تمرین ۴

سینا اسکندری ۹۷۵۲۱۰۵۴

۱-

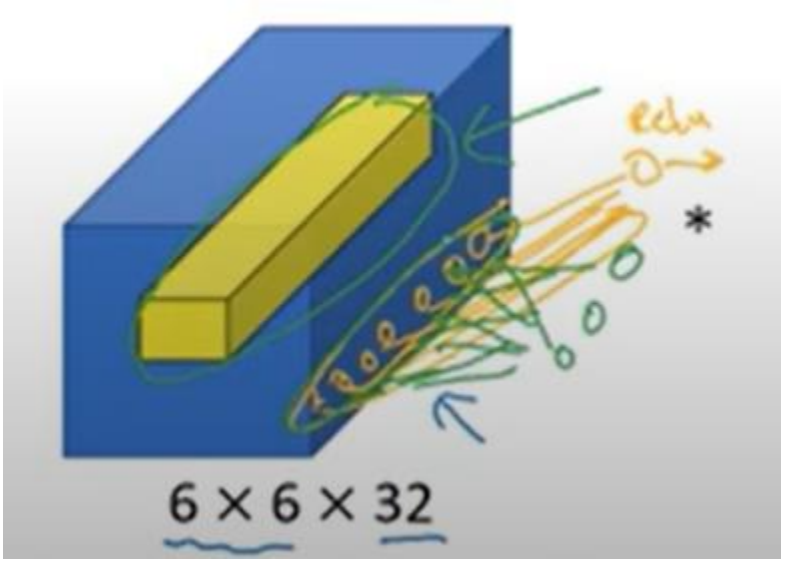
0	20	-70	20	0
0	30	-60	30	0
0	30	-60	30	0
0	30	-60	30	0
0	20	-70	20	0

این کرنل تغییرات را نمایان کرده است و همچنین شبیه مشتق می باشد و لبه یابی می کند.

۲-

کانولوشن ۱ در ۱ با تعداد فیلتر کمتر می تواند باعث شود که بعد سوم لایه بعدی کمتر شود مثلا اگر ورودی یک لایه 28x28x192 باشد و از ۳۲ تا فیلتر ۱ در ۱ استفاده کنیم ابعاد خروجی 28x28x32 می شود. اگر تعداد فیلتر برابر باشد یک لایه غیر خطی بیشتر اضافه می شود که توانایی یادگیری پیچیده تری دارد.

اصطلاح شبکه درون شبکه برای این گفته می شود چون در واقع بردار 1x1x32 از تصویر با یک فیلتر 1x1x32 ضرب می کند که همان کار لایه fully connected می باشد.



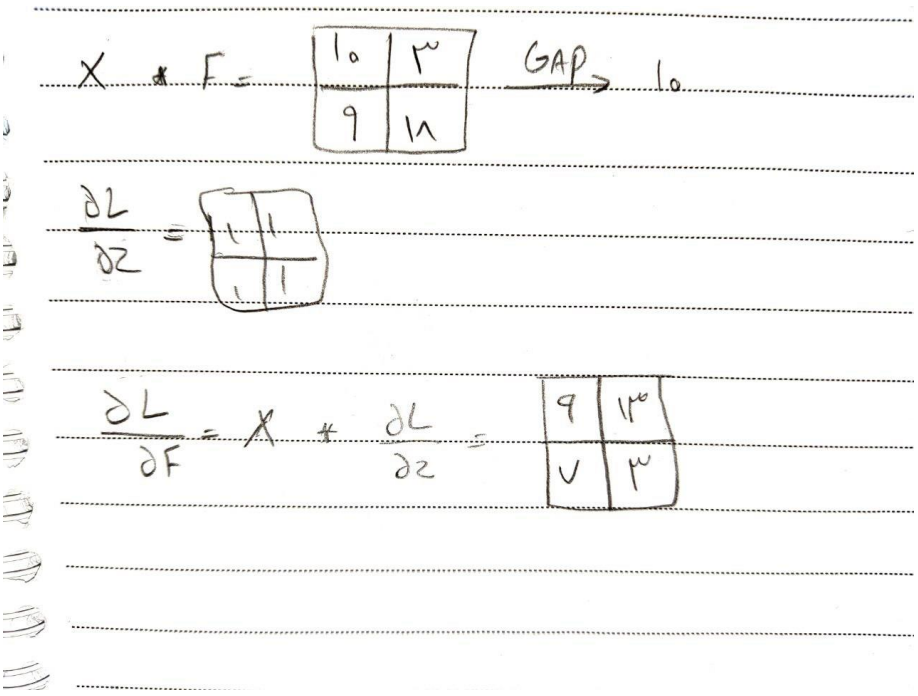
۳-

الف) 14x14x32

ب) 7x7x32

پ) لایه flatten که وزنی ندارد و وزن های لایه Dense آخر چون ۵ نورون دارد: 7845 = 5 * (1 + 32 * 7 * 7)

