———————1 слайд———————

## **Что такое компьютерное зрение**

Компьютерное зрение (computer vision, CV) — это область искусственного интеллекта, занимающаяся автоматизацией задач, связанных с анализом, распознаванием и пониманием визуальных данных. Этому направлению удалось добиться успехов благодаря глубокому обучению (deep learning), использующему сложные архитектуры нейронных сетей для решения задач распознавания объектов, классификации, сегментации и других.

Основная цель компьютерного зрения — научить компьютер воспринимать и интерпретировать изображения или видео так, как это делает человек.

———————2 слайд———————

## **Основные задачи компьютерного зрения**

Компьютерное зрение решает четыре основные задачи:

1. **Классификация**: отнесение объекта на изображении к определённому классу (например, "кошка" или "собака").
2. **Локализация**: определение положения объекта на изображении (обычно выделяется ограничивающая рамка).
3. **Детекция** **объектов**: нахождение и классификация всех объектов определённых классов (например, пешеходов на улице).
4. **Сегментация**: разметка изображения на уровне пикселей, чтобы выделить области, принадлежащие к одному классу.

Сегментация делится на два типа:

* **Семантическая** сегментация: пиксели, относящиеся к одному классу, помечаются одинаково.
* **Инстансная** сегментация: различаются отдельные объекты внутри одного класса.

———————3 слайд———————

### Основные типы нейронных сетей и их задачи

1. **Полносвязные нейронные сети (Fully Connected Networks)**

В таких сетях каждый нейрон в одном слое соединён со всеми нейронами в следующем. Эти сети часто применяются для задач классификации данных, где входные данные имеют фиксированную размерность. Их обычно используют для задач обработки табличных данных, прогнозирования или анализа данных, где пространственные связи не играют существенной роли. Полносвязные сети плохо подходят для обработки изображений, поскольку они не учитывают пространственные зависимости между пикселями, что критично для понимания структуры изображения.

1. **Рекуррентные нейронные сети (RNN)**

RNN обрабатывают последовательные данные (например, текст или временные ряды), передавая информацию от одного временного шага к другому. Они эффективны для задач, связанных с временной последовательностью, таких как распознавание речи, перевод текста и обработка временных рядов. RNN редко используются для анализа изображений, поскольку не могут эффективно учитывать пространственные взаимосвязи, которые важны для распознавания объектов и их особенностей на изображении.

1. **Свёрточные нейронные сети (Convolutional Neural Networks, CNN)**

CNN используют свёртки (то есть фильтры), чтобы выделять иерархические признаки на изображении — от простых, таких как контуры, до сложных, таких как формы и текстуры. Они являются стандартом для задач обработки изображений и подходят для классификации, детекции объектов и сегментации изображений. CNN хорошо работают с пространственными зависимостями, обладают высокой точностью в распознавании объектов и их особенностей и поддерживают многослойные структуры для последовательного извлечения признаков.

1. **Сети генеративного типа (Generative Networks)**

Генеративные сети, такие как вариационные автоэнкодеры и генеративно-состязательные сети (GANs), созданы для генерации новых данных, основанных на распределении существующих данных. Эти сети применяются в задачах генерации изображений, видеоданных, воссоздания недостающих частей изображений или улучшения их разрешения.

Сверточные нейронные сети (CNN, Convolutional Neural Networks) обычно применяются для задач обработки изображений и распознавания объектов, поскольку они лучше справляются с анализом пространственных и локальных признаков на изображениях по сравнению с обычными полносвязными нейронными сетями.

———————4 слайд———————

## **Технологии компьютерного зрения**

Первым делом нейросеть должна изучить изображение. Свёрточные нейросети работают с помощью фильтра, который называют ядром свёртки.  
  
Ядро свёртки — это квадрат с цифрами в окошках, в основном 3х3, но может быть 5х5 или 7х7. Числа внутри ядра нейросеть подбирает сама — в этом заключается процесс её обучения. Процесс свёртки — это движение ядра из левого верхнего угла изображения до правого нижнего, слева направо и сверху вниз.  
  
