Сверточные нейронные сети

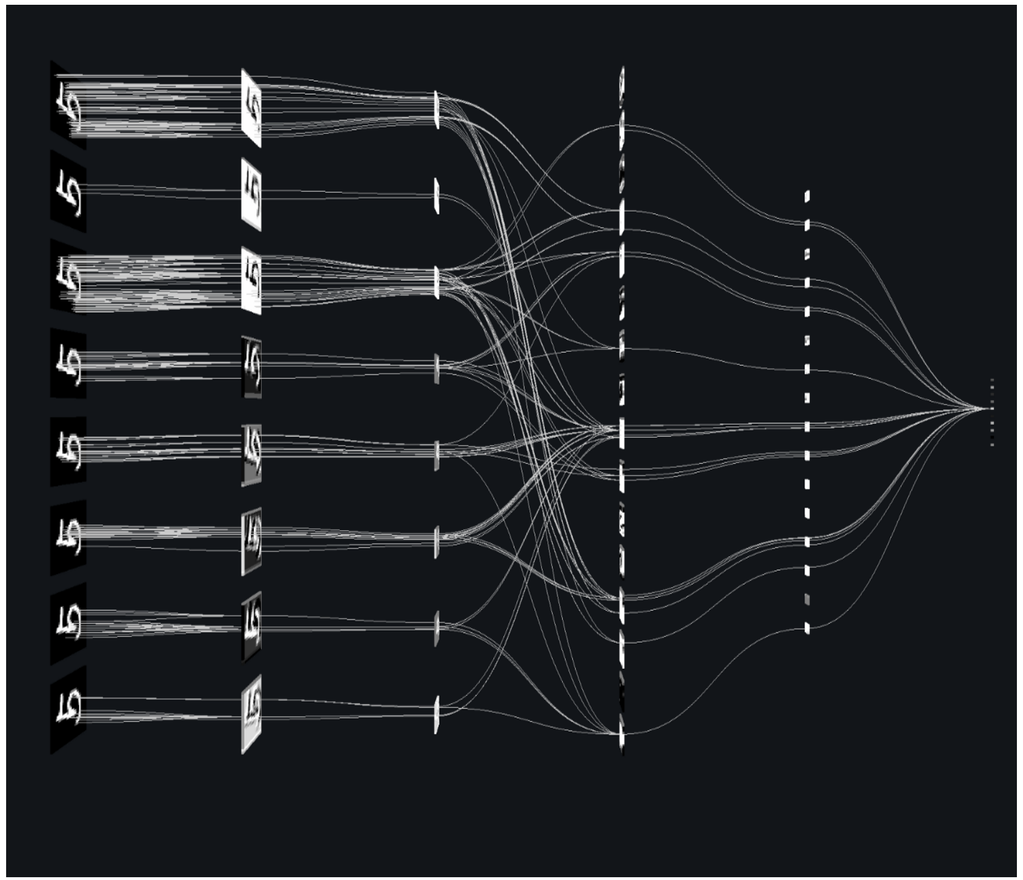
Посевин М.Э. 2022

CNN

Определение

Сверточная нейронная сеть (Convolutional Neural Network — CNN) — это Deep Learning-алгоритм, который может принимать входное изображение, присваивать важность различным областям/объектам в изображении и может отличать одно от другого. Предварительной обработки в CNN требуется значительно меньше по сравнению с другими алгоритмами классификации.

Архитектура CNN схожа с архитектурой связности нейронов в человеческом мозге и была вдохновлена организацией зрительной коры. Отдельные нейроны реагируют на раздражители только в ограниченной зоне поля зрения, известной как рецептивное поле. Совокупность таких полей накладывается, чтобы покрыть всю зону поля зрения.