۴-



۵-

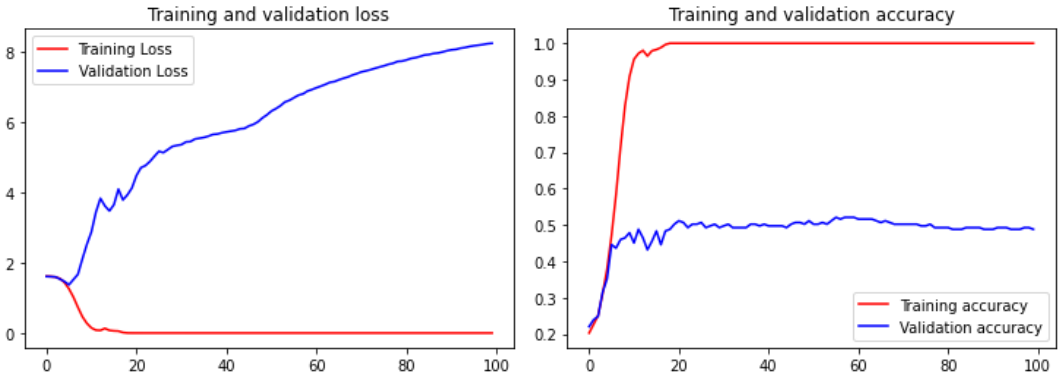
ابتدا ساختار فولدر دیتاست را عوض و دیتا را به نسبت ۰/۸، ۰/۱ و ۰/۱ به train، test و val تقسیم میکنم.

```
classes = ['iranKhodro_dena', 'kia_cerato', 'mazda_3', 'peugeot_206', 'saipa_saina']

train_dir = './train'
os.mkdir(train_dir)
test_dir = './test'
os.mkdir(test_dir)
val_dir = './val'
os.mkdir(val_dir)

train_split = 0.8
test_split = 0.1
val_split = 0.1
for c in classes:
    os.mkdir(os.path.join(train_dir, c))
    os.mkdir(os.path.join(test_dir, c))
    os.mkdir(os.path.join(val_dir, c))
    files = glob.glob(f'dataset/{c}/*')
    total = len(files)
    random.shuffle(files)
    for i in glob.glob(f'dataset/{c}/*')[:int(train_split * total)]:
        shutil.copyfile(i, i.replace(f'dataset/{c}/', f'train/{c}/{c}_'))
    for i in glob.glob(f'dataset/{c}/*')[int(train_split * total):int((1 - test_split) * total)]:
        shutil.copyfile(i, i.replace(f'dataset/{c}/', f'test/{c}/{c}_'))
    for i in glob.glob(f'dataset/{c}/*')[int((1 - test_split) * total):]:
        shutil.copyfile(i, i.replace(f'dataset/{c}/', f'val/{c}/{c}_'))
```

ساختار شبکه:

```
model = Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)))
model.add(layers.MaxPool2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPool2D((2, 2)))

model.add(layers.Conv2D(128, (3, 3), activation='relu', strides=2))
model.add(layers.MaxPool2D((2, 2)))

model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(num_classes, activation='softmax'))

model.summary()

model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

با استفاده از این شبکه پس از ۱۰۰ epoch دقت ها و ضرر ها به شکل زیر می شود.



برای شبکه از پیش آموخته شده ابتدا mobilenet را فریز می شود.

```
conv_base = MobileNetV2(include_top=False,
                        input_shape=(224, 224, 3),
                        weights='imagenet')
conv_base.summary()

model = Sequential()
model.add(conv_base)
model.add(layers.Flatten())
model.add(layers.Dense(256, activation='relu'))
model.add(layers.Dense(num_classes, activation='softmax'))

model.summary()

conv_base.trainable = False
```

با این حالت epoch ۲۰ شبکه را آموزش می دهیم سپس چند لایه آخر را از حالت فریز در میاوریم.

```
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

model.fit(train_generator,
          epochs=20,
          validation_data=validation_generator)

conv_base.trainable = True
set_trainable = False
for layer in conv_base.layers:
    if layer.name == 'block16_expand':
        set_trainable = True

    if set_trainable:
        layer.trainable = True
    else:
        layer.trainable = False
```

دقت و ضرر در شبکه: