МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОННОЙ ТЕХНИКИ

Институт системной и программной инженерии

и информационных технологий (Институт СПИНТех)

**Лабораторный практикум по курсу**

**"Свёрточные нейронные сети в компьютерном зрении"**

**(02/20 – 06/20)**

Лабораторная работа 5

Свёрточные нейронные сети

На этом занятии компьютерного практикума вы изучите, что такое свёрточные нейронные сети и как они работают. В частности, мы поговорим:

* о биологии – о зрительной коре головного мозга;
* о том, что такое свёртки и как они «работают»;
* подробно рассмотрим принцип работы свёрточных сетей;
* разработаем свёрточную сеть для распознавания цифр.

Прежде чем приступить собственно к программированию, настоятельно рекомендуется ознакомиться с материалом лекций, дополнительными материалами, имеющими отношение к свёрточным нейронным сетям, функциям активации нейронов, глубокому обучению, а также с языком программирования Python и основами работы с Jupyter Notebook, подробно рассмотренными в первом компьютерном практикуме, и с основами работы с библиотекой глубокого обучения TensorFlow, подробно рассмотренными в третьем компьютерном практикуме данного курса.

Файлы, включенные в задание:

*convolutional\_network.ipynb* – программа, реализующая пошаговое обучение свёрточной нейронной сети на примере классической задачи распознавания рукописных цифр.

Итак, в этом компьютерном практикуме вы познакомитесь с весьма широким классом архитектур, а именно, свёрточными нейронными сетями, и научитесь создавать и обучать их для решения классической задачи распознавания рукописных цифр с помощью языка программирования Python, библиотеки общего назначения TensorFlow, и блокнота Jupyter. В этой лабораторной работе вы будете в соответствии с заданием дописывать блокнот *convolutional\_network.ipynb* по инструкциям упражнений.

## **Зрительная кора головного мозга**

Свёрточные нейронные сети (*convolutional neural networks, CNN*) – это весьма широкий класс архитектур, основная идея которых состоит в том, чтобы переиспользовать одни и те же части нейронной сети для работы с разными, маленькими, локальными участками входов. В наши дни уже нашлось много самых разнообразных применений свёрточным сетям, но основной задачей, ради которой люди когда-то придумали свёрточные сети, остаётся *обработка изображений*.

И это не случайно: идея свёрточных сетей во многом мотивирована исследованиями о зрительной коре головного мозга.

Люди давно задумывались о том, как именно мы видим окружающий мир, как работает зрение? Но нас, конечно, в данном практикуме интересует не глаз, а то, что потом происходит с прочитанным изображением.

Зрительный нерв – толстый пучок аксонов ганглионарных клеток, по которому информация с сетчатки доходит до мозга. Зрительный нерв входит в таламус, отдел мозга, обрабатывающий информацию от органов чувств, там первичная обработка происходит в так называемом латеральном коленчатом теле (lateral geniculate nucleus, LGN), а затем зрительная информация от LGN поступает собственно в зрительную кору (visual cortex).

Наша цель сейчас отметить несколько важных черт в устройстве зрительной коры, которым можно найти некоторые условные соответствия в архитектуре современных нейронных сетей.

Итак, зрительная кора, как ни странно, расположена сзади, в затылочной доле головного мозга. Она делится на несколько частей:

* *зрительная зона* V1 (visual area one) или *стриарная кора*: здесь выделяются локальные признаки небольших участков считанного с сетчатки изображения; это для нас сейчас самое интересное!
* *зрительная зона* V2 (visual area two): продолжает выделять локальные признаки, слегка обобщая их и добавляя бинокулярное зрение (то есть стереоэффект от двух глаз);
* *зрительная зона* V3: распознаёт цвет, текстуры объектов, появляются первые результаты их сегментации и группировки;
* *зрительная зона* V4: начинает распознавать геометрические фигуры и очертания объектов, пока несложных;
* *зрительная зона* V5: занимается распознаванием движений;
* *зрительная зона* V6: обобщает данные о всей картинке;
* *зрительная зона* V7: распознавание сложных объектов (например, человеческих лиц).

Такая функциональная специализация хорошо соответствует тому. Что мы обычно видим в глубоких нейронных сетях: более высокие уровни нужны для того, чтобы выделять более общие признаки, соответствующие абстрактным свойствам входа, а на нижних уровнях признаки более конкретные.

Кроме того, между нейронами в головном мозге всегда присутствует очень сильная обратная связь от более высоких уровней к более низким. В искусственных нейронных сетях тоже вводят подобные механизмы внимания, помогающие учесть какие именно входы и промежуточные выходы сети сейчас нужно учитывать больше всего – это одна из важнейших открытых задач нейробиологии.

Свёрточные сети для обработки изображений – это самые глубокие из существующих сетей! Это одна из самых популярных архитектур современных нейронных сетей, и именно ей посвящен данный компьютерный практикум.

## **Свёртки**

Что же, собственно, такое в данном случае *свёртка* и какое она имеет отношение к нейронным сетям?

Краеугольным камнем всех нейронных сетей являются *аффинные преобразования*. В каждом слое полносвязной сети повторяется одна и та же операция: на вход подаётся вектор, который умножается на матрицу весов, а к результату добавляется вектор свободных членов; только после этого к результату применяется некая нелинейная функция активации.

Однако многие типы данных имеют свою собственную структуру. Главный пример такой структуры – изображение, которое обычно представляют, как массив векторов чисел: если изображение чёрно-белое, то это просто массив интенсивностей, а если цветное, то массив векторов из трёх чисел, обозначающих интенсивности трёх основных цветов (красного, зелёного и синего в стандартном RGB).

Основная идея свёрточной сети состоит в том, что обработка участка изображения очень часто должна происходить независимо от конкретного расположения этого участка. Грубо говоря, если вы хотите узнать на фотографии своего друга Васю, совершенно не важно, на 100 или на 200 пикселов ухо Васи отстоит от левого края фотографии.

Поэтому свёрточная сеть попросту покрывает вход небольшими окнами (скажем 5\*5 пикселов) и выделяет одни и те же признаки в каждом таком окне. Затем результат опять можно будет представить в виде «картинки», заменяя окна 5\*5 на их центральные пикселы, и на ней можно будет применить уже второй свёрточный слой.

Мы будем полагать, что в каждом пикселе входного изображения стоит некоторый тензор (обычно одномерный, то есть вектор чисел), и его компоненты называются *каналами* (channels). Такие же матрицы будут получаться и после свёрточного слоя, однако каналов в них теперь может стать больше. Значения каждого признака, которые мы выделили из окон в исходном изображении, теперь будут представлять собой целую матрицу – *карту признаков* (feature map).

Теперь осталось формально определить, что такое свёртка. Свёртка – это линейное преобразование входных данных особого вида. Если – карта признаков в слое под номером , то результат двумерной свёртки с ядром размера 2d+1 и матрицей весов W размера (2d+1)\*(2d+1) на следующем слое будет таким:

где – результат свёртки на уровне , а – её вход, то есть выход всего предыдущего слоя. Пример такой свёртки проиллюстрирован на рисунке 1.

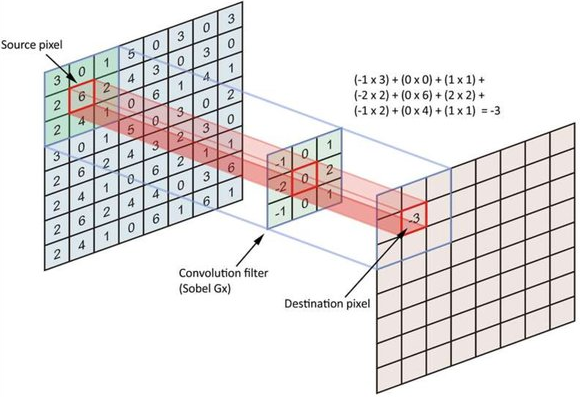


Рис.1. Пример подсчёта результата свёртки.

**Шаг и padding (отступ)**

Синие карты являются входными данными, а голубые карты - выходными*.*

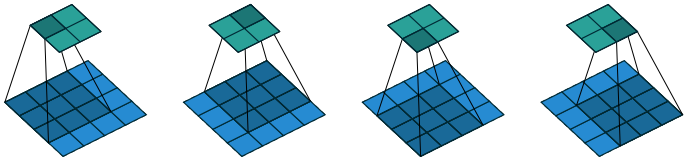


Рис.2. *No padding, unit stride*. Свёртка ядра 3×3 с входом 4×4 с единичным шагом.

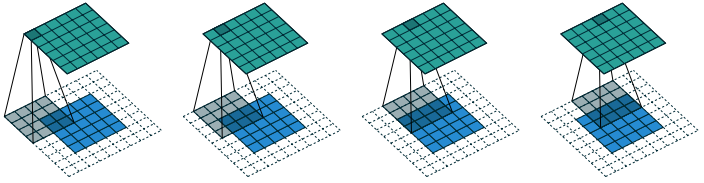
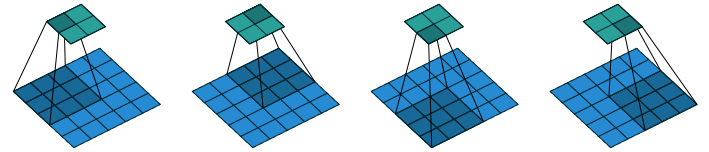


Рис.3. *Arbitrary padding, unit strides*. Свёртка ядра 4×4 с входом 5×5, дополненном границей нулей размера 2×2, с единичным шагом.

 Рис.4. *No zero padding, arbitrary strides*. Свертка ядра 3×3 с входом 5×5 с шагом 2×2.

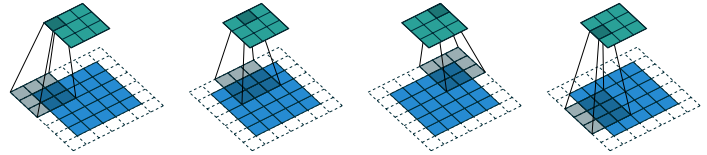


Рис.5. *Arbitrary padding and strides*. Свертка ядра 3×3 с входом 5×5, дополненном границей нулей размером 1×1, с шагом 2×2.

## **Свёрточные нейронные сети**

Идея свёрточных сетей появилась по меркам машинного обучения появилась очень давно. Можно сказать, что первой настоящей свёрточной сетью был Neocognitron Кунихико Фукусимы, появившийся в 1979-1980 годах. Впрочем, Фукусима не использовал градиентный спуск и вообще обучение с учителем. А его работы были довольно прочно забыты. Снова свёрточные сети в уже вполне современной форме появились только в работах группы Яна ЛеКуна в конце 1980-х годов, и с тех пор и до наших дней они вполне успешно применяются для распознавания изображений и многих других задач.

Давайте теперь разберёмся как создавать свёрточный слой на TensorFlow. Свёрточный слой должен брать скользящие окна из исходного изображения и применять к ним одни и те же веса. Свёртка у нас двумерная, то есть «скользящее окно» – это квадратик размером в несколько пикселов.

Чтобы определить свёртку, сначала нужно разобраться с размерностями тензоров. В TensorFlow двумерные свёртки реализуются с помощью функции *conv2d*, которая принимает на вход тензор свёрточных весов и тензор карт признаков набора изображений. Входные данные для двумерной свёртки в TensorFlow должны иметь четырёхмерную структуру, которая выглядит так:

[размер батча, высота, ширина, каналы].

Например, если мы используем мини-бати размером 32 изображения в каждом и обучаем сеть на RGB-изображениях лиц размером 28×28 пикселов, то итоговая размерность тензора данных будет [32, 28, 28, 3].

А размерность тензора свёрточных весов определяется размерами ядра свёртки и числом каналов как на входе, так и на выходе; получается снова четырёхмерный тензор, который выглядит так:

[высота, ширина, входные каналы, выходные каналы].