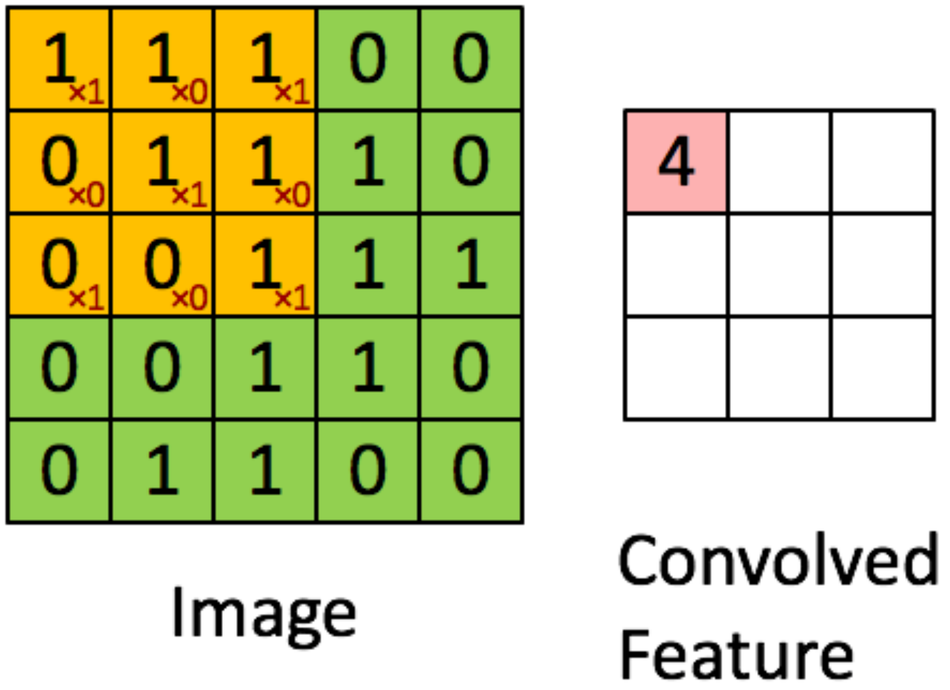
CNN

Свертки

CNN работают на основе фильтров, которые занимаются распознаванием определенных характеристик изображения (например, прямых линий).

Фильтр — это коллекция кернелов; иногда в фильтре используется один кернел.

Кернел — это обычная матрица чисел, называемых весами, которые «обучаются» с целью поиска на изображениях определенных характеристик.



Свертывание изображения 5x5x1 с кернелом размерностью 3x3x1 для получения свернутой функции 3x3x1

CNN

Свертки

Фильтр может перемещаться вдоль матрицы входных сигналов с шагом, отличным от единицы. Шаг перемещения фильтра называется **страйдом** (stride).

Страйд определяет, на какое количество пикселей должен сместиться фильтр за шаг.

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

n_{in}: number of input features
n_{out}: number of output features
k: convolution kernel size
p: convolution padding size
s: convolution stride size

