**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 математической кибернетики и информационных технологий

**Отчет по лабораторной работе №1**

по дисциплине «Системы машинного зрения»

на тему:

«Разработка нейросетевых функций. Операция Convolution2D»

Выполнил:

Студент группы БВТ2002

Дерябин С. С.

Содержание:

1 Введение….……………………………………………………………….3

2 Выполнение……………………………………………………………….4

3 Вывод………………………………………………………...……………6

**Введение**

В данной лабораторной работе требуется реализовать встроенную в библиотеку PyTorch функцию 2D-свертки Convolution 2D, а также сравнить готовую реализацию функции и написанную в рамках лабораторной работы с помощью группы тестов. Входные параметры функции должны совпадать с готовой реализацией.

Функция torch.nn.Conv2d в библиотеке PyTorch представляет собой операцию свертки (convolution) для двумерных данных, таких как изображения. Эта функция используется в глубоком обучении для создания сверточных нейронных сетей. Вот принцип работы и входные параметры этой функции:

Принцип работы:

* Входные данные (Input): Conv2d принимает на вход четырехмерный тензор: (N, C, H, W), где
* N - размер пакета (batch size),
* C - количество каналов (например, для RGB изображений C=3),
* H - высота изображения,
* W - ширина изображения.

Ядро (Kernel): Conv2d также принимает ядро (или фильтр) для свертки. Ядро - это небольшой тензор, который скользит по входному изображению и вычисляет взвешенную сумму значений входных пикселей в соответствии с весами ядра.

Шаг (Stride) и Заполнение (Padding): Можно задать параметры stride (шаг) и padding (заполнение), чтобы контролировать, как ядро перемещается по входному изображению. Stride определяет, насколько смещается ядро при каждой свертке, а Padding добавляет нулевые значения вокруг входных данных, чтобы учесть краевые эффекты.

Выходные данные (Output): Результатом операции свертки является новый тензор с формой (N, C\_out, H\_out, W\_out), где

* C\_out - количество выходных каналов (зависит от количества ядер),
* H\_out и W\_out - высота и ширина выходного изображения.

**Выполнение**

Согласно заданию лабораторной работы, мною было реализовано два Python файла. Первый – fu\_custom\_cov2d.py, представляет собой кастомную реализацию функции Conv2d, которая принимает на вход:

* input\_tensor – входной тензор,
* weight - ядро свертки,
* stride - шаг свертки,
* padding - значение паддинга для входного тензора,
* bias: флаг указывающий, должен ли использоваться смещение (по умолчанию True).

Листинг кода представлен на рисунке 1.

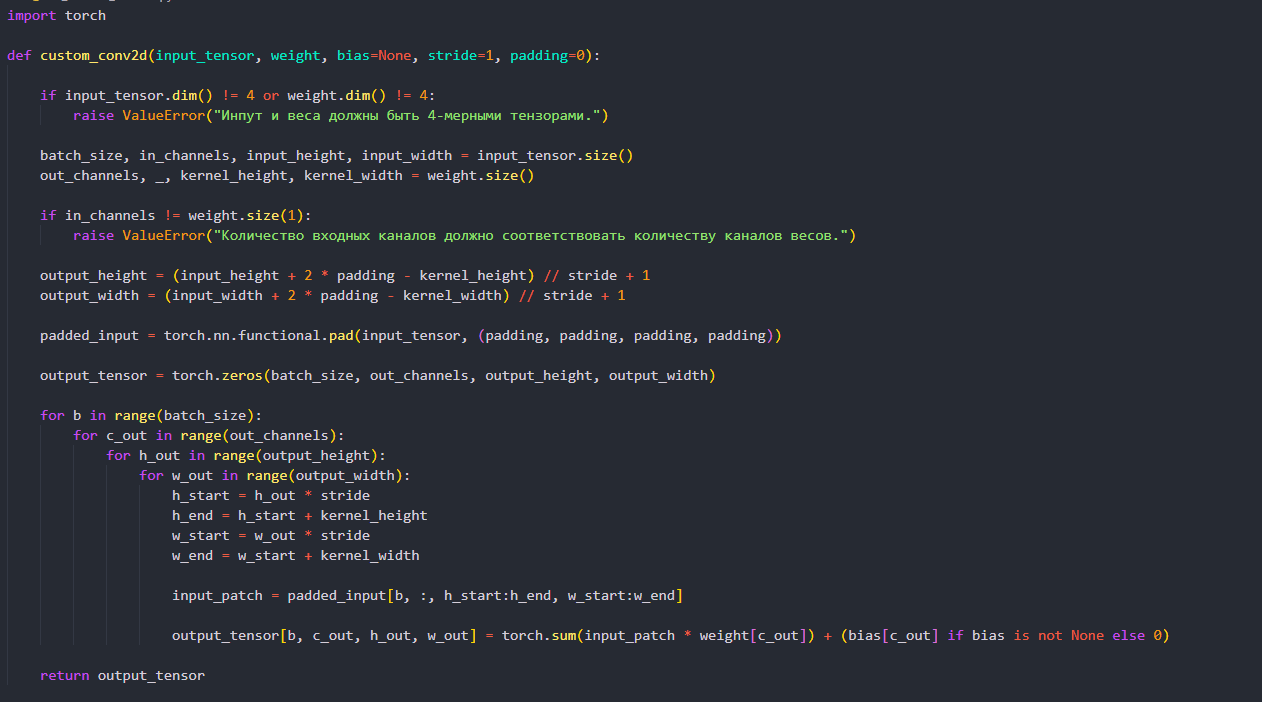


Рисунок 1 – реализация функции custom\_cov2d

Также для тестирования правильной работы функции были написаны unit-тесты, которые хранятся в файле – lab1.py. Структура тестов:

1. Первый тест описан для тестирования общего функционала на примерах 2-ух случайных тензеров.
2. Второй тест описан для тестирования функционала кастомной свертки без использования bias.
3. Третий тест описан для тестирования передачи больших данных на вход функции.