Например, для фильтра размером 3×3, применяемого к тому же самому трёхканальному изображению и дающему на выходе 32 карты признаков, мы получим тензор размерности [3, 3, 3, 32], в нём будет всего 3×3×3×32=288 весов. А на выходе эти веса, если мы будем предполагать, что в картинку 28×28 пикселов помещается 26×26 окон 3×3, превратят входную картинку в 26×26×32-21632 числа, а это значит, что в полносвязной сети для аналогичной сети потребовалась бы матрица размером 28×28×21632=16959488 весов – разница в пятьдесят тысяч раз!

Конечно, эта разница получается исключительно из-за того, что мы переиспользовали одни и те же веса много раз. В реальности выходит, что сама форма операции свёртки служит очень сильным регуляризатором, который выражает идею о том, что выделение локальных признаков в изображении не должно зависеть от конкретного места, где эти признаки располагаются. И, безусловно, такая огромная разница в требующемся числе весов приводит к тому, что процесс обучения сети тоже сильно упрощается.

Заменяя полносвязный слой свёрточным, мы убиваем сразу двух зайцев:

* добавляем в явном виде предположение, что признаки нужно выделять локально.
* для локальной сети, которая призвана выражать эти локальные признаки, резко увеличивается объём данных на входе.

## **Выполнение**

Теперь кода теоретическая часть свёрточных сетей более или менее ясна, давайте попробуем применить всё это на практике. В этом лабораторном практикуме мы вернёмся к распознаванию рукописных цифр из датасета MNIST и попробуем улучшить решение, полученное в шестой лабораторной работе.

**Упражнение**

В блокноте *convolutional\_network.ipynb* представлен уже реализованный пример свёрточной нейронной сети для распознавания цифр.

Начинается блокнот с загрузки данных и импорта библиотек. Во второй ячейке задаются параметры обучения, размеры данных и т.д. А также происходит объявление заглушек x и y для тренировочных данных. В третьей ячейке создаются слои нашей свёрточной сети: задаётся размер и количество фильтров для слоёв (weights), а также задаются свободные члены (bias). В четвёртой мы преобразовываем данные к нужному формату. Затем, в пятой ячейке, задаётся собственно сама нейронная сеть: функции *conv2d* применяют заданные на предыдущем шаге свёрточные фильтры и выполняют линейную часть работы. Наша модель свёрточной сети содержит:

- свёрточные слои (*conv2d*),

- слои субдискретизации (*maxpool2d*),

- активации (*relu*),

- дропаута (*dropout*).

Функция *reshape* в данном примере преобразует плоский тензор в двумерный и наоборот. Теперь нашу модель на TensorFlow можно записать как представлено в ячейке шесть. В седьмой ячейке описан непосредственно сам цикл обучения и теста. Все используемые при его реализации операции уже известны вам из предыдущих компьютерных практикумов.

Ваша задача заключается в том, чтобы подробно разобраться в реализации данной нейронной сети и настроить параметры обучения наиболее оптимальным образом (то есть так, чтобы функция ошибки *Lost* уменьшалась при обучении и оказалась как можно меньше, а точность классификации тестовой выборки *Accuracy* как можно выше).

Затем, когда вы добьётесь наилучшего результата, сравните полученное в данной работе значение *Accuracy* со значением *Accuracy,* полученном в шестой лабораторной работе данного курса. Поясните полученный результат.

**Задание к лабораторной работе**

1) Выполните все упражнения из данного документа.

2) Представьте отчёт в виде отредактированного файла *convolutional\_network.ipynb* с результатами вашей работы.

3) Сравните полученные в данной работе и в шестой лабораторной работе значения *Accuracy*. Поясните полученный результат.

4) Ответьте на контрольные вопросы:

1. Для решения какой основной задачи были придуманы свёрточные сети?

2. Что общего между зрительной корой человеческого мозга и свёрточными нейронными сетями?

3. Что такое свёртка?

4. Какое значение в свёртках имеют шаг и padding?

5. Что называется картой признаков(feature map)?

6. Когда появилась первая свёрточная нейронная сеть?

7. Как в TensorFlow реализуются двумерные свёртки?

8. В чём заключается главное преимущество использования свёрточных слоёв вместо полносвязных?