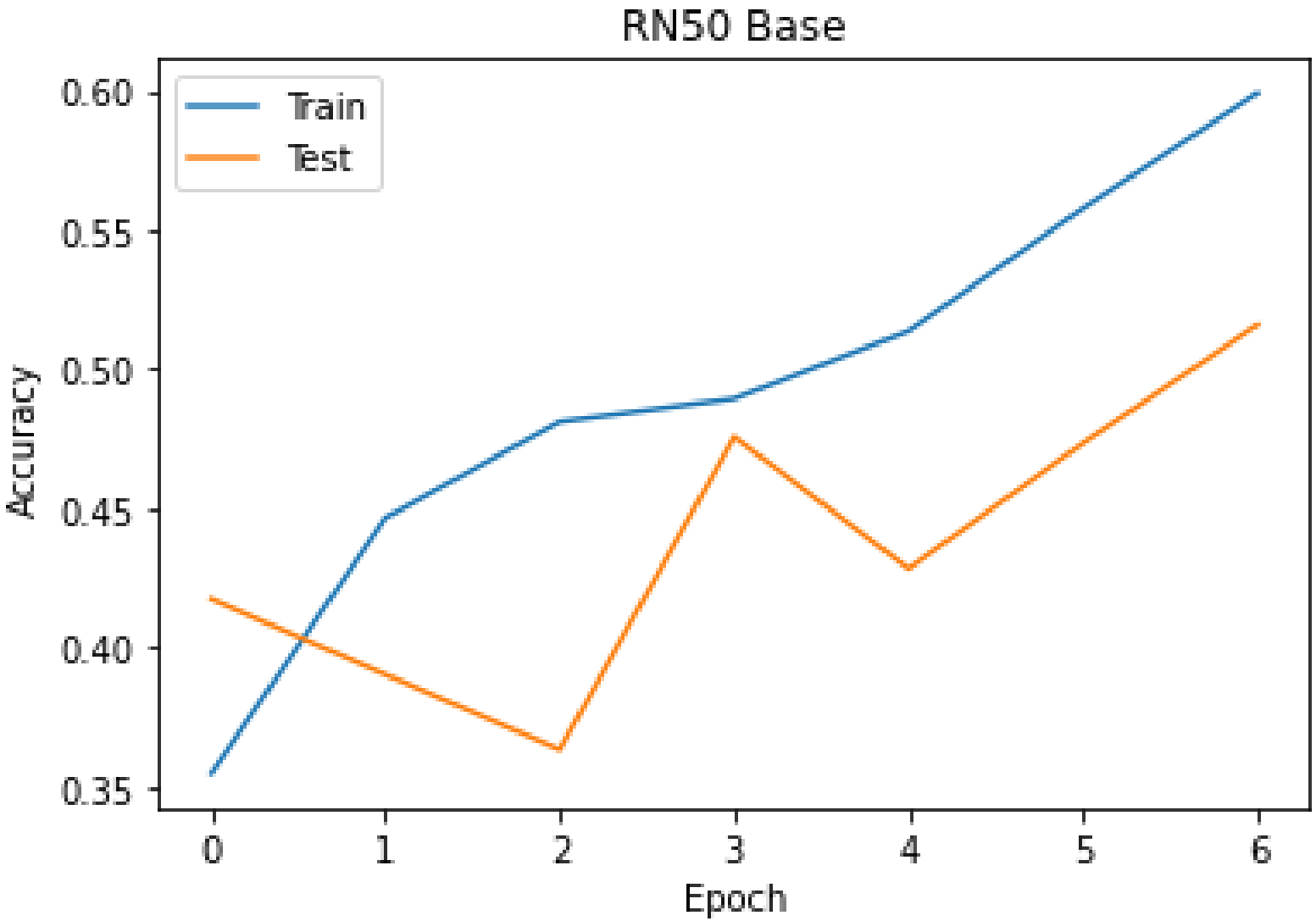
- ResNet50
- ResNet50 (w/ Overfit handling)
- ResNet50 (w/ Vanishing handling)
- CNN
- CNN (w/ Overfit handling)
- CNN (w/ Vanishing handling)

Implementation

Intentionally left blank.

Too much to cover here, read the notebook.