Если взять в качестве примера ядро свёртки размером 3х3, то более детально этот процесс происходит так:  
  
1. Цифры свёрточного фильтра накладываются на область из 9 пикселей и перемножаются друг на друга поэлементно, а затем эти результаты складываются. Итоговое число попадает только в центральный пиксель, поэтому в итоговом изображении по краям отрезается по одному пикселю.

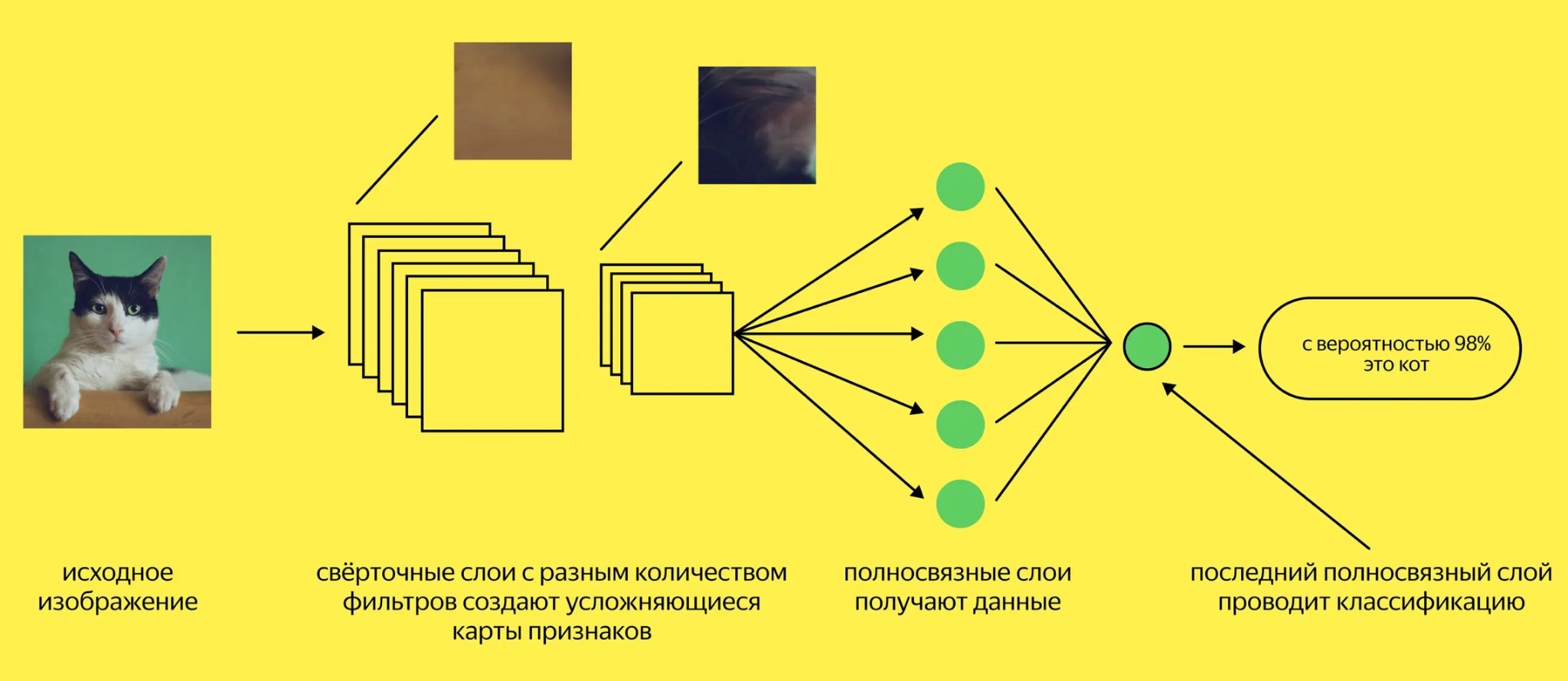
———————5 слайд———————

Стоит отметить, что в случае rgb картинки единственное отличие будет в трехмерном сверточном ядре

2. Если работает фильтр, который «высвечивает» границы, то границы появляются на стыке положительных и отрицательных значений. Положительное число получается в тех областях, где слева изображение ярче, чем справа, а отрицательное, наоборот, — справа ярче, чем слева. Однотонные области обозначаются нулём.  
  
