

Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

**ЭССЕ СТУДЕНТА 317 ГРУППЫ**

**«Свёрточные нейронные сети»**

Выполнил:

студент 3 курса 317 группы

Сидоров Леонид Станиславович

Москва, 2021

## **Аннотация**

Успехи свёрточных нейронных сетей в области классификации изображений в 2012 году пробудили общественный интерес к сфере глубокого обучения. С тех пор эту архитектуру оттеснили более новые активно развивающиеся алгоритмы и парадигмы, однако наряду с рекуррентными и полносвязными сетями подходы на основе свёртки являются основой машинного обучения, многие принципы из которых повлияли на всю предметную область.

Именно поэтому в этой работе мы ознакомимся с основными принципами функционирования свёрточных сетей, с их предшественниками и попробуем подумать о будущем этой парадигмы.

# Содержание

<b>1 Введение</b>	<b>3</b>
1.1 Что такое изображение . . . . .	3
1.2 Линейный подход к классификации на несколько классов . . . . .	4
<b>2 Устройство свёрточной нейронной сети</b>	<b>5</b>
2.1 Операция дискретной свёртки . . . . .	5
2.2 Свёртка как основная часть структуры свёрточного слоя . . . . .	7
2.3 Свёртка как линейная операция . . . . .	10
2.4 Нелинейность и пулинг . . . . .	11
2.5 Полносвязный слой . . . . .	12
2.6 Свёртка $1 \times 1$ . . . . .	13
2.7 Разреженная связность . . . . .	14
2.8 Организация свёрточной нейронной сети . . . . .	15
2.9 Смысловая интерпретация свёрточных сетей . . . . .	16
2.10 Ещё немного о свёртках . . . . .	17
<b>3 Свёрточные нейронные сети сегодня</b>	<b>18</b>
3.1 Недостатки свёрточных сетей . . . . .	18
3.2 Классификация звука на основе спектrogramм . . . . .	20
3.3 Свёрточная сеть для изучения графов . . . . .	21
<b>4 Заключение</b>	<b>22</b>
<b>Список литературы</b>	<b>23</b>

# 1 Введение

Свёрточная нейронная сеть (*convolutional neural network, CNN*) [1] — это специальный вид нейронной сети для обработки данных, имеющих сеточную топологию. Примерами могут служить временные ряды, которые можно рассматривать как одномерную сетку измерений, выбираемых через регулярные промежутки времени, а также изображения, рассматриваемые как двумерная сетка пикселей. Свёрточные сети чрезвычайно успешны в практических приложениях. Своё название они получили благодаря использованию математической операции свёртки. Свёртка — это линейная операция на паре функций, которую можно интерпретировать как «схожесть» одной функции с отражённой и сдвинутой копией другой. Свёрточные сети — это нейронные сети, в которых вместо общей операции умножения на матрицу, по крайней мере в одном слое, используется свёртка.

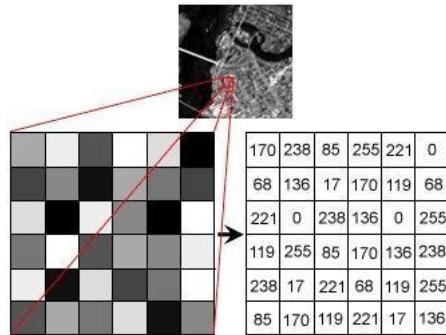


Рис. 1: Монохромное изображение в виде двумерной матрицы [2].

## 1.1 Что такое изображение

В последнее время особенным успехом пользуются свёрточные сети, применяемые для классификации изображений. Как уже упоминалось выше, цифровое изображение можно представить в виде сетки пикселей. В случае монохромного изображения эту сетку реализует двумерный массив размера  $\hat{H} \times \hat{W}$  ( $\hat{H}$  и  $\hat{W}$  — высота и ширина изображения в пикселях соответственно), элементами которого являются значения яркости пикселей из диапазона  $[0, 255]$  (Рис. 1). Случай цветного изображения несколько более сложный, для кодирования цвета могут использоваться различные цветовые схемы, например, **XYZ**, **HSV**, или **YC<sub>b</sub>C<sub>r</sub>**, однако самой попу-

лярной является схема **RGB**, в которой цвет пикселя кодируется тремя значениями из диапазона  $[0, 255]$ , каждое из которых соответствует своему цвету — красному, зелёному или синему. В этом случае изображение представляет собой трёхмерный тензор размера  $\hat{H} \times \hat{W} \times \hat{C}$ , где  $\hat{C}$  — количество цветов (в нашем случае 3). Этот вариант представления пользуется особой популярностью в сфере обучения свёрточных сетей.

## 1.2 Линейный подход к классификации на несколько классов

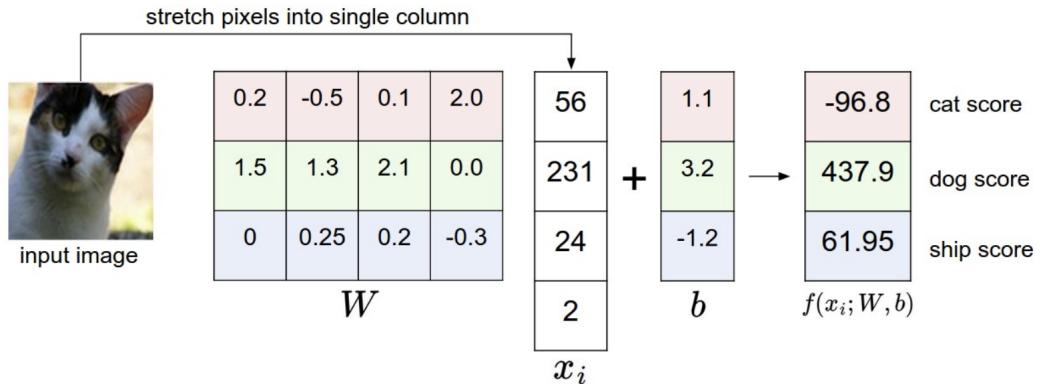


Рис. 2: Пример работы линейной модели классификации изображений [3].