Results

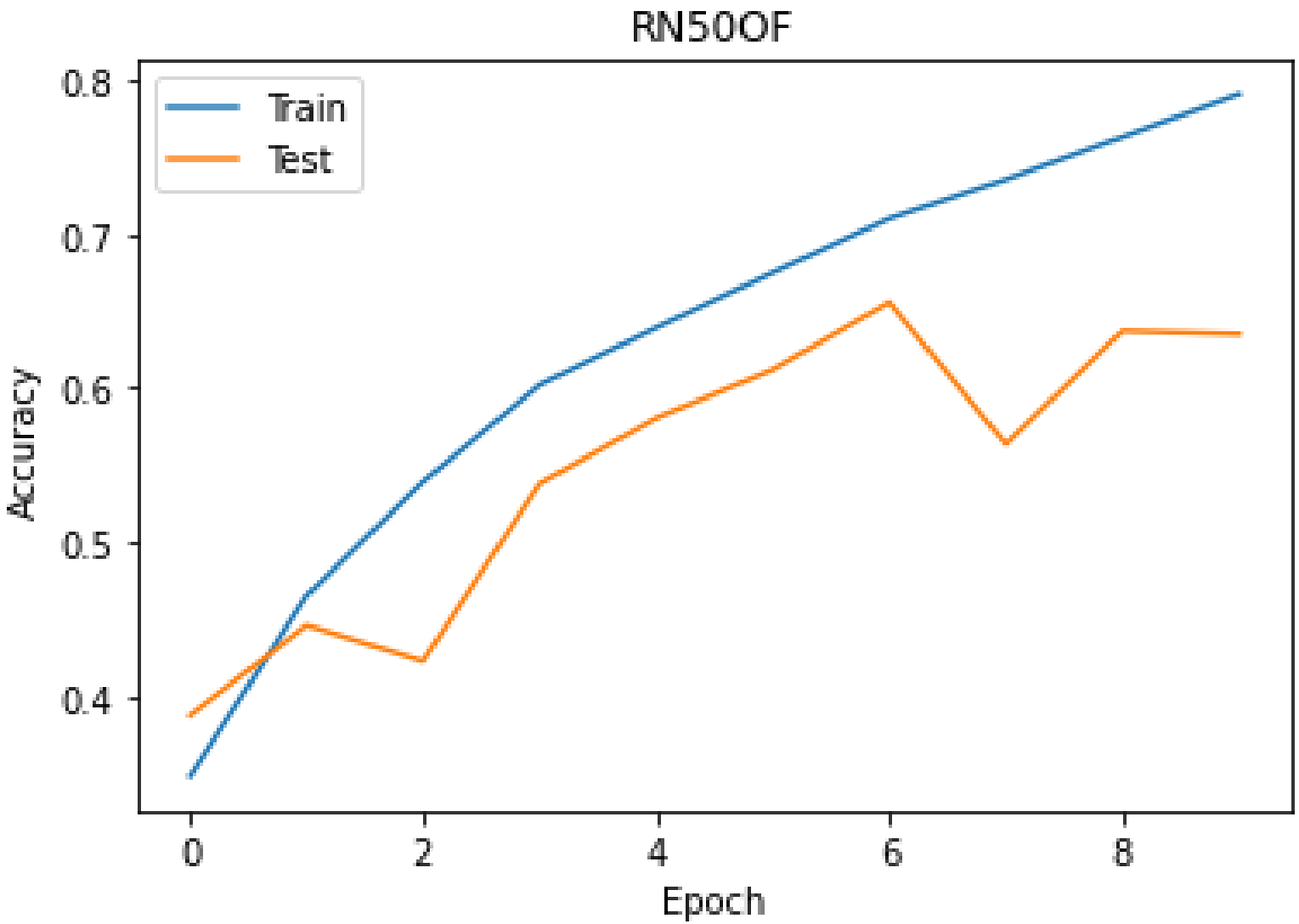

RN50 Base
Train
Test
Accuracy
0.60
0.55
0.50
0.45
0.40
0.35
0
1
2
3
4
5
6
Epoch

Overview

```
39      X = AveragePooling2D(pool_size=(2, 2), padding='same')(X)
40      X = Flatten()(X)
41      #X = Dense(256, activation='relu', name='fc1', kernel_initializer='glorot_uniform')(X)
42      #X = Dense(128, activation='relu', name='fc2', kernel_initializer='glorot_uniform')(X)
43
44      X = Dense(256, activation='relu', name='fc1', kernel_initializer='glorot_uniform'\
45                , kernel_regularizer=regularizers.l2(0.01))(X)
46      X = Dense(128, activation='relu', name='fc2', kernel_initializer='glorot_uniform'\
47                , kernel_regularizer=regularizers.l2(0.01))(X)
```

Replace these lines with regularizers.
Dropout doesn't work here since they're not tensors.