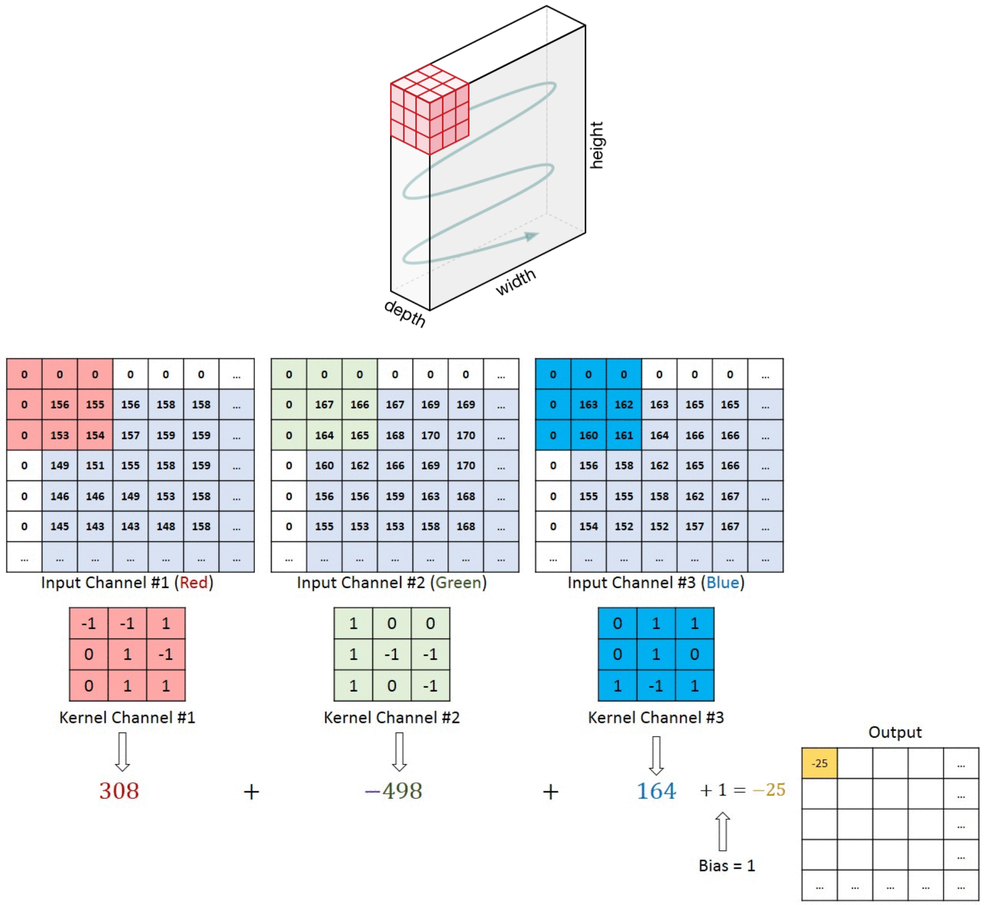
Формула для расчёта количества выходных значений после операции свертки

CNN

Свертки

Фильтр перемещается вдоль изображения и определяет, присутствует ли некоторая искомая характеристика в конкретной его части.

Для получения ответа такого рода совершается операция свертки, которая является суммой произведений элементов фильтра и матрицы входных сигналов.

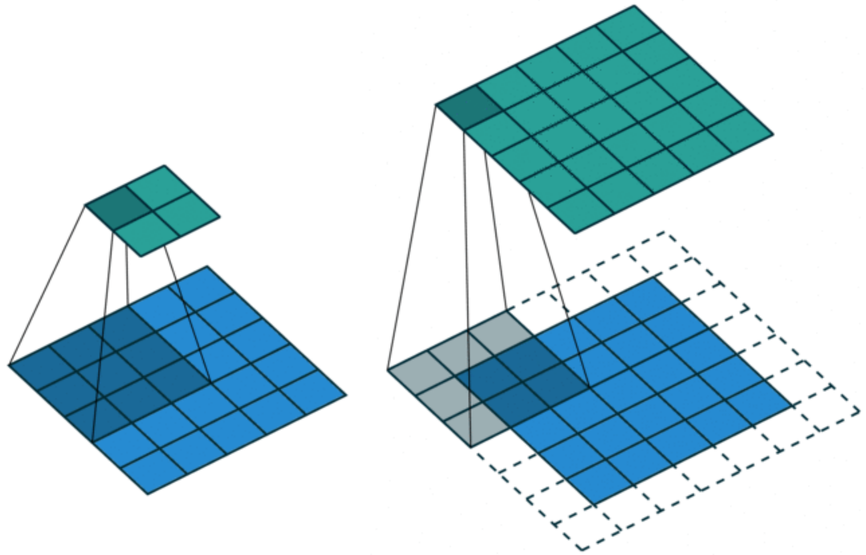


Операция свертки на матрице изображения MxNx3 с кернелом размерностью 3x3x3

CNN

Свертки

Операция свертки может давать два типа результатов: один, в котором свернутый признак имеет меньшую размерность по сравнению с входным, в другом же размерность либо увеличивается, либо остается неизменной. Это делается путем применения **паддинга без дополнения** (Valid Padding) в случае первого или **паддинга с дополнением нулями** (Same Padding) в случае последнего.