Листинг кода с тестами представлен на рисунке 2.

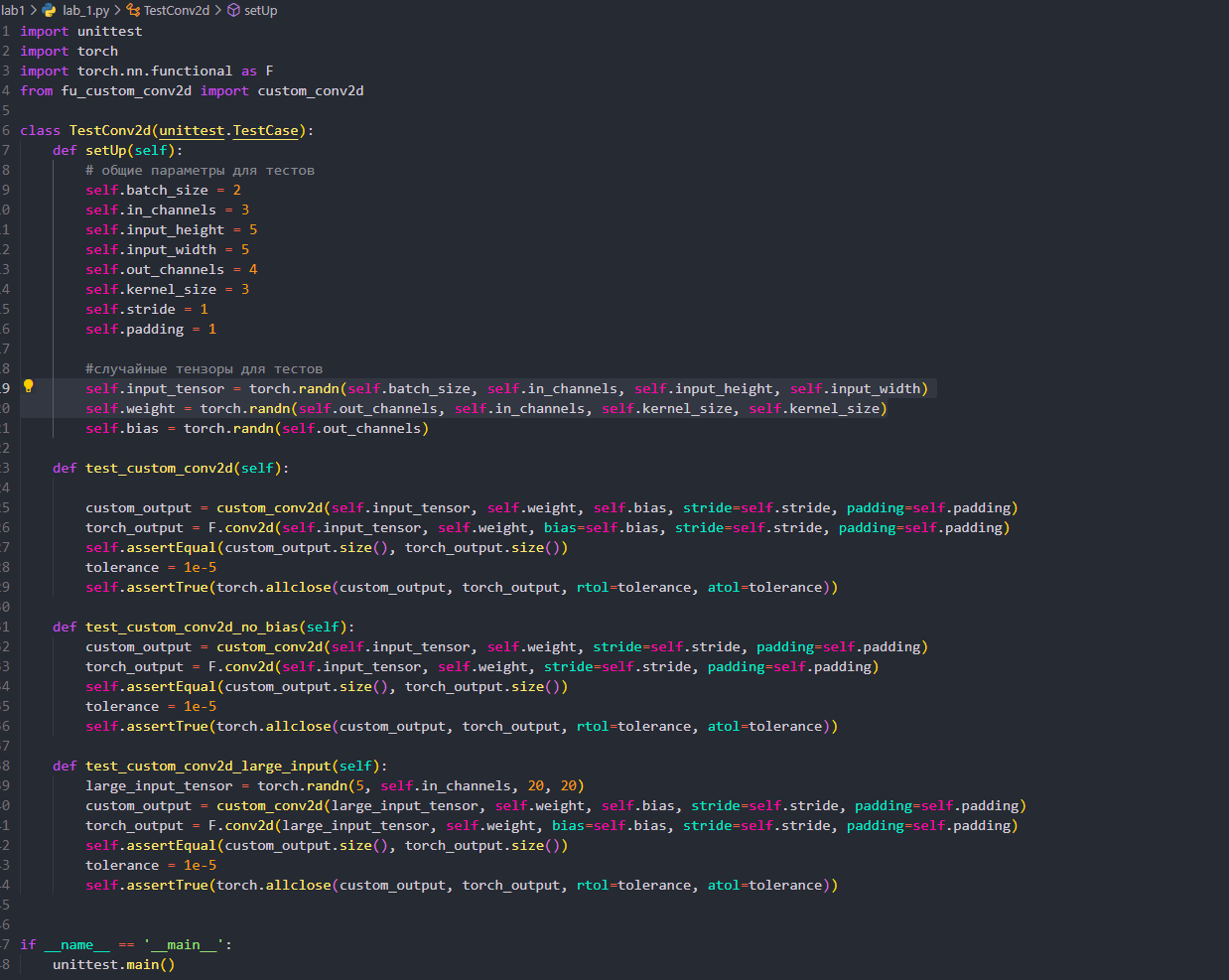
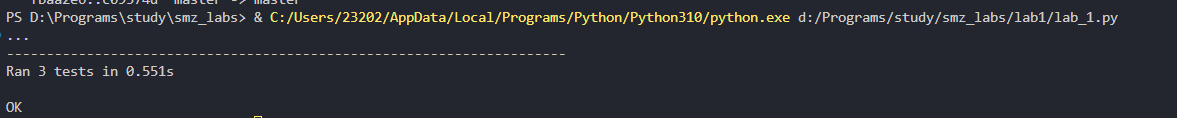


Рисунок 2 – реализация unit-тестов

Успешное прохождение тестов представлено на рисунке 3.



Ссылка на код лабораторной работы: <https://github.com/vraniye/smz_labs/tree/master/lab1>

**Вывод**

В ходе выполнения данной лабораторной работы мной была реализована функция custom\_cov2d, которая, как и оригинальная функция conv2D из библиотеки PyTorch, осуществляет операцию двумерной свёртки. Был произведён ряд тестов, в котором, в результате сравнения результата работы данных функций на одних и тех же данных, мной была выявлена корректность работы реализованной мной функции Convolution2D.