Изначально изображения обрабатывались линейными моделями. Приведём пример такого подхода. Допустим, мы решаем задачу классификации изображений, тогда  $x_i \in \mathbb{R}^D, i = 1, \dots, N$  — это обучающая выборка, состоящая из изображений, то есть трёхмерных тензоров, приведённых к виду одномерного вектора размера  $D = \hat{H} \times \hat{W} \times \hat{C}$ ,  $N$  — размер выборки.  $y_i \in 1, \dots, K$  — это метки классов для каждого объекта выборки, где  $K$  — количество классов. Линейная классификатор имеет следующий вид:

$$f(x_i, W, b) = Wx_i + b,$$

где  $W \in \mathbb{R}^{K \times D}$  и  $b \in \mathbb{R}^K$  — обучаемые параметры. Модель получает на вход вектор длины  $D$  и возвращает вектор длины  $K$ , содержащий оценки для каждого класса. Матрицу  $W$  часто называют весами модели, а вектор  $b$  смещением, потому что он влияет на выходные оценки, но не взаимодействует с входными данными  $x_i$ .

Пример работы алгоритма изображён на Рис. 2, изображению присуждается класс с наивысшей оценкой.



Рис. 3: Матрица весов обученного линейного классификатора [3].

Преимуществами такого подхода являются низкая вычислительная сложность и интерпретируемость результатов, на Рис. 3 изображены веса обученной модели. Как мы видим, они в целом напоминают типичных представителей каждого класса. Поэтому при интерпретации мы просто можем сказать, что входное изображение элементарно "похоже" на один из эталонов. Однако уже на этом изображении мы видим серьёзные недостатки рассмотренного алгоритма, фон у типичных представителей классов "самолёт" и "корабль" является синим, то есть любое синее изображение с большой долей вероятности будет отнесено к одному из этих классов. Помимо этого мы можем заметить, что и ракурс у большинства эталонов является строго заданным, поэтому алгоритм способен распознать автомобиль только с правильного угла. Также линейная модель не учитывает главной особенности изображений — взаимосвязь соседних пикселей.

Эти и многие другие недостатки линейных моделей решают свёрточные нейронные сети, к изучению устройства которых мы и переходим со следующего раздела.

## 2 Устройство свёрточной нейронной сети

### 2.1 Операция дискретной свёртки

Для начала ответим на вопрос, что же такое свёртка. Прообразом операции, используемой в нейронных сетях является линейное преобразование из функционального анализа  $(f * g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau$ , где  $f$  и  $g$  — некоторые функции.

Его дискретной аппроксимацией является выражение  $(f * g)[n] = \sum_m f[m]g[n - m]$ , где  $f$  и  $g$  — векторы одинаковой длины, а индекс  $m$  "пробегает" все их элементы.

Свёрткой в нейронных сетях называется обобщение этого преобразования на случай векторов разной длины, для начала рассмотрим его одномерный случай.

Пусть имеется два вектора  $A = (a_1, a_2, \dots, a_n)$  и  $B = (b_1, \dots, b_k)$ . Тогда одномерной свёрткой  $A$  по  $B$  называется вектор  $C = (c_i)_{i=1}^{n-k+1}$ , в котором:

$$c_i = \sum_{j=1}^k a_{i+j-1} b_j$$

$B$  также называется **ядром** свёртки. Пример такой одномерной свёртки показан на рис. 4, где  $B = (1, 0, -1)$ . Как мы видим, такое преобразование изменяет размер исходного вектора. Решить эту проблему можно с помощью добавления отступа к краям исходного изображения, в нашем случае это отступ шириной один, заполненный нулями. Однако существуют разные виды отступов:

- **нулевой** — отступ заполняется нулями;
- **константный** — отступ заполняется указанной константной величиной;
- **зеркальный** — отступ заполняется зеркально относительно края основного вектора;
- **циклически** — отступ заполняется элементами с другого края основного вектора.

В правой части рисунка изображена свёртка с шагом два, её мы рассмотрим в следующих разделах.

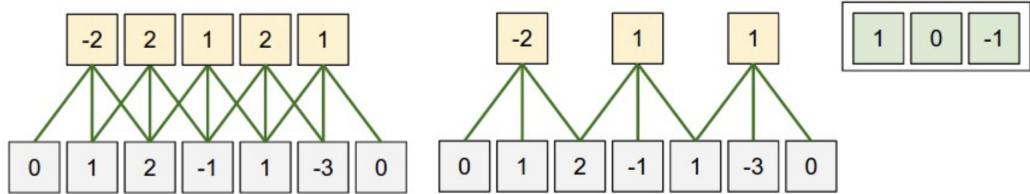


Рис. 4: Пример использования одномерной свёртки [3].

Одномерная свёртка распространена в теории сигналов, где она применяется к некоторому временному ряду.

Аналогичную идею можно применить и к объектам большей размерности, например, к матрицам или многомерным тензорам. Размерность ядра, соответственно, тоже будет увеличиваться.

Для матриц двумерная свёртка определяется следующим образом: пусть  $A = (a_{ij})^{n \times m}$  - исходная матрица,  $K = (k_{ij})^{k \times l}$  - ядро свёртки, тогда двумерной свёрткой матрицы  $A$  по ядру  $K$  будет являться матрица  $C = (c_{ij})^{n-k+1 \times m-l+1}$ , в которой

$$C_{ij} = \sum_{x=1}^k \sum_{y=1}^l A_{i+x-1, j+y-1} K_{xy}.$$

Пример двумерной свёртки представлен на рис. 5.

3 <sub>0</sub>	3 <sub>1</sub>	2 <sub>2</sub>	1	0
0 <sub>2</sub>	0 <sub>2</sub>	1 <sub>0</sub>	3	1
3 <sub>0</sub>	1 <sub>1</sub>	2 <sub>2</sub>	2	3
2	0	0	2	2
2	0	0	0	1

12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0

0	1	2
2	2	0
0	1	2

(a) Вычисление одного элемента

(b) Ядро свёртки

Рис. 5: Пример использования одномерной свёртки [4].

В компьютерном зрении, используются самые разные виды свёрток. Выбрав определённое ядро, мы можем добиться самых разных результатов, например, размытия изображения, выделения вертикальных или горизонтальных линий, выделения контуров и т.д. .