Паддинги с дополнением и без

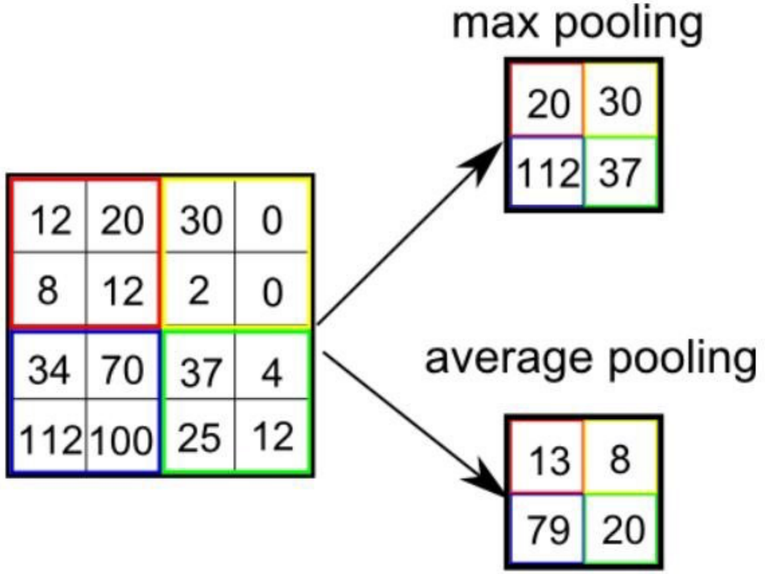
CNN

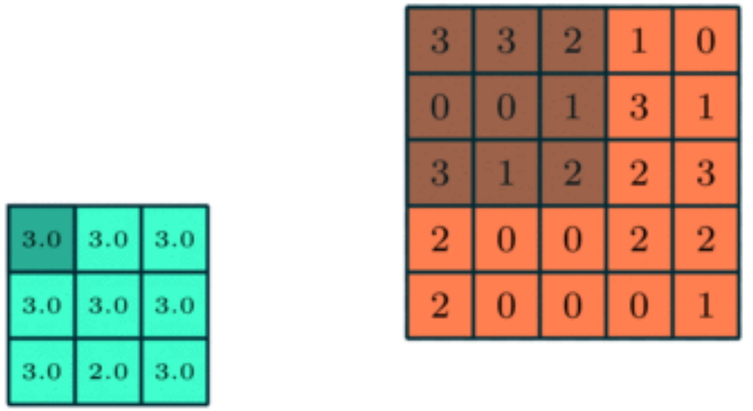
Пулинговый слой

Подобно сверточному слою, пулинговый слой необходим для уменьшения размера свернутого элемента в пространстве. Это помогает уменьшить вычислительную мощность, необходимую для обработки данных, за счет уменьшения размерности. Кроме того, это важно и для извлечения доминирующих признаков, инвариантных вращения и позиционирования, таким образом поддерживая процесс эффективного обучения модели.

Существует два типа пулинга: максимальный (**Max Pooling**) и средний (**Average Pooling**).

Максимальный пулинг также выступает в роли шумоподавителя. Он полностью исключает шумовые сигналы, а также совмещает подавление шума с уменьшением размерности. Средний пулинг просто использует уменьшение размерности как способ подавления шума.