3. Затем по изображению «пробегают» другие свёртки, с другими числами. Их свёрточные фильтры определяют горизонтальные и вертикальные границы, градиенты, текстуры и цвета. А более сложные верхнеуровневые признаки, например кошачье ушко, улавливаются уже в последующих свёрточных слоях. Главная задача — постепенно выделить значимые для нейросети черты. Размер ядра и количество фильтров в каждом свёрточном слое нейросети закладывает разработчик, когда задаёт архитектуру этой нейросети.  
  
4. После работы свёрток остаётся набор карт признаков (feature maps). Например, он может выглядеть как тёмный фон с высвеченными белыми чертами объекта. Теперь это оцифрованные признаки, которые компьютер сможет распознать.

———————6 слайд———————

**Дальнейшие действия нейросети зависят от задачи:**  
  
● Провести классификацию. Тогда данные о признаках с помощью дополнительных слоёв нейросетей «вытягиваются» в вектор — одномерный массив, то есть ряд чисел. После этого его можно будет отдать в полносвязный слой на выходе нейронной сети. В последнем полносвязном слое определяется вероятность принадлежности изображения к тому или иному классу. использовать можно например софтмакс



Если перед нейросетью стоит задача классификации, то карты признаков передаются в полносвязные слои. На иллюстрации нейросеть выполняет бинарную классификацию: отличает кота от, например, собаки. Поэтому на финальном этапе остаётся один нейрон, который определяет вероятность принадлежности к одному из классов

● Провести детекцию. Детекция объектов — задача, в которой нейросеть определяет не только принадлежность изображения к определённому классу, но и локализует объекты на изображении. В результате нейросеть выдаёт bounding box (ограничивающую рамку) вокруг каждого объекта, а также его класс. Для решения этой задачи свёрточные нейронные сети (CNN) объединяются с архитектурами, адаптированными под детекцию. **FCN** тоже используется в конечных слоях сети, например, для предсказания координат рамок или классификации объектов.

———————7 слайд———————

● Провести сегментацию. Чтобы размыть фон или нарисовать маску на изображении, полносвязные слои не нужны. Эту задачу берёт на себя свёрточная нейронная сеть с другой структурой, где вместо полносвязных слоёв — деконволюционные свёрточные слои. В общих чертах эти слои проводят операцию, обратную процессу свёртки.

Рассмотрим этапы архитектуры U-net:

1. **Энкодер (Encoder) — "Сжатие":**
   * Цель: Выделение иерархических признаков изображения.
   * Состоит из нескольких блоков свёрток и операций понижения размерности (обычно **max pooling**).
2. **Боттлнек (Bottleneck):**
   * Центральная часть сети, где данные имеют минимальное пространственное разрешение, но максимально богатое представление признаков.
3. **Декодер (Decoder) — "Обратное восстановление":**
   * Цель: Восстановление пространственного разрешения до исходного размера.
   * Состоит из:
     + **Деконволюций (transposed convolutions)** или **upsampling** для увеличения размерности изображения.
     + **Слияния (skip connections)** с соответствующими слоями энкодера.

**Skip Connections:**Они объединяют данные из соответствующих уровней энкодера и декодера. Это помогает декодеру восстанавливать детали, утерянные при понижении разрешения.

(на рисунке это горизонтальные пунктирные стрелочки)

1. **Выходной слой:**
   * Последний слой — это свёртка, которая выдаёт карту сегментации, где каждому пикселю соответствует предсказание класса.