Results

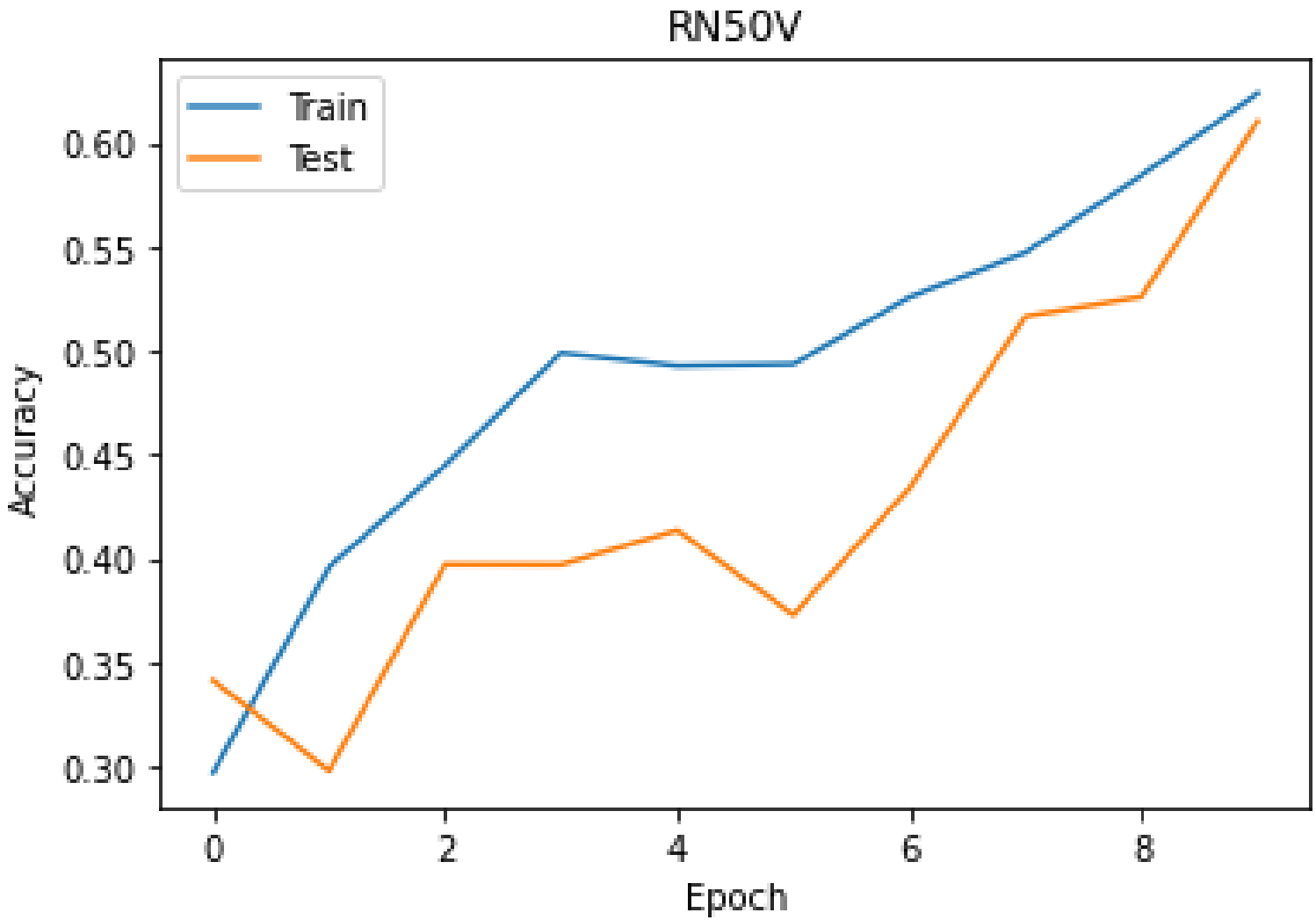


Implementation


```
16    # add batch normalization after ReLU activation
17    X = convolutional_block(X, f=3, filters=[64, 64, 256], stage=2, block='a', stride=1)
18    X = BatchNormalization(axis=3, name='bn_conv2')(X)
19    X = ReLU()(X)
20    X = identity_block(X, 3, [64, 64, 256], stage=2, block='b')
21    X = BatchNormalization(axis=3, name='bn_conv3')(X)
22    X = ReLU()(X)
23    X = identity_block(X, 3, [64, 64, 256], stage=2, block='c')
24    X = BatchNormalization(axis=3, name='bn_conv4')(X)
25    X = ReLU()(X)
26
27    X = convolutional_block(X, f=3, filters=[128, 128, 512], stage=3, block='a', stride=2)
28    X = BatchNormalization(axis=3, name='bn_conv5')(X)
29    X = ReLU()(X)
30    X = identity_block(X, 3, [128, 128, 512], stage=3, block='b')
31    X = BatchNormalization(axis=3, name='bn_conv6')(X)
32    X = ReLU()(X)
33    X = identity_block(X, 3, [128, 128, 512], stage=3, block='c')
34    X = BatchNormalization(axis=3, name='bn_conv7')(X)
35    X = ReLU()(X)
36    X = identity_block(X, 3, [128, 128, 512], stage=3, block='d')
```

We add BatchNormalization after ReLU.