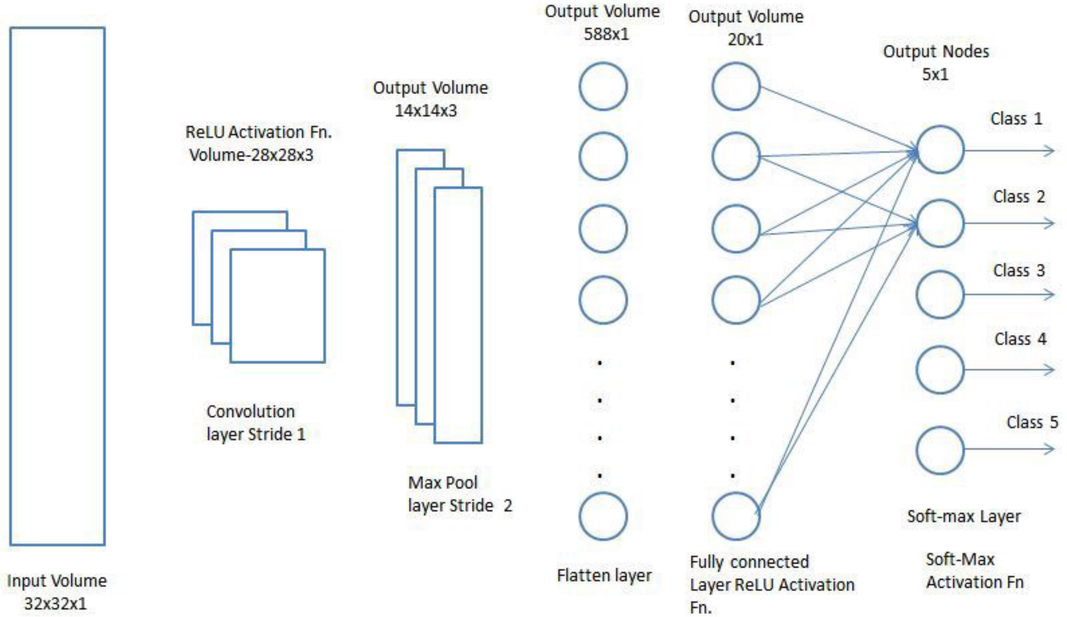
Max pooling

CNN

Полносвязный слой. Результат

Добавление FC Layer — это (обычно) способ изучить нелинейные комбинации высокоуровневых признаков, представленных выходными данными сверточного слоя. FC Layer изучает возможную нелинейную функцию в этом пространстве.

После преобразования входного изображения в подходящий формат для многослойного перцептрона, можно свести изображение в вектор-столбец. Сглаженный вывод подается в нейронную сеть с прямой связью, и затем к каждой итерации обучения применяется обратное распространение. После серии эпох модель способна различать доминирующие и некоторые низкоуровневые признаки изображений и классифицировать их с помощью техники Softmax.

