МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Математическая кибернетика и информационные технологии»

Отчет по лабораторной работе №4

по дисциплине «Системы машинного зрения»

Выполнил:

студент группы БВТ2001

Степанов М.Н.

Оглавление

[Цель работы 3](#_Toc153003983)

[Задание 3](#_Toc153003984)

[Требования к выполнению работы 3](#_Toc153003985)

[Ход работы 4](#_Toc153003986)

[Вывод 7](#_Toc153003987)

[Приложение 7](#_Toc153003988)

# Цель работы

Разработать алгоритм, используя язык python, реализующий работу функции upsamling.

# Задание

При выполнении лабораторной работы необходимо:

1. ознакомиться с описанием операции библиотеки PyTorch (https://pytorch.org/docs/stable/generated/torch.nn.Upsample);
2. используя язык программирования Python написать алгоритм, реализующий алгоритм функции upsampling по методу билинейной интерполяции;
3. составить отчет по лабораторной работе;

# Требования к выполнению работы

1. Должна быть реализована функция upsampling по методу билинейной интерполяции. Входные параметры функции должны соответствовать входным параметрам функции Upsampling библиотеки PyTorch;
2. Должны быть разработаны группы тестов, для проверки реализованного функционала;
3. Составить отчет по лабораторной работе, включающий краткое описание выполненной работы, ссылки на место хранения разработанного алгоритма и существенные замечания, возникшие в ходе выполнения работ;

# Ход работы

В ходе лабораторной работы была реализована работа функции для транспонированной свёртки. За основу была взята функция свёртки upsample библиотеки pytorch. Опираясь на документацию библиотеки, была реализована функция upsample, принимающая те же аргументы, что и оригинальная функция: input, size, scale\_factor, mode, но не принимающая align\_corners. Реализация данной функции представлена ниже:

def custom\_bilinear\_upsample(

input: Tensor,

new\_size: tuple[int, int] | None = None,

scale\_factor: float | tuple[float, float] | None = None

) :

if scale\_factor is not None and type(scale\_factor) is not tuple:

scale\_factor = (scale\_factor, scale\_factor)

if scale\_factor is not None:

print(scale\_factor)

new\_size = (round(input.shape[-2] \* scale\_factor[0]), round(input.shape[-1] \* scale\_factor[1]))

if new\_size is not None:

scale\_factor = (new\_size[0] / input.shape[-2], new\_size[1] / input.shape[-1])

print(new\_size)

batch, channels, input\_h, input\_w = input.shape

resized = torch.zeros((batch, channels, new\_size[0], new\_size[1]))

for h in range(new\_size[0]):

for w in range(new\_size[1]):

x = h / scale\_factor[0]

y = w / scale\_factor[1]

x\_floor = floor(x)

x\_ceil = min(input.shape[-2]-1, ceil(x))

y\_floor = floor(y)

y\_ceil = min(input.shape[-1]-1, ceil(y))

if x\_ceil == x\_floor and y\_ceil==y\_floor:

i = input[:, :, int(x), int(y)]

elif (x\_ceil == x\_floor):

i1 = input[:, :, int(x), int(y\_floor)]

i2 = input[:, :, int(x), int(y\_ceil)]

i = i1 \* (y\_ceil - y) + i2 \* (y - y\_floor)

elif y\_ceil == y\_floor:

i1 = input[:, :, int(x\_floor), int(y)]

i2 = input[:, :, int(x\_ceil), int(y)]

i = (i1 \* (x\_ceil - x)) + (i2 \* (x - x\_floor))

else:

v1 = input[:, :, x\_floor, y\_floor]

v2 = input[:, :, x\_ceil, y\_floor]

v3 = input[:, :, x\_floor, y\_ceil]

v4 = input[:, :, x\_ceil, y\_ceil]

i1 = v1 \* (x\_ceil-x) + v2 \* (x-x\_floor)

i2 = v3 \* (x\_ceil-x) + v4 \* (x - x\_floor)

i = i1 \* (y\_ceil - y) + i2 \* (y - y\_floor)

resized[:, :, h, w] = i

return resized

В данной функции были реализованы: input – входной тензор, которому требуется изменить размер; size – финальный размер тензора; scale\_factor – коэффициент, на который нужно увеличить тензор. Для корректной работы должен быть заполнен либо параметр size, либо параметр scale\_factor. Параметр align\_corner не был реализован, поэтому функция неполностью повторяет оригинал, что нужно учитывать при использовании.