## 2.2 Свёртка как основная часть структуры свёрточного слоя

Мы вплотную подобрались к основному элементу свёрточной нейронной сети — свёрточному слою. Как известно, все операции, проводимые внутри нейронной сети, осуществляются над тензорами большой размерности. Введём в нейронную сеть слой, который будет принимать на вход тензор и применять у нему свёртку с помощью ядра заданного размера. Будем рассматривать частный случай — трёхмерные тензоры, которыми чаще всего оперируют при работе с изображениями. Кроме того, будем рассматривать ядра, которые имеют такую же третью размерность, что и входные тензоры ( $\hat{C}$  — количество каналов). Математически это записывается следующим образом:

$$C_{ij} = \sum_{l=1}^h \sum_{p=1}^w \sum_{k=1}^c X_{i+l-1,j+p-1,k} K_{lpk},$$

где  $X \in \mathbb{R}^{\hat{H} \times \hat{W} \times \hat{C}}$  — входной тензор,  $K \in \mathbb{R}^{h \times w \times \hat{C}}$  — ядро свёртки,  $C \in \mathbb{R}^{H-h+1 \times W-w+1}$  — свёртка,  $h$  и  $w$  — высота и ширина свёртки соответственно.

Заметим, что на выходе слоя мы получаем тензор состоящий всего из одного канала. Основная идея свёрточного слоя заключается в использовании нескольких ядер свёртки. Каждому ядру соответствует своё преобразование, например, детектирование контуров изображения или применение некоторого фильтра. Для повышения предсказательной способности алгоритма мы бы хотели иметь доступ сразу к нескольким версиям преобразованного изображения, поэтому использование множества различных свёрточных ядер оправдано. На выходе в таком случае мы будем получать тензор, количество каналов которого равно количеству используемых в слое ядер свёртки. Пример подобного преобразования представлен на Рис. 6.

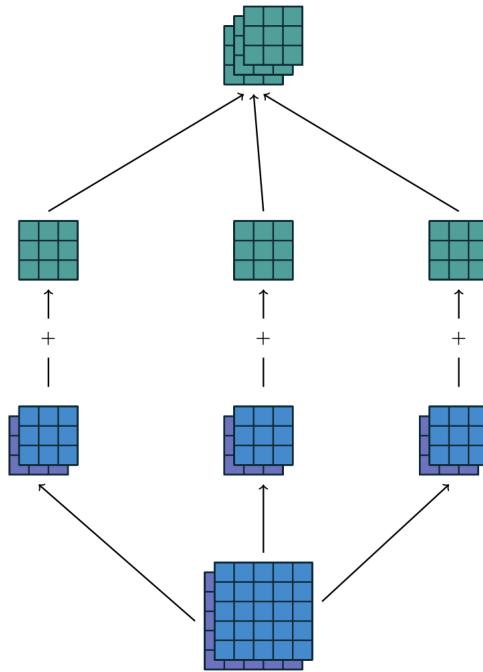


Рис. 6: Принцип работы свёрточного слоя с несколькими ядрами [4].

При рассмотрении одномерного случая мы упомянули, что, варьируя размер отступа, можно сохранять первоначальную длину входного вектора. В многомерном

случае это правило также выполняется. Если ширину и высоту входного тензора необходимо сохранить, то это также можно сделать с помощью отступа (рис. 7a).

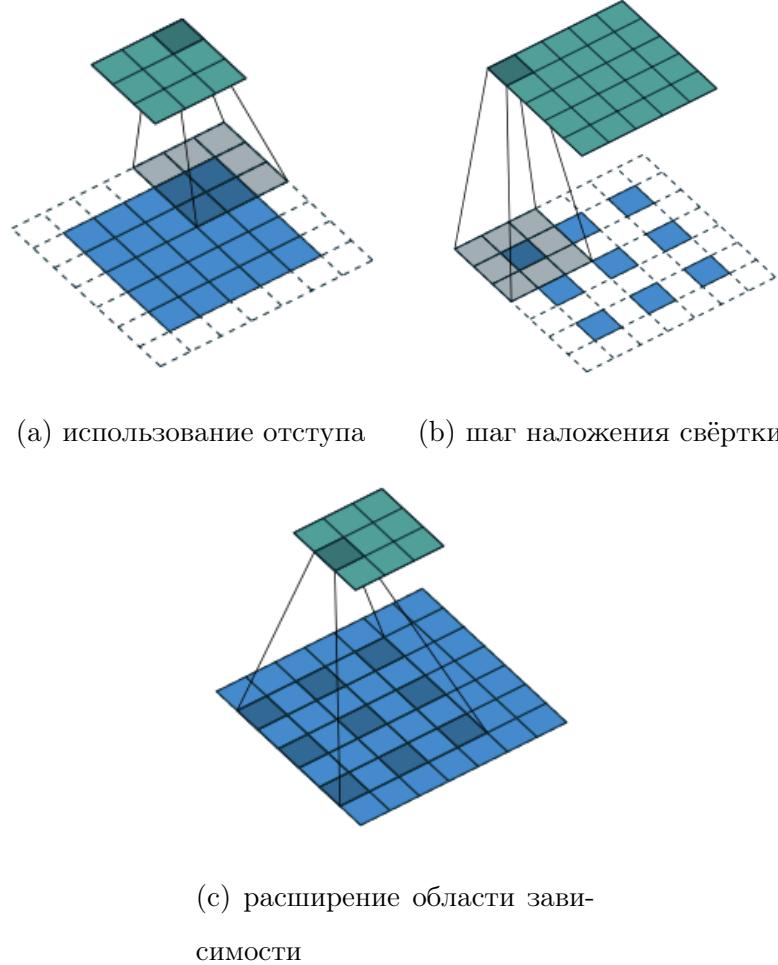


Рис. 7: Примеры модификаций операции свёртки [4].

Ещё одна опция — это значительное понижение размерности, выбирая шаг, с которым будет перемещаться окно свёртки (Рис. 7b), можно кратно уменьшить размер входного тензора. То есть движения этого окна осуществлять не на соседние клетки, а через одну или даже несколько.

Мы можем сказать, что свёртки собирают информацию с некоторой области, которую можно по-разному интерпретировать в зависимости от вида ядра, например, судить, есть ли в данном фрагменте изображения вертикальные линии. В качестве ещё одной модификации процесса свёртки предлагается расширение области действия ядра, но не за счёт фактического увеличения размера ядра, а за счёт увеличения покрываемой области (Рис. 7c). Элементы, с которых берутся значения, не

расположены по соседству, а удалены друг от друга на некоторую одинаковую для всех величину.

Все эти базовые приёмы манипуляции с ядром свёртки позволяют нам проводить гибкую настройку структуры нейронной сети в целом. Но на этом рассмотрение свёрточного слоя не заканчивается, помимо процесса свёртки в него входят этапы нелинейного преобразования признаков и их агрегации. Эти преобразования мы рассмотрим в следующих разделах.

## 2.3 Свёртка как линейная операция

Но перед этим отметим, что применение свёртки в нейронной сети всё ещё остаётся линейной операцией.

Возьмем для примера свертку с  $3 \times 3$  ядром, применённую к  $4 \times 4$  изображению (все остальные параметры стандартные). Если бы вход и выход были развернуты в векторы (слева направо, сверху вниз), свертка могла бы быть представлена в виде разреженной матрицы  $C$ , где ненулевыми элементами являются элементы  $w_{i,j}$  ядра (причем  $i$  и  $j$  являются строкой и столбцом ядра соответственно). Матрица  $C$  представлена на Рис. 8.

$$\begin{pmatrix} w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} \end{pmatrix}$$

Рис. 8: Свёртка в виде матрицы, размер ядра  $3 \times 3$ , а изображения  $4 \times 4$  [4].

Эта линейная операция принимает матрицу, представленную в виде 16-мерного вектора, а возвращает 4-мерный вектор, который позже преобразуется в выход размера  $2 \times 2$ .

Используя это представление, обратный проход легко получается путем транспонирования  $C$ . То есть, как и в случае линейного слоя, градиент пропускается путем его умножения на  $C^T$ . Эта операция принимает 4-мерный вектор в качестве входа и возвращает 16-мерный вектор в качестве выхода, его шаблон связности совместим с  $C$  по построению.

Примечательно, что ядро  $w$  определяет как  $C$ , так и  $C^T$ , используемые для прямого и обратного проходов.

Также отметим, что если поменять  $C$  и  $C^T$  местами (то есть прямой и обратный проходы), то мы получим операцию обратной свёртки (*deconvolution*), которая применяется, например, в моделях сегментации. Однако эта операция, применённая непосредственно после свёртки, не гарантирует восстановления исходных данных, потому что свёртка является необратимым преобразованием, сохраняется лишь форма исходного тензора ( $\hat{H}$  и  $\hat{W}$ ).

## 2.4 Нелинейность и пулинг

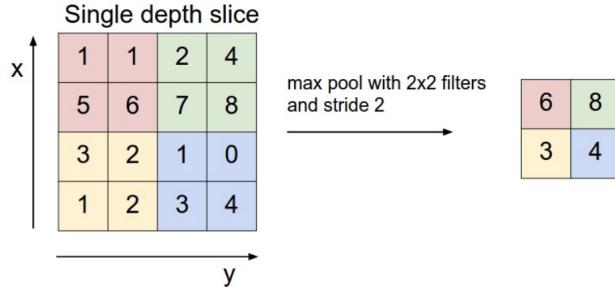


Рис. 9: Пример пулинга — взятие максимума, или *MaxPool* [3].

Важной частью свёрточного слоя является процесс преобразования и фильтрации полученных признаков. Процесс свёртки заканчивается передачей сформированного тензора признаков, его необходимо обработать, пропустив через нелинейные функции активации. Выбор нужной функции очень сильно влияет на функционал нейронной сети. Варьируя разные функции активации, можно добиваться разных результатов. Укажем наиболее популярные функции:

- ReLu:  $f(x) = \max(0, x)$ ;
- Сигмоида:  $f(x) = \frac{1}{1+e^{-x}}$ ;
- Гиперболический тангенс:  $f(x) = \frac{e^{2x}-1}{e^{2x}+1}$ .

Следующим и последним этапом свёрточного слоя будет процесс понижения размерности, пулинг. Его основная задача заключается в уменьшении размера тензора (а именно  $\hat{H}$  и  $\hat{W}$ ), чтобы следующие слои свёрток оперировали над большей

областью (генерировали более абстрактные признаки). Для этого входной блок делится на части, над каждой из которых выполняется некоторая операция агрегации, чаще всего это взятие максимума или среднего (Рис. 9).

Однако на этом виды пулинга не заканчиваются. Они могут отличаться друг от друга как видом функции агрегации ( $L_2$ -норма, выбор значения с некоторой вероятностью), так и формой входа или выхода (выбор  $k$  максимальных элементов). Например, в динамическом пулинге размер областей, на которые делится изображение, может быть разным (Рис. 10).

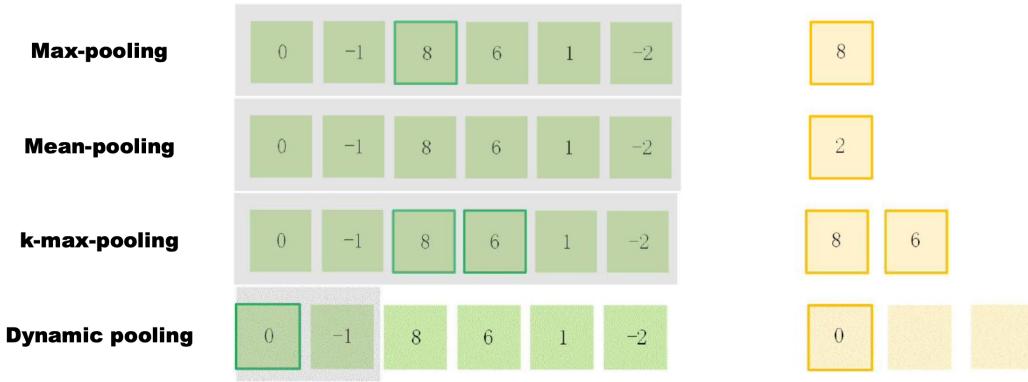


Рис. 10: Примеры разных видов пулинга [5].

Этот процесс, помимо уменьшения размера тензора, позволяет добавить инвариантность представлению изображения к небольшим поворотам или сдвигам, ускоряет вычисления и привносит в сеть дополнительную нелинейность.

Заметим, что здесь нет параметров, по которым бы вычислялся градиент. Во время обратного распространения ошибки градиент передаётся только по тем ячейкам, которые были задействованы при вычислении результата (в случае максимума это только максимальный элемент).

## 2.5 Полносвязный слой

Полносвязный слой в свёрточной нейронной сети фактически повторяет устройство линейной модели многоклассовой классификации, рассмотренной в разделе 1.2. Только на этот раз модель делает предсказания не на основе значения пикселей изображения, а опираясь на высокоуровневые признаки, полученные после применения

нескольких свёрточных слоёв. Поэтому эти предсказания зачастую инвариантны к сдвигу и повороту изображения, а также лишены многих других недостатков линейных классификаторов.

Добавим, что в случае полносвязного слоя размерность выхода не ограничена количеством классов или какими-либо ещё свойствами задачи. Выход этого слоя может передаваться на следующий, вообще говоря, не финальный слой сети (например, как на Рис. 13).

Однако полносвязный слой вовсе не обязан присутствовать в свёрточной сети, например, в [6] он заменён на усреднение по всем элементам тензора (*average pooling*).

## 2.6 Свёртка $1 \times 1$

Отдельного рассмотрения заслуживает свёртка с размером ядра  $1 \times 1$ , которая является основой многих других подходов в построении свёрточных нейронных сетей.

Впервые представленный в [7], слой свёртки  $1 \times 1$  использовался для "сквозного уменьшения количества каналов" или объединения перекрестных каналов. Другими словами, свёртка  $1 \times 1$  используется для уменьшения количества каналов при введении нелинейности (Рис. 11).

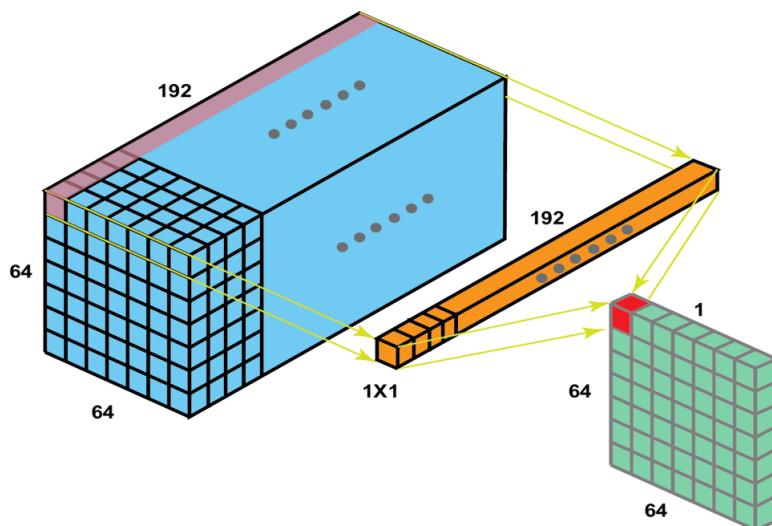


Рис. 11: Пример работы  $1 \times 1$  свёртки [8].

Этот трюк особенно полезен при построении глубоких и очень глубоких сетей, многие знаменитые архитектуры использовали его для сокращения количества вычислений [6, 9, 10]. Дело в том, что при применении свёртки к слишком глубокому

тензору (например,  $\hat{C} = 256$ ), мы можем предварительно уменьшить его глубину с помощью свёртки  $1 \times 1$ . Таким образом мы можем сохранить глубину самой сети, при этом на порядки уменьшая количество обучаемых параметров.

Отметим, что  $1 \times 1$  свёртка вместе с некоторой нелинейностью фактически образуют нейронную сеть, принимающую на вход сквозные проекции элементов тензора ("столбцы" размера  $1 \times 1 \times \hat{C}$ ). Отсюда происходит и название оригинальной статьи [7] — "Сеть в сети" (*Network in network*).

## 2.7 Разреженная связность

В слоях традиционной нейронной сети применяется умножение на матрицу параметров, в которой взаимодействие между каждым входным и каждым выходным элементами описывается отдельным параметром. Это означает, что каждый выходной элемент взаимодействует с каждым входным элементом. В свёрточных сетях же взаимодействия обычно разреженные (это свойство называют ещё разреженной связностью). Визуальное сравнение представлено на Рис. 12.

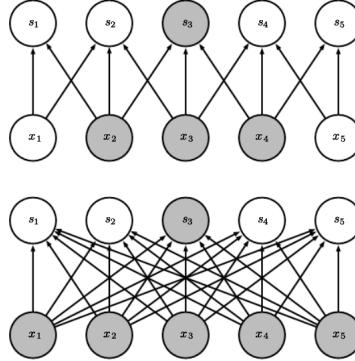


Рис. 12: Сравнение количества параметров для линейного и свёрточного слоёв [11].

Это достигается за счет того, что ядро свёртки меньше её входа. Например, входное изображение может состоять из тысяч или миллионов пикселей, однако мы можем обнаружить небольшие значимые признаки, такие как границы, с помощью ядер свёртки, которые занимают только десятки или сотни пикселей. Это значит, что мы можем хранить значительно меньше параметров (достаточно хранить все ядра свёртки), что одновременно снижает требования по памяти модели и повышает ее статистическую эффективность. Кроме того, вычисление выхода требует меньшего

го количества операций. Вместе все перечисленные улучшения намного повышают эффективность модели.

## 2.8 Организация свёрточной нейронной сети

Опишем наиболее распространённую архитектуру свёрточной нейронной сети. На первых уровнях используется несколько слоёв свёртки, когда же размеры тензоров становятся достаточно малы (здесь это  $\hat{H}$  и  $\hat{W}$ ), их значения агрегируются, например, с помощью полносвязных слоёв. Пример подобной сети можно увидеть в [12], эта архитектура изображена на Рис. 13.