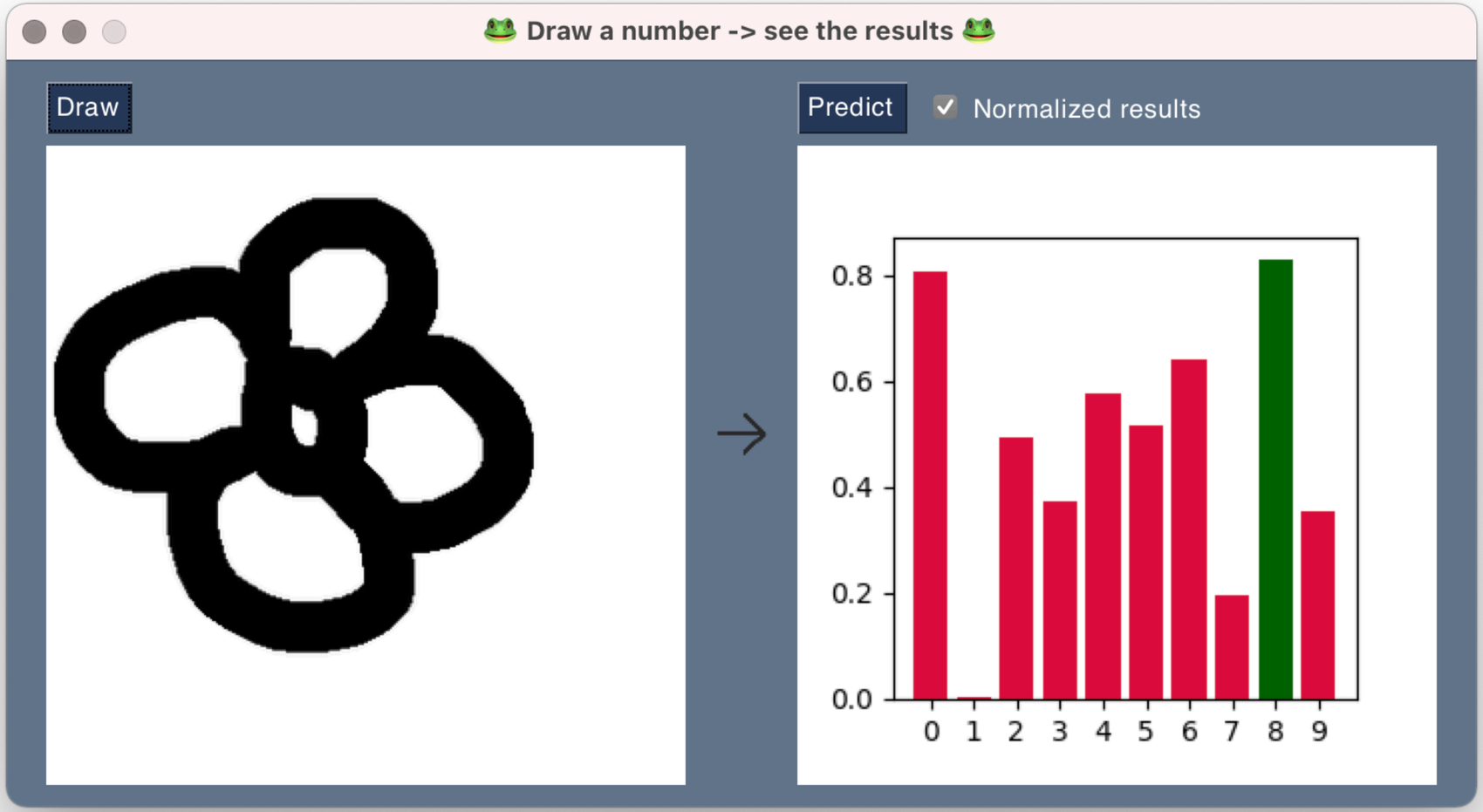


CNN

Существуют множество архитектур доступных CNN, которые сыграли ключевую роль в построении алгоритмов, которые обеспечивают сейчас и будут обеспечивать работу Искусственного интеллекта как такового в ближайшем будущем. Некоторые из них перечислены ниже:

- LeNet
- AlexNet
- VGGNet
- GoogLeNet
- RESNET
- ZFNet

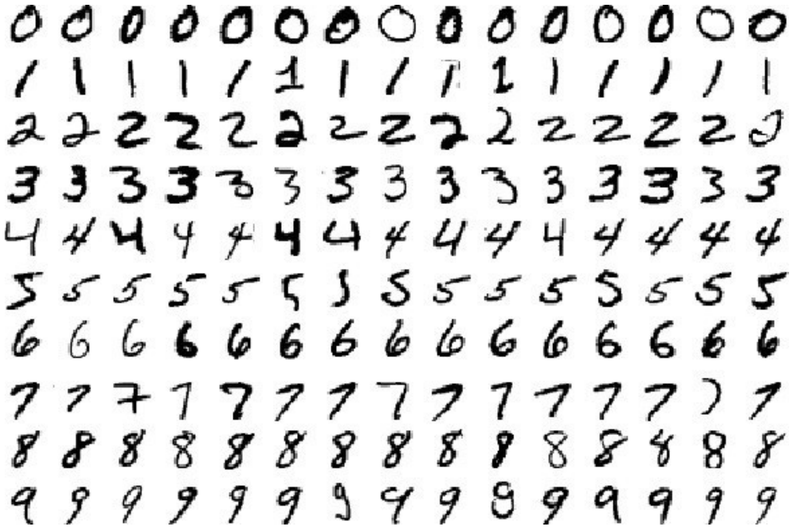
Практические результаты



Практические результаты

Датасет

Один из самых распространенных бенчмарков для оценки скорости работы алгоритма компьютерного зрения является его обучение на базе данных MNIST. Она представляет из себя коллекцию из 70 тыс. написанных от руки цифр (от 0 до 9). Задача заключается в разработке настолько точного алгоритма распознавания цифр, насколько это возможно.