Ниже, на рисунке 1, представлен пример применения реализованной функции на реальном изображении очень низкого разрешения.



Рисунок 1 — Применение функции к изображению.

После чего с помощью библиотеки pytest были реализованы тесты, отвечающие за проверку корректности выполненной работы.

Далее приведены сами тесты:

%%ipytest

def test\_1():

inputs = torch.rand(1, 1, 2, 2)

scale\_factor = 2

reference = F.upsample(inputs, scale\_factor=scale\_factor, mode='bilinear', align\_corners=True)

subject = custom\_bilinear\_upsample(inputs, scale\_factor=scale\_factor)

print(reference)

print(subject)

assert torch.any(torch.eq(reference, subject))

def test\_2():

inputs = tensor\_img

scale\_factor = 1.5

reference = F.upsample(inputs, scale\_factor=scale\_factor, mode='bilinear', align\_corners=True)

subject = custom\_bilinear\_upsample(inputs, scale\_factor=scale\_factor)

print(reference)

print(subject)

assert torch.any(torch.eq(reference, subject))

def test\_3():

inputs = torch.rand(2, 3, 6, 6)

new\_size = [27, 27]

reference = F.upsample(inputs, size=new\_size, mode='bilinear', align\_corners=True)

subject = custom\_bilinear\_upsample(inputs, new\_size=new\_size)

print(reference)

print(subject)

assert torch.any(torch.eq(reference, subject))

Сравнение результатов изначальной функции и функции библиотеки pytorch было проведено с помощью функционала этой же библиотеки, а именно функции allclose, отвечающей за приблизительное сравнение полученных обоими функциями значений, т.к. полная эквивалентность невозможна.

Все 4 реализованных теста прошли успешно, что можно увидеть на рисунке 1. Предупреждение получено из-за устаревания используемой функции.

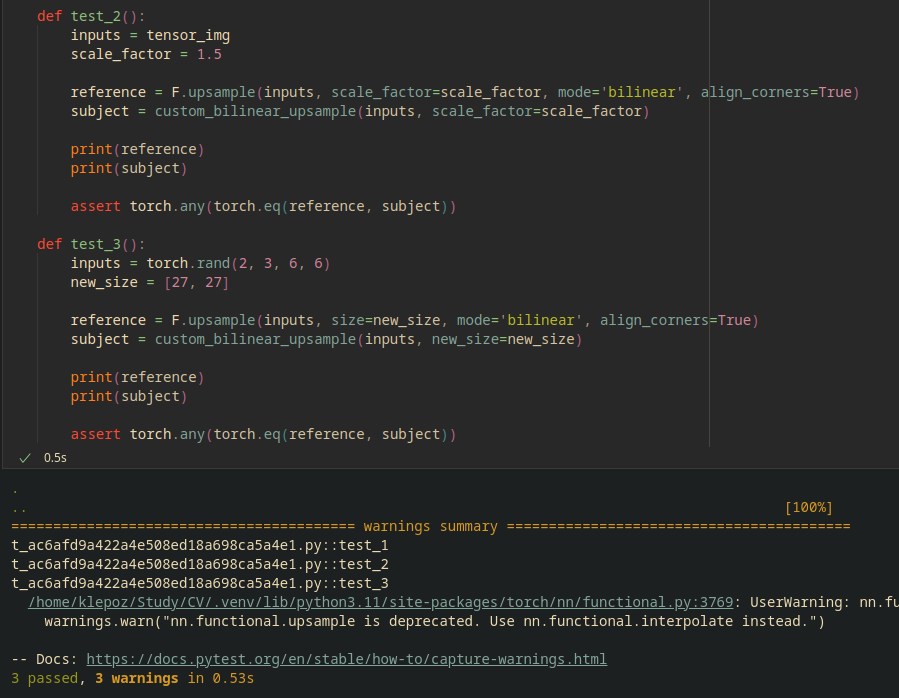


Рисунок 2 — Результат работы тестов.

# Вывод

В ходе работы была реализована функция upsample, включающая в себя большую часть возможностей оригинала. Кроме того, были проведены тесты с успешным результатом, что означает корректность работы реализованного функционала.

# Приложение

Ссылка на репозиторий гитхаба с реализованным функционалом: <https://github.com/d1agnozzz/CV_labs>