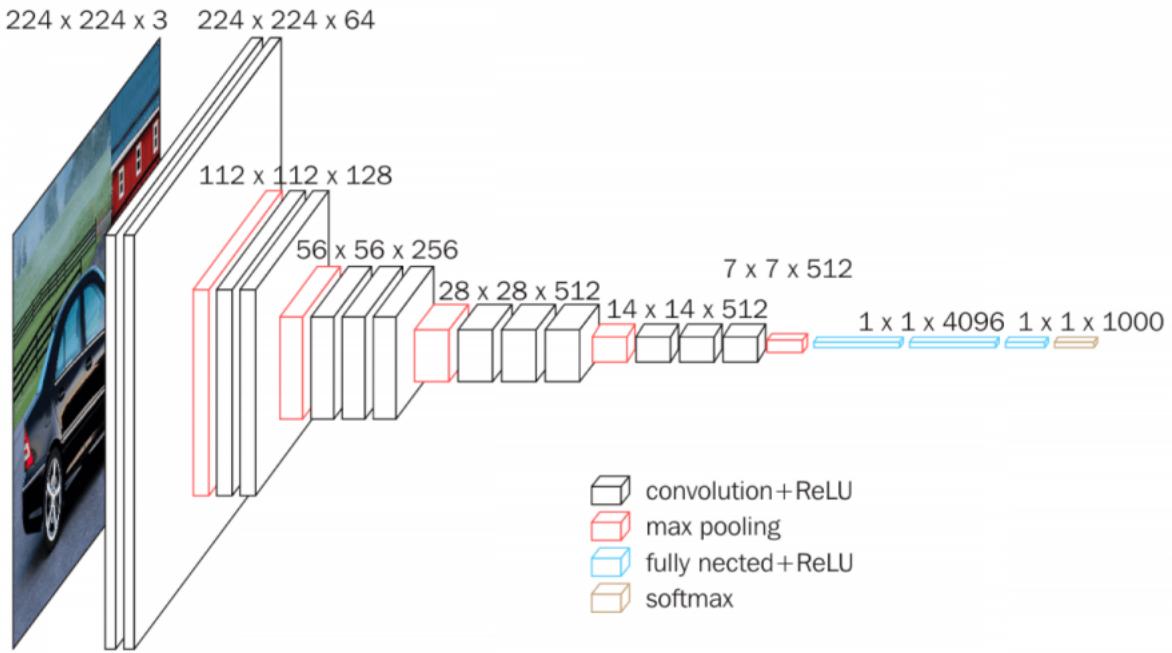


Рис. 13: Пример архитектуры свёрточной нейронной сети (VGG16) [13].

Выбор элементов ядер остаётся за нейросетью, в процессе обучения, например, с помощью метода обратного распространения ошибки нейросеть сама подбирает нужные значения элементов ядер свёртки с целью минимизации выбранного функционала ошибки.

Отметим, что свёрточный слой переводит тензор в тензор, обычно размер выхода отличается от размера входа. На практике используют большое количество свёрточных слоёв. С каждым новым слоём высота  $\hat{H}$  и ширина  $\hat{W}$  поникаются, но уве-

личивается количество каналов  $\hat{C}$ . Для чего это делается, мы опишем в следующем пункте.

## 2.9 Смысловая интерпретация свёрточных сетей

Как было упомянуто ранее, различные ядра могут выполнять различные функции, например, в двумерном случае ядра могут размывать изображения или выявлять, есть ли в области изображения особые структуры, такие как прямые линии, и т.д. . При использовании свёрток информация собирается не со всего объекта целиком, а ограниченными участками. Размер каждого такого участка характеризуется размером и способом наложения ядра этой свёртки.

Соответственно, в каждой свёртке обученной нейронной сети хранится конкретная информация об областях изображения (если мы работаем изначально с изображением). Повторив операцию свёртки, мы получим уже более высокоуровневое представление изображения. То есть, если первая свёртка несла в себе информацию о расположении прямых линий, то вторая уже будет содержать информацию об изогнутых. Повторяя процесс свёртки многократно, мы будем получать сведения о все более сложных абстракциях. Процесс свёртки помогает перейти от конкретных деталей к более абстрактным. На последнем шаге мы будем получать высокоуровневые представления. Например, от вопроса, есть ли на изображении вертикальные линии, постепенно мы перейдём к вопросу, есть ли на изображении машина или медведь. Визуализация иерархии признаков показана на Рис. 14.

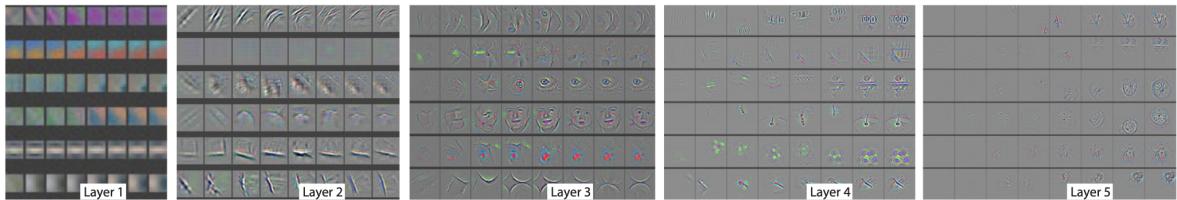


Рис. 14: Визуализация признаков на различных слоях свёрточной сети (для каждого слоя подвыборка признаков выбиралась случайно, в каждом слое показана эволюция признака по мере обучения)[14].

Немаловажным процессом является пулинг. Его польза обусловлена тем, что он уменьшает размеры тензоров, с которым ведутся операции, и процесс уменьшения размера происходит на каждом канале независимо друг от друга.

Отметим ещё одну важную мысль: использование, например, *MaxPooling* идеёно меняет принцип работы нейросети. Эта операция выбирает из некоторой области максимальное значение, тем самым акцентирует внимание не на расположении какого-либо объекта на изображении, а на самом факте наличия этого объекта. В итоге мы получаем информацию о содержании изображения.

## 2.10 Ещё немного о свёртках

В предыдущих разделах были описаны основные принципы, на которые опираются свёрточные нейронные сети. В этой же части мы рассмотрим популярные модификации свёрток:

### 1. Пространственные свёртки (Spatial Separable Convolutions)

Основная идея заключается в представлении исходного ядра в виде произведения двух векторов (Рис. 15). В этом случае мы будем применять две свёртки вместо одной, но это значительно сократит количество обучаемых параметров (6 против 9 в случае  $3 \times 3$  ядра). Отсюда следует и название метода, каждый вектор соответствует размерностям  $\hat{H}$  и  $\hat{W}$ . Важнейшим недостатком этого метода является ограниченность класса матриц, представимых подобным образом.

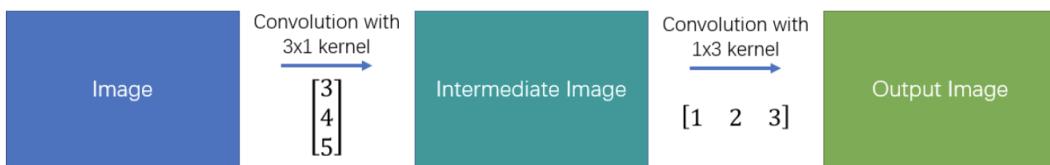


Рис. 15: Пример применения пространственной свёртки размера  $3 \times 3$  [15].

### 2. Свёртка с разделением на группы (Group Convolutions)

Главной идеей этого метода является разделение входного тензора на несколько частей, каждая из которых обрабатывается отдельным ядром, после чего результаты конкатенируются (Рис. 16). В такой свёртке тоже меньше параметров,

чем в "полной" однако класс описываемых преобразований значительно больше, чем в предыдущем примере.