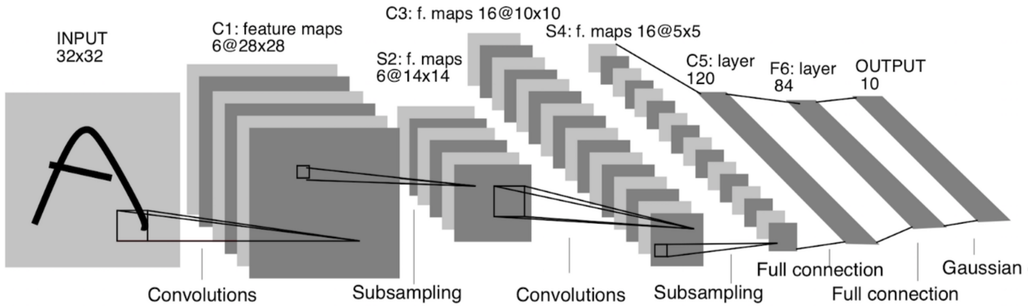


Образец цифр из базы данных MNIST

Практические результаты
CNN

LeNet является одной из первых сверточных нейронных сетей (1994) и способствовал развитию области глубокого обучения. С 1988 года, после многих успешных итераций, этот новаторский результат, достигнутый Яном ЛеКуном, получил название LeNet5.

LeNet-5 является очень эффективной сверточной нейронной сетью для распознавания рукописных символов. Lenet-5 был предложен Янном Лекуном, и точность распознавания набора данных MNIST может достигать 99.2%.



Архитекура LeNet-5

Практические результаты

Реализация

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 24, 24, 6)         156

 max_pooling2d (MaxPooling2D  (None, 12, 12, 6)        0
 )

 conv2d_1 (Conv2D)           (None, 8, 8, 16)          2416

 max_pooling2d_1 (MaxPooling  (None, 4, 4, 16)         0
 2D)

 flatten (Flatten)           (None, 256)               0

 dense (Dense)               (None, 120)               30840

 dense_1 (Dense)             (None, 84)                10164

 dense_2 (Dense)             (None, 10)                850

=================================================================
Total params: 44,426
Trainable params: 44,426
Non-trainable params: 0
_________________________________________________________________
```

Model

```
Epoch 1/5
60/60 [==============================] - 37s 607ms/step - loss: 1.0514 - accuracy: 0.7105
Epoch 2/5
60/60 [==============================] - 20s 336ms/step - loss: 0.2283 - accuracy: 0.9324 - val_loss: 0.1497 - val_accuracy: 0.9554
Epoch 3/5
60/60 [==============================] - 19s 314ms/step - loss: 0.1466 - accuracy: 0.9558
Epoch 4/5
60/60 [==============================] - 20s 331ms/step - loss: 0.1127 - accuracy: 0.9654 - val_loss: 0.0859 - val_accuracy: 0.9737
Epoch 5/5
60/60 [==============================] - 19s 321ms/step - loss: 0.0954 - accuracy: 0.9708
<keras.callbacks.History at 0x7ff17e041f10>
```

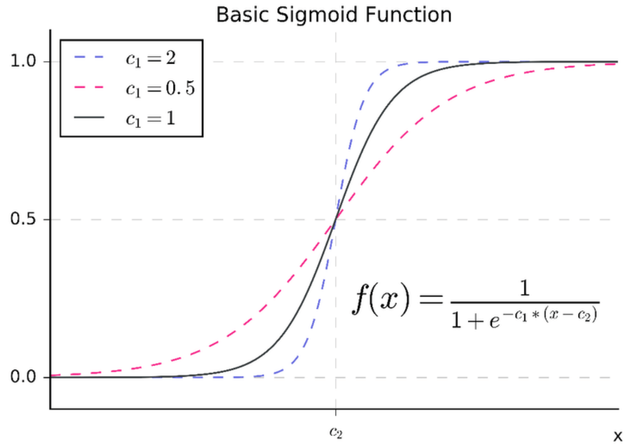
Fitting

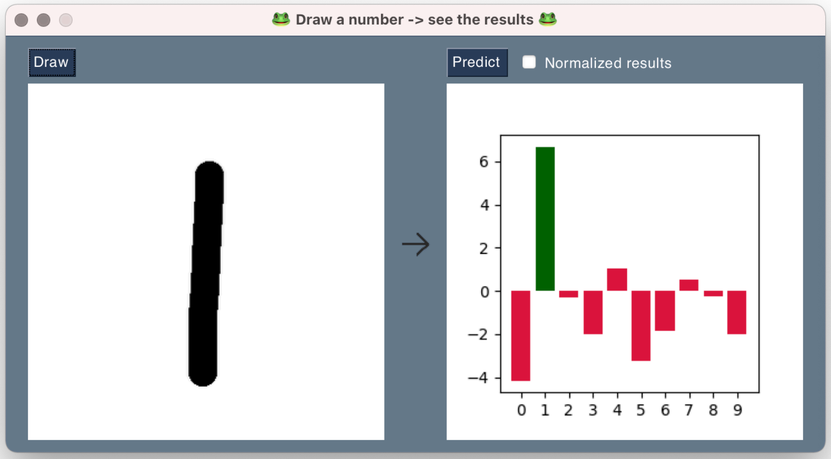
```
loss: 0.0793 - accuracy: 0.9740
```

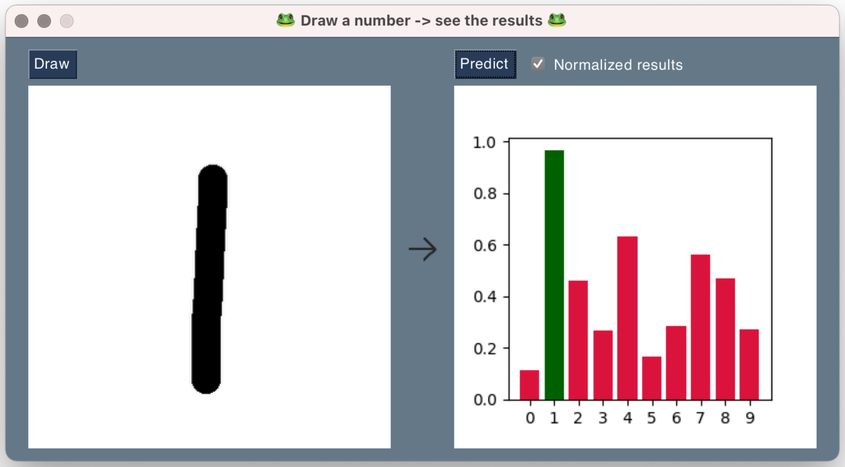
Results

Практические результаты

https://github.com/Mikhail11235/LeNet-5







Практические результаты

https://github.com/Mikhail11235/LeNet-5

Источники

- https://neurohive.io/ru/osnovy-data-science/glubokaya-svertochnaja-nejronnaja-set/
- https://medium.com/@balovbohdan/сверточные-нейронные-сети-с-нуля-4d5a1f0f87ec
- https://github.com/vdumoulin/conv_arithmetic
- https://congyuzhou.medium.com/lenet-5-своими-руками-b60ae3727cd3
- https://datastart.ru/blog/read/kompleksnoe-rukovodstvo-po-svertochnym-neyronnym-setyam-dlya-chaynikov