general studies with mentor on 18&20th Nov 2022

so as part of a task i wrote a code for resnet50, efficient net bo for high resolution inputs. i did it in a hard coded manner needed to do it in a nested for loop manner with lambda function as value to the keys the tf.applications names.

```
MODELS={"mobilenet:tf.keras.applications.mobilenet.MobileNet()",
.
.
.
}
```

Lambda Layers

they are annonymus function defined without a name. they are one line. they are defined with the key word.
`lambda`
why they are refered to as lambda functions. they are used along wiht other higher order functions that require function objects as argument and these function object are only used for a short term and not used in a wide veriety of places. e.g, used along side map() and filter().

how to use them
`lambda arguments: expression`

a lambda can have ,many argument but only one expresion thats why its one line. on using it the expression get computed and returned. tis used when the function object are returned.

```
double = lambda x: x * 2
```

is same as

```
def double(x):
   return x * 2
```

it is similar to this way we assign the labda of tf.application to the dict key. then we itterate through the dict its input list and save and convert the model.

not like deifned bellow.

```
def initialize_application():
    model01=tf.keras.applications.efficientnet.EfficientNetB0(input_shape=(shape_set[0]),include_top=True,weights=None)
    model11=tf.keras.applications.resnet50.ResNet50(input_shape=(shape_set[0]),include_top=True,weights=None)
    model02=tf.keras.applications.efficientnet.EfficientNetB0(input_shape=(shape_set[1]),include_top=True,weights=None)
    model12=tf.keras.applications.resnet50.ResNet50(input_shape=(shape_set[1]),include_top=True,weights=None)
    model03=tf.keras.applications.efficientnet.EfficientNetB0(input_shape=(shape_set[2]),include_top=True,weights=None)
    model13=tf.keras.applications.resnet50.ResNet50(input_shape=(shape_set[2]),include_top=True,weights=None)
    model00=tf.keras.applications.efficientnet.EfficientNetB0(include_top=True,weights=None)
    model10=tf.keras.applications.resnet50.ResNet50(include_top=True,weights=None)

    model_list=[model01,model02,model03,model00,model11,model12,model13,model10]

    return model_list
```

it should more like be

```
import tensorflow as tf
import numpy as np
import pathlib

MODELS={
    "Resnet50": lambda input_shape :
tf.keras.applications.efficientnet.EfficientNetB0(input_shape=input_shape,in
clude_top=True,weights=None),
    "EfficentNetB0": lambda input_shape :
tf.keras.applications.resnet50.ResNet50(input_shape=input_shape,include_top=
True,weights=None)
}
shape_set=[[1270,720,3], [1920,1080,3],[3840,2160,3]]

def Execute():
    for model_key in MODELS:
        for in_shape in shape_set:
            model=MODELS[model_key](in_shape)

            batch_size=2

            batch=generate_Noise_Data(model.input_shape,batch_size)
            tflite_convert(model,batch)

model.save("./output/{0}_{1}.h5".format(model.name,model.input_shape))

def generate_Noise_Data(shape=(1,28,28,1),batch_size=1):
    if None in shape:
        shape=list(shape)
        shape[0]=batch_size
    noise=np.array(np.random.randint(0,255,shape).astype(np.float32))
    return noise/255
```

```
def tflite_convert(model,data):
    def representative_data_gen():
            for input_value in data:
                input_value = input_value[np.newaxis, ...]
                yield [input_value] # shape should be (1, <data point size))
    converter = tf.lite.TFLiteConverter.from_keras_model(model)
    converter.optimizations = [tf.lite.Optimize.DEFAULT]
    converter.representative_dataset = representative_data_gen
    # Ensure that if any ops can't be quantized, the converter throws an
error
    converter.target_spec.supported_ops =
[tf.lite.OpsSet.TFLITE_BUILTINS_INT8,tf.lite.OpsSet.SELECT_TF_OPS]

    converter.inference_input_type = tf.int8
    converter.inference_output_type = tf.int8

    tflite_model = converter.convert()
    #explore_buffer(model)
    tflite_models_dir = pathlib.Path("./output/tflite_models/")
    tflite_models_dir.mkdir(exist_ok=True, parents=True)
    tflite_model_file =
tflite_models_dir/"model_{0}_{1}.tflite".format(model.name,model.input_shape
)
    tflite_model_file.write_bytes(tflite_model)
```

then manually computed the output at a node point in the resnet50 the add layer when its input shape changed

conv out and pooling out equation:
`1+(input_size - kernal_size+(padding_left +padding_right))/stride`

the pad layer before any conv layer can be seen as padding same.

also observed that the convolution in a network can be of differnet groups when this happens proper tflite file of it supported in our system is hard to build so we identify those layers and then split them channel wise using strided slice and concate them afterwards. widening the network.

make blog about group convolution handling.

wrote a simnple code to load a model and check for groups in them and print count of each in conv sepconv and depthconv

```
mport tensorflow as tf
from collections import Counter
import pdb
import pathlib

CHECK_LIST=(
    str(type(tf.keras.layers.Conv2D(3,(1,1)))),
    str(type(tf.keras.layers.SeparableConv2D(3,(1,1)))),
    str(type(tf.keras.layers.DepthwiseConv2D((1,1))))
)

def get_model(path):
    model=tf.keras.models.load_model(path)
    return model

def check_group(model):
    group_list=[]
    for layer in model.layers:
        if str(type(layer)) in CHECK_LIST:
            #pdb.set_trace()
            group_list.append(layer.groups)
    print("Group set:{0} individual count of groups:
{1}".format(set(group_list),Counter(group_list)),"\nConv layer count for
:",len(group_list))

def execute():
    model=get_model(path="./output/resnext50_32x4d.h5")
    check_group(model)

execute()
```

after this built a dummy network for the multi group models then explored the idea of identifiying sublocks of layers repeating in the model and deifning function blocks to handle them. use only when the pattern in the model is repeating even the basics properties other wise it may become too complex to handle. and going with the default structure lsiting one would have been wiser.

once that was done identified the sub group convs as a seperate block and replaced them with strided slice wrt channel i.e., -1 then layer into group of spead parallel layers feasting on slice of the input channels segments

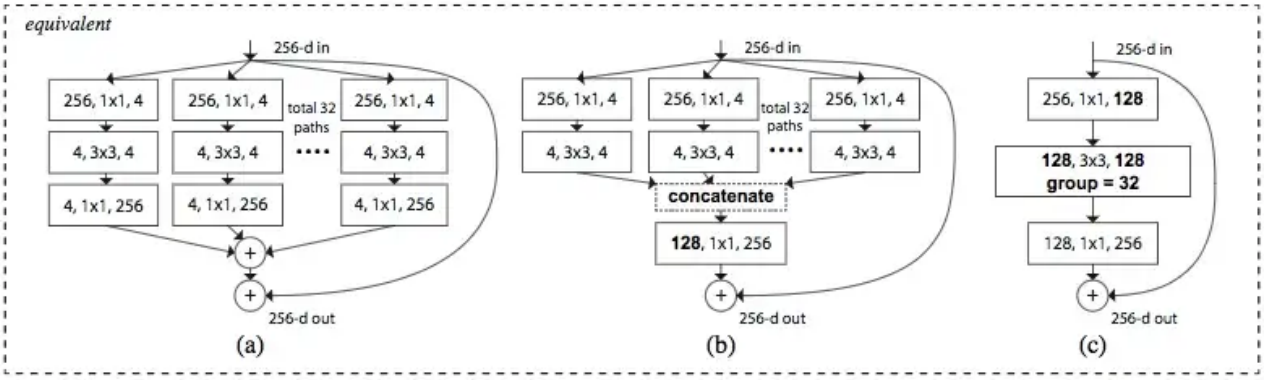

Figure 3. Equivalent building blocks of ResNeXt. **(a)**: Aggregated residual transformations, the same as Fig. 1 right. **(b)**: A block equivalent to (a), implemented as early concatenation. **(c)**: A block equivalent to (a,b), implemented as grouped convolutions [24]. Notations in **bold** text highlight the reformulation changes. A layer is denoted as (# input channels, filter size, # output channels).

here the group gives the number parralel convs there and the replacing layers filter size and channel size corresponds to the segment of the ouput each parallel layer/operation will be contibuting and segment of input the layer will be feasting or working on.

but during the use of tride slice some issue regarding the slice indexing occoured where even when using -1 to identify the last diamensional extreme we are getting slices clipped at (last_index -1) so had to use the input diamensions to the blocks indexed width and height diamensions to get proper clipping which proved to be unclean code. need to explore that further.

on finfishing the model one needs to test it in the gpu sever

use putty

login at the aws sever at provided link with the given credentials
create a work space in our named directiory in the server.

becarefull while using it as chance of affecting everyones code also exist so be extra carefull of the location you are in in the server and what changes you are making.

we have four windows to work in
we can use cretain code to keepcode running in server even when we close the code on our local system
need to learn vim commands to work properly there.
there are apps to view the files in server as a seperate directory of the local.

after the wok is done while commiting be careful to commit only after checking each changed files change with diff and only add for commit them if you are sure of the changes you have made. don't use git add all. once commited note the id for sharing later on to others and later self

git push to brach only not main only after pushing to brach and varifiyng shoud it be merged to main varify request to mentor or overheads. once done merge to main then pull to update our code too with any latest changes.

make sure to ask mentors varification regarding the changes before main merge

once done wiht a task update it in bugzilla as resolved and create the next one

for us its: `new bugs, software, cortiapps v1 linux message save`
also create a issue for ourself in the git hub and assign ourselfves to it and close it on resolve with adequete documentations.

while daily reporting use teams in it note about what all was done and remark the code changes if any to the repo mention any example if in repo and tag-@ the person /s who had instructed you for it. also mention any issues to be carefull off.
personally message those who are urgently in need of your result along side the daily report. make sure to be civil in your conversations.

note while defineing models always specify whcih parameter you are seeting wiht which variable with arugment_name=passing_varibale

19112022

---

go today defined the dummy network of efficent net lite 0 int 8 using he methodology discussed before. so identified the repeting layers of **conv relu deep conv relu and conv** also saw a repeating branched add just like in the resnet 50. so seperate blocks were defined for them then remaining layers implemented as is. but observed that their parameters were differnent.

explored logics for handling it like dictionarizing the encounterd parameters and then using them when and where needed.

insted settled on the loginc tpo pas the parameters as a list of dicionaries of filters kernel size and stride along with padding

used one line if else

```
expression_1 if conditon else expression_2
```

thought about using mulitpliers but that proved to be bit more complicated so settled on list.

the dummy h5 was created.

learned that if we change the input the output of pooling layers as well as all the layers will change. so reshape layers used inbetween will need their target spec change.

using gloobal average pooling instead of average pooling helps collaps the spatial diamension.

i.e., for an input of 1,8,8,32 we need avg pooling of window size 8x8 to collapse saptial diamensions but if input gets changed then the spatial diamesions won't get properly collapsed. in such case global average pooling naturally collapses the spatial diamensions.

My Wish list to be done

need to refer yolo v5
make keras analyser tha make excell of all layers in a model and their arrtibute lists also get macc and memmory usage
same for tflite.

aslo another script that gets the excel of all dir results of a command to be later explored by me.

learn the tmux command sheet to operate in it

pathlib path dir