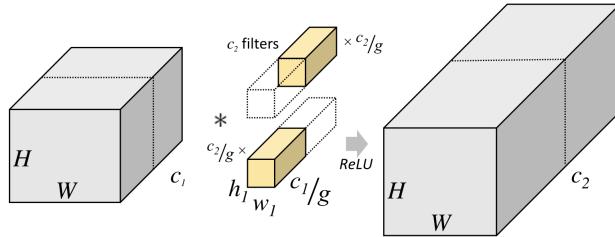


Рис. 16: свёртка с разделением на группы [16].

### 3. Свёртка по каналам (Depth-wise Convolution)

В этом методе тензор делится на каналы, каждый из которых будет обрабатываться своим ядром независимо от других. В таком случае требуется число ядер, кратное числу каналов входного тензора. Полученные свёртки конкatenируются, образуя выходной тензор. Свёртка по каналам является частным случаем групповой свёртки, в котором "экономия" параметров максимальна (Рис. 17a).

### 4. Свёртка по каналам с объединением (Depth-wise Separable Convolution)

Является модификацией предыдущего метода, к выходу свёртки по каналам применяется свёртка  $1 \times 1$  (Рис. 17b) (причины применения свёртки  $1 \times 1$  описаны в разделе 2.6).

Мы рассмотрели наиболее популярные модификации свёрток. Комбинируя описанные выше подходы, можно придумать огромное количество новых методов, подходящих под каждую конкретную задачу.

## 3 Свёрточные нейронные сети сегодня

### 3.1 Недостатки свёрточных сетей

Теперь поговорим о последних тенденциях в области свёрточных нейронных сетей. Джекфри Хинтон, человек стоящий у истоков глубокого обучения и соавтор сети AlexNet [18], положившей начало буму нейронных сетей в 2012 году, на своём

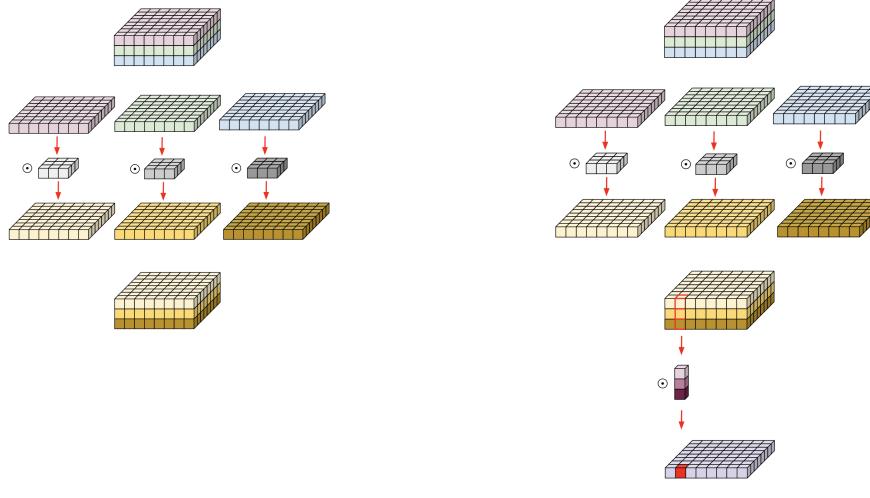


Рис. 17: Частные случаи групповой свёртки [17]

недавнем выступлении [19] заявил, что свёрточные нейронные сети недостаточно хороши для распознавания изображений.

Он полагает, что будущее этой области стоит за новым семейством архитектур, например, капсульными сетями [20]. Причиной подобному заявлению послужили два основных недостатка свёрточных нейронных сетей:

- несмотря на то, что они инварианты по отношению к повороту и сдвигу изображения, сети испытывают большие сложности, когда речь идёт о смене перспективы или освещения;
- свёрточные сети уязвимы по отношению к состязательным атакам (*adversarial attacks*), а значит, при обработке изображений они руководствуются логикой, отличной от человеческой.

Таким образом, свёрточные сети тяжело приспособить к распознаванию объектов реального мира, например, через датчики робота или само пилотируемого автомобиля. Однако это не значит, что мы должны полностью отказываться от архитектуры. За последние годы исследователи придумали для неё множество приложений, выходящих за рамки распознавания изображений. Сейчас мы рассмотрим пару таких подходов.

### 3.2 Классификация звука на основе спектрограмм

В [21] исследуется поведение свёрточной сети, предобученной на датасете **ImageNet**, в задаче классификации звука на основе спектрограмм. Мотивация исследования состоит в том, что современные модели для классификации звука на основе трансформеров имеют огромное количество параметров и используют нетривиальные признаковые пространства. Авторы же утверждают, что они способны достичь того же уровня качества при помощи гораздо более простых свёрточных сетей.

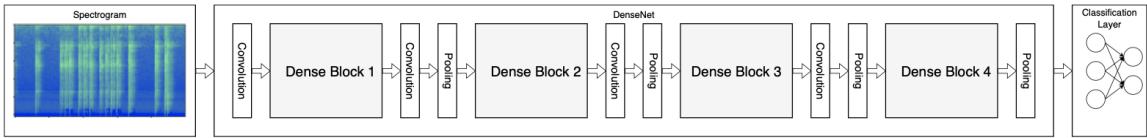


Рис. 18: Пример работы сети для классификации звука [21].

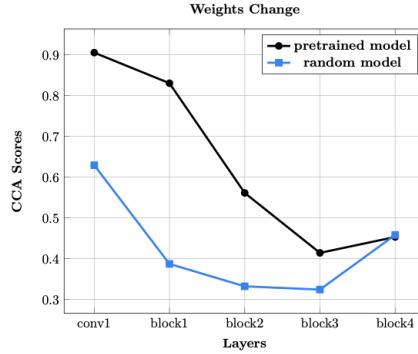
Алгоритм заключается в переводе звукового файла в спектрограмму и последующего применения сети **DenseNet** [22] к полученному изображению. Ключевой особенностью этого подхода является то, что сама сеть никак не модифицировалась. В итоге авторам удалось добиться лучшего качества на двух из трёх заявленных датасетах, используя ансамбль свёрточных сетей (Рис. 19а).