Results

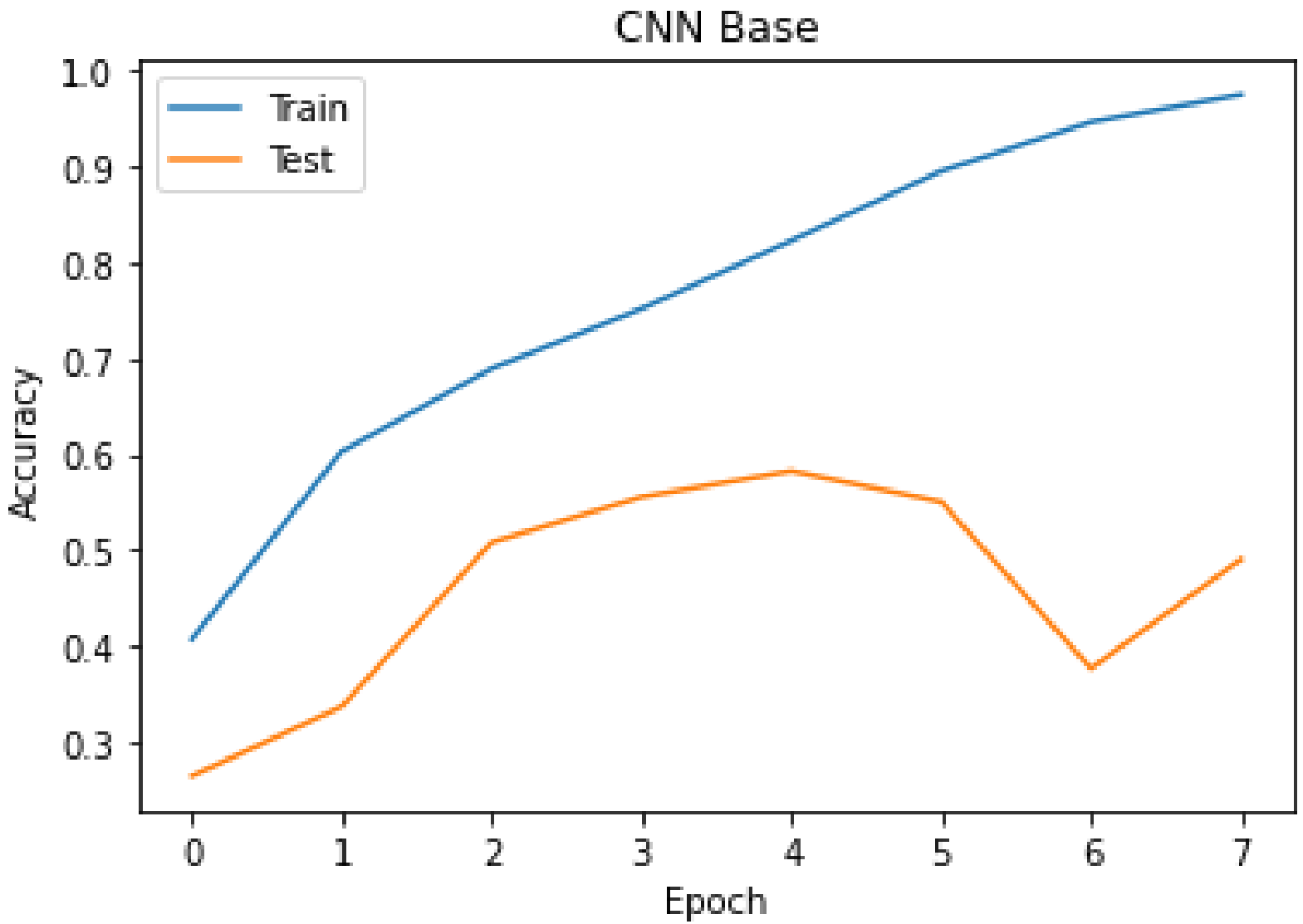


Overview


```
model = models.Sequential(
[

    layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same', input_shape=(32, 32, 3)),
    layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),

    layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),

    layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),

    layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
    layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),

    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.BatchNormalization(),

    layers.Dense(10, activation='softmax')
])
```

Simple enough.

Results

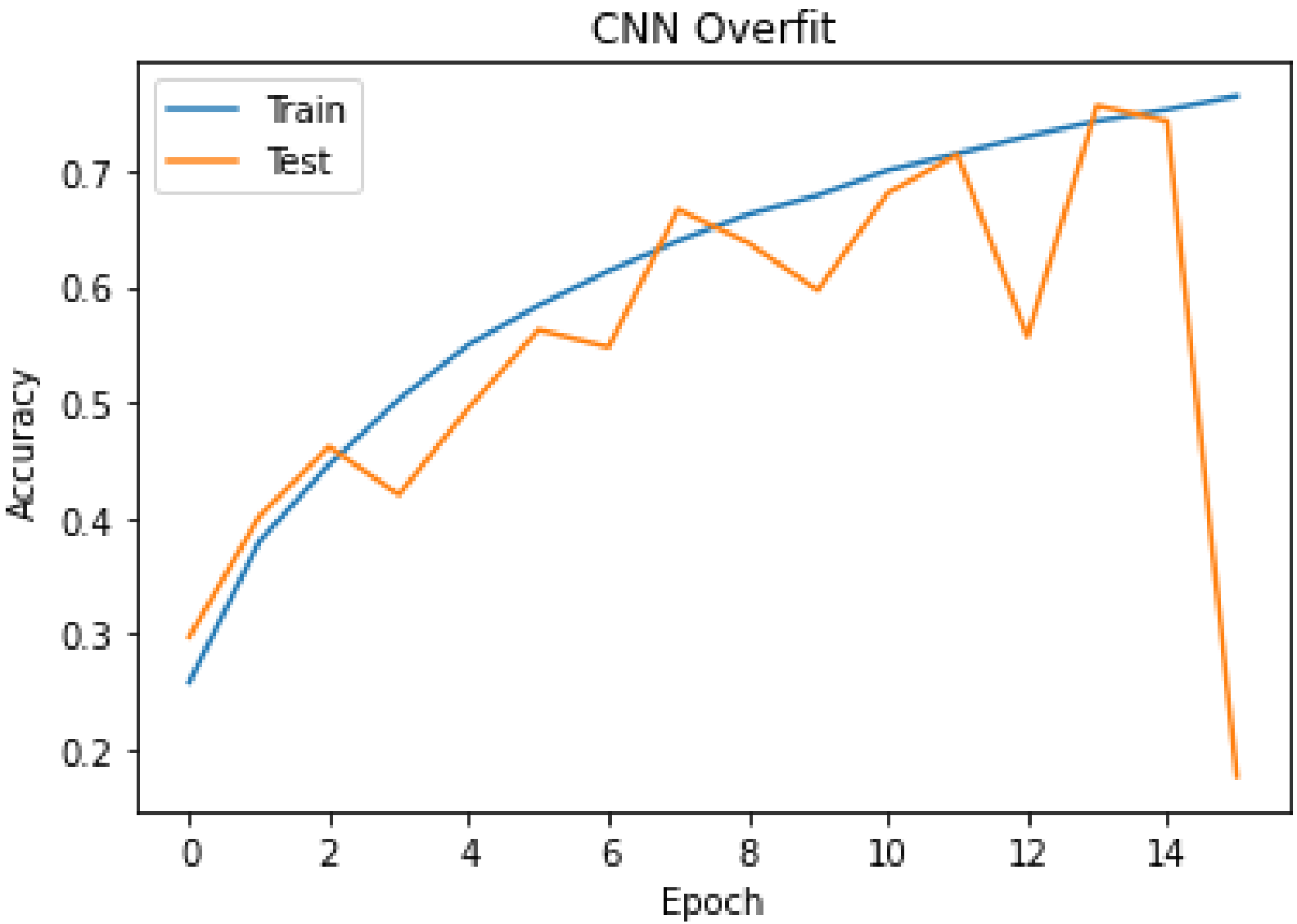


Overview


```
layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same', inp
layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.1),

layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.1),

layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.1),

layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.1),
```

We simply add Dropout in between the blocks.