Так же в работе изучалось изменение предобученных весов в сети по мере обучения (Рис. 19б). Как оказалось, веса, находящиеся на начальных слоях почти не претерпели изменений, в то время как веса, находящиеся на глубоких уровнях сети, сильно адаптировались под новые данные. Эти выводы согласуются с нашими рассуждениями из раздела 2.9, ведь на начальных слоях сеть запоминает простейшие абстракции, такие как линии и изменения цвета, а на последних она хранит информацию о высокоуровневых объектах, например, о человеческом лице или колесе автомобиля.

Таким образом, путём несложных преобразований, мы можем приспособить свёрточные сети для совершенно нового класса задач, главное представить вход в нужном виде. В этом случае проблемы, перечисленные в предыдущем пункте, неактуальны, потому что спектрограммы — это высокоточное признаковое описание звука, в котором невозможны смена перспективы или яркости.

Model	GTZAN	ESC-50	UrbanSound8K
Choi, Keunwoo, et al.[45]	89.80%	-	69.10%
Multi-Stream Network[18]	-	84.90%	-
Attention-Based CRNN[11]	-	86.10%	-
ES-ResNet [32]	-	91.50%	85.42%
GTZAN [60]	94.50%	-	-
DenseNet (Random)	88.50%	72.50%	76.32%
DenseNet (Pretrained)	91.39%	91.16%	85.14%
DenseNet (Pretrained Ensemble)	90.50%	<b>92.89%</b>	<b>87.42%</b>

(a) сравнение ансамбля DenseNet с лучшими алгоритмами



(b) корреляция весов со значениями до обучения

Рис. 19: Изучение свойств DenseNet, обученной на спектrogramмах [21].

### 3.3 Свёрточная сеть для изучения графов

Помимо модели применения поменялись также и сами архитектуры свёрточных сетей. Например, популярностью пользуется сеть, созданная для предсказания дорожного трафика [23]. В ней используется графовая свёртка

$$\hat{X} = U \Lambda_\theta U^T X, \quad (1)$$

где  $X \in \mathbb{R}^n$  — входной графовый сигнал (вектор состояний вершин),  $U \in \mathbb{R}^{n \times n}$  — матрица, содержащая базис из собственных векторов Лапласиана исходного графа,  $\hat{X} \in \mathbb{R}^n$  — выход свёртки,  $\Lambda_\theta \in \mathbb{R}^{n \times n}$  — ядро свёртки,  $n$  — количество вершин в графе.

Подобный вид преобразования позволяет извлекать локальные признаки из графовой структуры. Также, так как  $U$  зависит только от вида самого графа, в процессе обучения мы ищем только значения диагональной матрицы  $\Lambda_\theta$ .

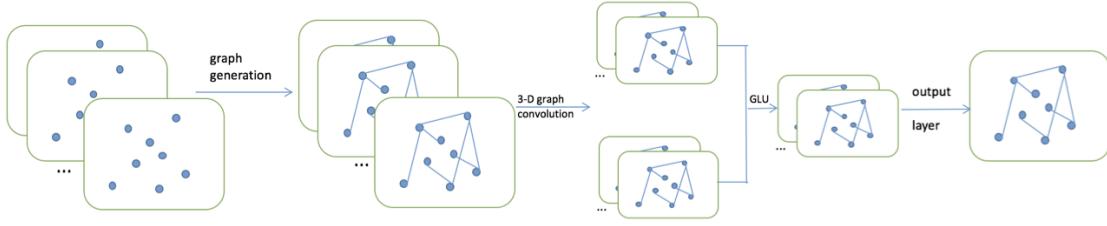


Рис. 20: Архитектура графовой свёрточной сети [23].

Также стоит отметить, что в этой сети используется трёхмерная свёртка (координаты и время), свёртка (1) применяется для каждого измерения. В отличие от предшественников, эта архитектура не пользуется априорной структурой графа, а создаёт своё внутреннее представление на основе сравнения имеющихся маршрутов. Итоговый результат проходит через пропускной линейный блок (*Gated Linear Unit, GLU*).

## 4 Заключение

В этой работе мы изучили основные строительные блоки свёрточной нейронной сети, посмотрели на то, как эти блоки можно настраивать и подумали о будущем этого сегмента глубокого обучения. Кроме того, мы затронули мотивацию тех или иных составляющих сети и немного поговорили об интерпретируемости самой модели.

Постепенно внимание сообщества переключается на более новые парадигмы, однако свёрточные сети всё ещё показывают лучшее качество в области анализа изображений и плотно встроены в промышленную разработку. Помимо прочего, хоть и в задаче анализа последовательностей их потеснили трансформеры, свёрточные сети широко используются при исследовании многомерных объектов с локальными свойствами, например, графов.

## Список литературы

- [1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, 12 1989.
- [2] Goutam Das, Nurul Anwar, Shovan Chowdhury, and Kazi Rahman. Design and fabrication of an image processing based autonomous weapon. *International Journal of Engineering Research*, ISSN:2319–68902347, 12 2016.
- [3] Andrej Karpathy. Cs231n convolutional neural networks for visual recognition, 2021. [Online; accessed 31-May-2021].
- [4] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.
- [5] Graham Neubig. Neural networks for nlp, 2020. [Online; accessed 2-June-2021].
- [6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [7] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.
- [8] Kunlun Bai. A comprehensive introduction to different types of convolutions in deep learning, 2019. [Online; accessed 2-June-2021].
- [9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [10] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

- [11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [13] Muneeb ul Hassan. Vgg16 – convolutional network for classification and detection, 2018. [Online; accessed 1-June-2021].
- [14] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- [15] Chi-Feng Wang. A basic introduction to separable convolutions, 2018. [Online; accessed 2-June-2021].
- [16] Yani Ioannou. A tutorial on filter groups (grouped convolution), 2017. [Online; accessed 2-June-2021].
- [17] Eli Bendersky. Depthwise separable convolutions for machine learning, 2018. [Online; accessed 2-June-2021].
- [18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
- [19] Geoff Hinton. Aaaai 20 / aaai 2020 keynotes turing award winners event / geoff hinton, yann le cunn, yoshua bengio, 2020. [Online; accessed 2-June-2021].
- [20] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829*, 2017.
- [21] Kamalesh Palanisamy, Dipika Singhania, and Angela Yao. Rethinking cnn models for audio classification. *arXiv preprint arXiv:2007.11154*, 2020.
- [22] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

- [23] Bing Yu, Mengzhang Li, Jiyong Zhang, and Zhanxing Zhu. 3d graph convolutional networks with temporal graphs: A spatial information free framework for traffic forecasting. *arXiv preprint arXiv:1903.00919*, 2019.