Results

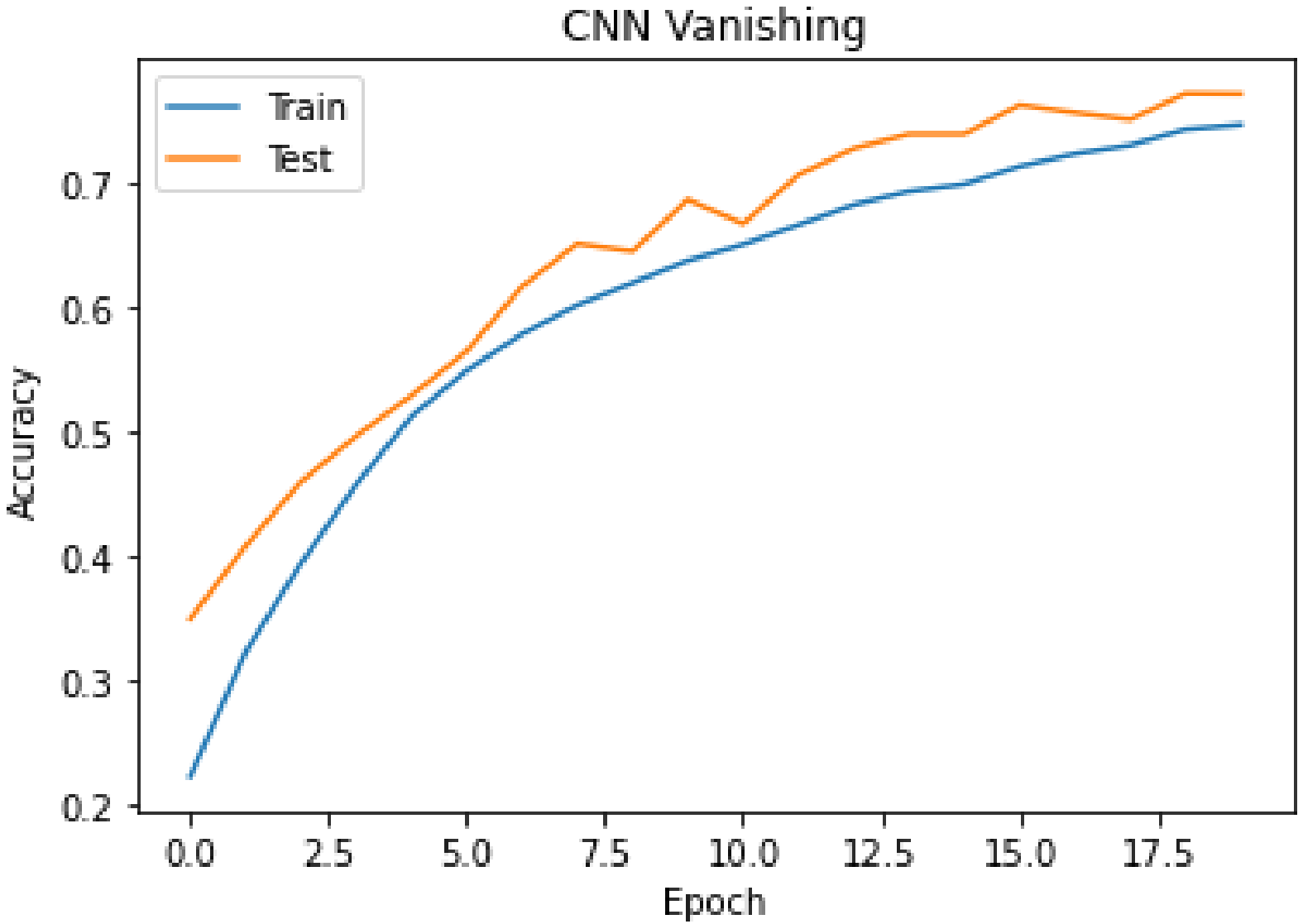


Implementation

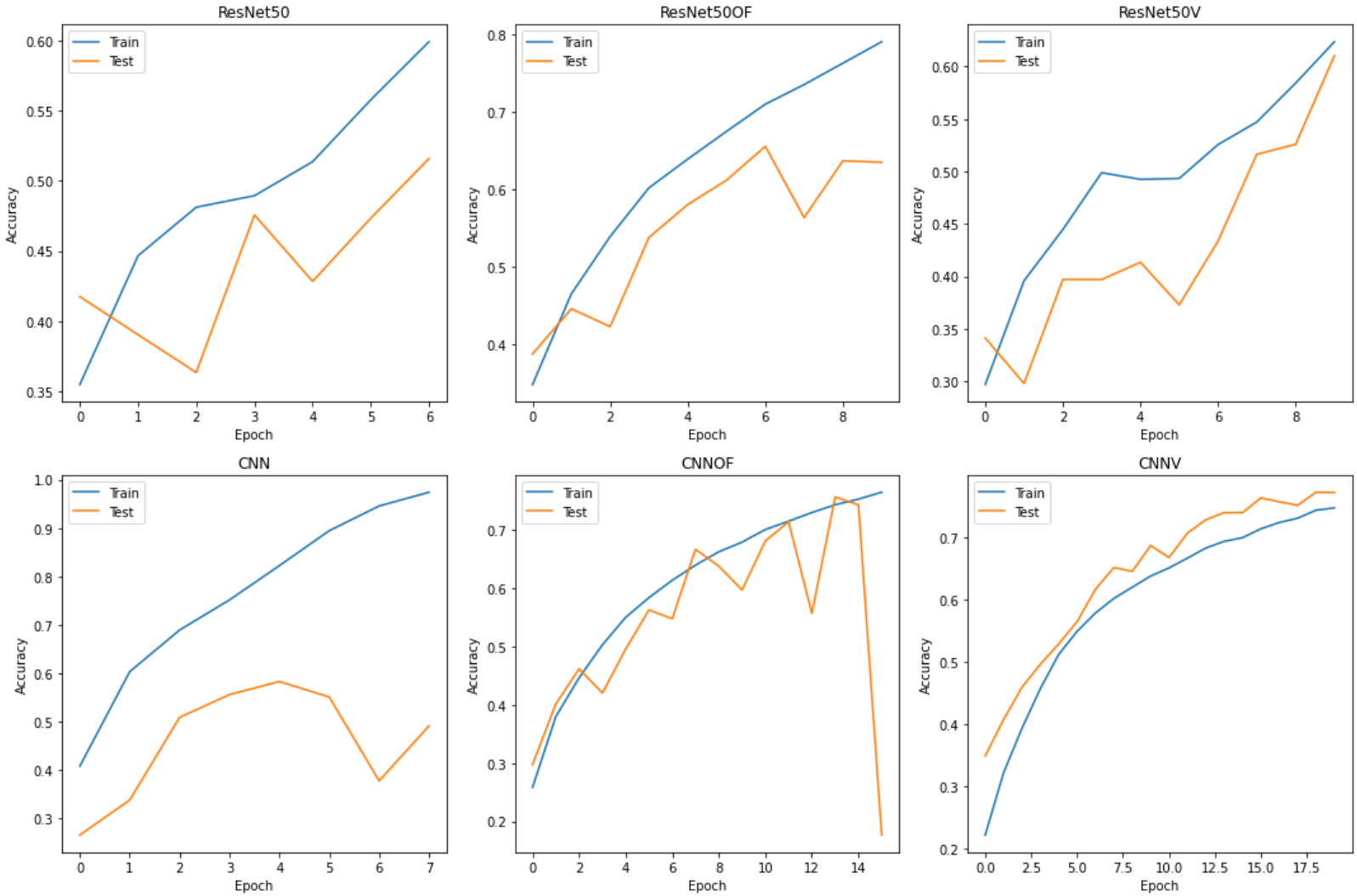

```
layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same', input_shape=(32, 32, 3)),
layers.BatchNormalization(),
layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.2),


layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.3),


layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.4),


layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_uniform', padding='same'),
layers.BatchNormalization(),
layers.MaxPooling2D((2, 2)),
layers.Dropout(0.5),
```

Add BatchNormalization in between Conv2D.

Results



All six of them together



Anything to talk about?